Finding the Trade-offs between Model Size and Performance in Lightweight ResNet-18 architecture for ECG Classification

University of Turku
Department of Computing
Master of Science Thesis
Health Technology
June 2025
Syed Faizan Ahmed

Supervisors:
Tuukka Panula
Usman Azmat

UNIVERSITY OF TURKU
Department of Computing

Syed Faizan Ahmed: Finding the Trade-offs between Model Size and Performance in Lightweight ResNet-18 architecture for ECG Classification

Master of Science Thesis, 51 p., 2 app. p.
Health Technology
June 2025

Deep learning approaches have shown remarkable success in electrocardiogram (ECG) classification, but their computational costs are often a bottleneck for deployment on low-resource platforms such as mobile health initiatives and edge devices. The thesis addresses the challenge by investigating exhaustively the trade-offs between performance and computational cost through architectural design modifications to a ResNet-18 base model for multi-label ECG classification. Using the extensive combined dataset of PhysioNet/Computing in Cardiology Challenge 2020 and Shandong Provincial Hospital, we conducted a series of 18 experiments to study the impact of layer ablation, architectural simplification, and their combinations.

Our findings indicate that extreme compression of the model is achievable with a negligible reduction in performance. We managed to compress model size by up to 99.2% (from 33.78 MB to as low as 0.28 MB) without sacrificing useful diagnostic performance. Interestingly, most of the smaller models, like Experiment E17 (2.24 MB, 93.4% compression), not only survived but outperformed the baseline model, with micro AUROC and micro precision of 0.88 and 0.45, respectively, compared to the baseline's 0.86 AUROC and 0.39 precision. This seemingly counterintuitive effect is attributed to factors such as reduced overfitting, enhanced architectural efficiency, superior signal-to-noise ratio, and superior optimization landscape in the smaller models.

Among the major effective compression techniques are substituting normal convolutions with depthwise separable convolutions, extensive reduction of feature maps, and removal of redundant layers selectively. These results counteract the notion that larger models equal better, showing how well-optimized small architectures are more efficient for some tasks like ECG classification. The contribution of this study is significant, enabling high-performance ECG analysis models to be implemented on hardware with limited resources, enabling real-time monitoring, and enabling the sustainability of AI in healthcare. The research offers empirical proof of the feasibility of highly efficient deep learning solutions, enabling broader accessibility and utilization of AI in clinical settings.

Keywords: Deep Learning, ECG Classification, Model Compression, ResNet, Efficient AI, Resource-Constrained Devices, Healthcare AI, Biomedical Signal Processing, Neural Network Optimization, Feature Map Reduction, Depthwise Separable Convolutions

Contents

List of Figures

List of Tables

1 Introduction

In recent years, deep learning has demonstrated compelling performance across many different applications, including medical diagnosis, where models are increasingly being utilized to solve advanced tasks such as electrocardiogram (ECG) classification [1]. Convolutional neural networks (CNNs), in particular, have been demonstrated to excel at learning hierarchical representations automatically from unprocessed ECG signals so that cardiac abnormalities can be detected reliably without hand-tuned feature engineering. But such performance is usually obtained at the cost of very high computational complexity and model size, which makes deployment in practical applications challenging, particularly in resource-constrained environments like mobile monitor devices or edge-based healthcare systems [2].

This trade-off between predictive power and computational efficiency has increased interest in model compression and lightweight architectures. Several techniques have been proposed to cut neural networks' memory footprint and inference delay without significantly compromising accuracy. These include weight pruning [3], quantization [4], knowledge distillation [5], and efficient architectural designs [6]. Among these, layer ablation, i.e., the process of systematically removing or altering components of a neural network, offers a relatively transparent and intuitive way to analyse the importance of different layers and structures within a model

[7]. It is also a helpful framework to evaluate how architectural simplifications can impact model performance and size.

This thesis investigates the effects of layer ablation and architectural simplification on model size and performance in the context of a ResNet-based deep learning model designed for ECG classification. The aim is not to optimize accuracy but to extract practical insights into which parts of the architecture contribute most to size, and how performance degrades (or improves) as layers are removed, reduced, or modified. Experiments include the removal of layers such as batch normalization and dropout, replacement of standard convolutions with depth wise separable convolutions [8], reduction in the number of residual blocks, halving feature map channels, and other size-affecting changes.

The ECG data used within this work derives from two public, large-scale datasets stored on PhysioNet to achieve real-world heterogeneity and broad clinical representativeness. The first is the PhysioNet/Computing in Cardiology Challenge 2020 dataset [9], comprising 12-lead ECG records taken from over 43,000 patients across different hospitals and continents. All the recordings are annotated with SNOMED CT codes, which cover a wide variety of rhythm disorders, conduction defects, and structural cardiac disorders. All recordings have varying lengths (6 to 60 seconds) with a sampling rate of 500 Hz, adding to the density and complexity of data. The second one is the large-scale multi-label 12-lead electrocardiogram database [10] from Shandong Provincial Hospital (SPH). It complements the first dataset and supports effective model training for arrhythmia detection. The datasets together offer a comprehensive benchmark for testing ECG classification models, particularly concerning generalization and real-world applicability.

To perform experimentation, this paper builds upon an openly shared ResNet-

based ECG classifier from the University of Turku [11]. This baseline offers a practical implementation for multi-label classification on the PhysioNet data and serves as the foundation for all architectural ablation and modification experiments conducted on it.

The rest of this thesis is structured as follows: the next chapter has a literature review section related to model compression, efficient architectures, and ECG classification. This is followed by the Methodology, where the experimental setup and ablation strategy are described. The Implementation section describes how the model changes were implemented. The Results section details performance and size measures across variations. Lastly, the Discussion elaborates on these findings, and the thesis concludes with an overview and some potential future research.

This thesis employed artificial intelligence (AI) tools for assistance work, and any unique intellectual work is my own contribution. ChatGPT was utilized to refine sentences to enhance written draft readability. Grok's DeepSearch and Consensus AI assisted in finding relevant papers in medical query-based response systems, retrieval-augmented generation, and prompt engineering for the Literature Review. Apart from these supporting roles, editing of text, and search for literature, all else in research design, experimentation, analysis, and conclusions is my original work, as fully adhering to the academic standards of integrity of the University of Turku.

2 Literature Review

This literature review aims to provide a summary of what is currently understood about deep learning for ECG classification with a specific focus on managing model size and computational performance challenges. It will cover the various DL architectures employed for ECG analysis, the primary approaches to model compression, and survey the development of efficient network architectures available for application in this area. Additionally, the review will examine the specific application and compression of Residual Networks (ResNet), a very prevalent architecture in computer vision and more recently in ECG analysis and connect it to the thesis work scope outlined in the provided introduction. The role and procedure of layer ablation studies in simplifying and interpreting deep learning models will also be discussed. By synthesizing the existing state of the art in these allied areas, this review seeks to establish the state-of-the-art, identify existing trends and challenges, and highlight the significance of research toward the realization of accurate and yet computationally feasible DL methods to ECG-based cardiovascular diagnosis, and ultimately situate the specific study on layer ablation and architectural simplification of a ResNet-based ECG classifier.

2.1 Deep Learning for ECG Classification

Deep learning (DL) has transformed the landscape of electrocardiogram (ECG) analysis by offering powerful tools for automating the detection and classification of various cardiac abnormalities, most notably arrhythmias. Conventional ECG interpretation techniques tend to be manual feature extraction from morphological features (such as P-wave, QRS complex, T-wave durations and amplitudes) and rhythm analysis, which is time-consuming, expertise-demanding, and may be subject to inter-observer variation [12] DL models, by contrast, possess the remarkable ability to automatically learn intricate patterns and hierarchical representations straight from raw or minimally processed ECG signals, with minimal or no requirement for hand-engineered feature extraction [1], [12]. This capacity has achieved incredible advancement in developing powerful and accurate frameworks for everything from heartbeat classification (i.e., classification of normal beats, premature ventricular contractions, atrial premature beats) to classification of challenging arrhythmias (like atrial fibrillation, atrial flutter, bundle branch blocks) and the detection of myocardial infarction or other structural heart disease [12], [13].

Several DL architectures were created and extended for ECG classification, all of which carry strengths and suitability for certain properties of ECG signal analysis. Convolutional Neural Networks (CNNs) are a particularly dominant architecture in the aftermath of their success in computer vision. For ECG analysis, both one-dimensional (1D) CNNs, which process the ECG signal as a time series, and two-dimensional (2D) CNNs, which often operate on transformed representations of the ECG signal (like spectrograms, wavelet transforms, or beat-to-beat correlation matrices), have been successfully employed [1], [12]. 1D CNNs excel

at capturing local temporal patterns and morphological features in a single heartbeat or short segments, while 2D CNNs can utilize spatial hierarchies in image-like representations. CNNs' ability to learn translation-invariant features makes them robust to slight variations in signal timing or morphology [14].

As ECG signals are sequential, the variants of Recurrent Neural Networks (RNNs), i.e., Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), have also been used widely. These structures are inherently well-suited to the modelling of temporal context and dependencies through time and thus are particularly well-suited to detecting long-range patterns in heart rhythms that can last several heartbeats, which is relevant in arrhythmia diagnosis with rhythm irregularities (e.g., atrial fibrillation) [12]. GRUs and LSTMs, with their gates, alleviate the vanishing gradient problem inherent in simple RNNs and are more able to learn long-term dependencies. But RNN-based models are sometimes computationally more intensive than CNNs, particularly during training [12].

Subsequently, Transformer models, initially presented to handle natural language, have shown promise for use in ECG processing. Thanks to self-attention mechanisms, Transformers can model dependencies between different points in the ECG sequence regardless of their distance and can therefore capture complex temporal interactions better than RNNs in some situations [12]. Hybrid configurations, which combine the aspects of CNNs and RNNs/Transformers (e.g., CNN layers for feature extraction followed by LSTM layers for sequence modelling), are also common, trying to utilize the strengths of different architectures [15]. Other types of architectures, including Deep Belief Networks (DBNs), composed of Restricted Boltzmann Machines (RBMs) and standard Multilayer Perceptrons (MLPs) have also been investigated, although CNNs and RNN types are currently the latest

pioneers [12].

The training and testing of such DL models hinge on obtaining access to large, well-annotated ECG datasets. Multiple public repositories have played a key role in propelling the field to its current position. MIT-BIH Arrhythmia Database, although diminutive by today's standards, is still an old standard. More recent, larger collections archived on sites such as PhysioNet have become an indispensable resource. They include PhysioNet/Computing in Cardiology Challenge datasets like the 2017 AF classification challenge, and the 2020 challenge to classify diverse abnormalities from 12-lead ECGs utilized in the thesis context [9], Large-scale Multi-Label 12-lead ECG Database for Arrhythmia Study [10], Such data sets provide scale and diversity on which to train robust DL models and benchmark relative performance over wide ranges of cardiac conditions and patients, facilitating attainment of clinically translatable AI-assisted ECG analysis.

2.2 Model Compression Techniques for Deep Neural Networks

The exceptional success of deep learning models across numerous applications like computer vision and signal processing, including ECG processing, has predominantly been accompanied by an enormous increase in model complexity. The highest-performing architectures usually consist of millions or even billions of parameters, which necessitates substantial storage requirements (memory footprint) and computationally expensive inference [16]. This complexity presents a substantial hurdle to deploying these powerful models on resource-limited platforms,

such as mobile phones, wearables, embedded systems, and edge computing nodes that are characterized by having low memory, computation resources, and energy budgets [2], [13]. In bridging this gap between model capacity and deployment possibility, a broad range of methods as a collective known as model compression have attracted considerable research attention. The primary goal of model compression is to reduce the size and/or computational needs of a trained neural network without causing much devastation to its predictive capability [17]. These techniques are crucial for enabling real-time inference, reducing power consumption, and enabling deep learning to be used in real-world applications, for example, mobile health monitoring systems for ECG analysis.

Many major categories of model compression algorithms have been introduced, each attempting to address varying aspects of redundancy in the model. One of the most comprehensively studied approaches is **Weight Pruning** [3]. The general idea here is that certain deep neural networks are over-parametrized with an excessively high number of weights or connections that make relatively small contributions towards the end result and can be safely removed with negligible loss in performance [16]. Pruning methods can be broadly classified as unstructured and structured pruning. Unstructured pruning removes individual weights based on certain conditions, like magnitude (small magnitude weights are less significant) or sensitivity, leading to sparse weight matrices that have to be executed on special hardware or software libraries for efficient execution. Structured pruning, however, removes entire structures like filters, channels, or even layers, creating smaller, denser models that can easily be accelerated on typical hardware [16]. There are a number of criteria and methods for selecting and removing these structures, often including iterative pruning and fine-tuning rounds to recover any lost accuracy [3].

Another highly effective compression technique is **Parameter Quantization**. It aims at reducing the numerical precision used to represent the model's weights and, in most cases, its activations during inference [4], [16]. Deep learning models are typically trained with 32-bit floating-point (FP32) precision. Quantization represents similar FP32 values in lower-bit forms, such as 16-bit floating-point (FP16), 8-bit integers (INT8), or even lower bit-widths (such as 4-bit, binary). Such precision loss comes at the cost of major model size reduction (e.g., INT8 reduces size by approximately 4x compared to FP32) and is capable of radically accelerating inference time, especially on hardware that supports low-precision math, and reducing power consumption [4]. Quantization may be performed post-training (Post-Training Quantization, PTQ), where an FP32 pre-trained model is quantized to lower precision, typically requiring a small calibration set. Alternatively, Quantization-Aware Training (QAT) simulates the quantization effect during training itself, typically resulting in better accuracy retention, especially for very low bit-widths, but at the cost of more advanced training [16].

Knowledge Distillation (KD) offers a different paradigm for accessing small models [5], [16]. Within KD, knowledge from a massive, complex, and high-performing model (teacher) is distilled into a smaller, but more efficient model (student). The student model not only learned to approximate the ground truth labels (hard targets) but also to approximate the output distribution (soft targets) or intermediate feature representations of the teacher model. By learning from richer information in the teacher's outputs or internal states, the student model can generally be considerably more accurate had it only been trained on ground truth labels alone, effectively capturing the knowledge of the larger model in a smaller one [5]. Various KD methods exist that target logit matching, feature

map matching, or relational knowledge between the various components of the network.

Finally, **Low-Rank Decomposition** (also known as Low-Rank Factorization) methods attempt to reduce redundancy in weight matrices or tensors of a neural network [16]. Very large weight matrices, particularly those in fully connected layers or even in convolutional layers, typically can be approximated as the product of some smaller matrices (i.e., factorized into lower-rank factors).

These model compression techniques can also be used in combination to achieve additional reduction in model size and complexity. A model can be pruned initially, with quantization being applied subsequently, or the lean architecture further compressed using quantization or KD. The choice and selection of algorithms are based on application requirements, intended target hardware platform, and acceptable level of loss between compression and degradation in accuracy. Understanding the range of various techniques is unavoidable when developing an efficient deep learning method for deployment to real-world devices in resource-constrained applications such as edge-based ECG analysis. The choice and selection of algorithms are based on application requirements, intended target hardware platform, and acceptable level of loss between compression and degradation in accuracy. Understanding the range of various techniques is unavoidable when developing an efficient deep learning method for deployment to real-world devices in resource-constrained applications such as edge-based ECG analysis.

2.3 Layer Ablation as a Model Analysis and Compression Tool

Understanding the internal workings of deep neural networks, which are traditionally characterized as "black boxes," is necessary to improve their design, identify faults, and create trust in their predictions. Ablation studies are a simple and traditional method of evaluating these complex models and quantifying the contribution of their components [18]. The term "ablation," from biology where tissues are excised using surgery, in AI is when one deactivates or removes successively pieces of a model, such as layers, connections, channels, features, or parts of training data, and quantifying the impact on model performance or its behaviour [19]. By comparing the performance of the ablated model to that of the original, researchers can deduce what value and role the removed piece serves.

Though quantization and pruning take care primarily of parameter reduction, layer ablation is structural: eliminating or tweaking architectural parts methodically to estimate their impact on performance [7]. Ablation studies, common in natural language processing and computer vision, offer insight into network redundancy that enables scientists to identify and delete non-vital components.

Zeiler and Fergus [7] first introduced visualization-driven ablation studies, showing how different convolutional layers affect network output on image classification. Ablation has recently been employed to investigate the roles of residual connections [20], normalization layers [21], and even single groups of neurons [22].

Ablation in medical AI is employed for both the purposes of enhancing interpretability and guiding model simplification. By monitoring performance degradation or stability when layers like batch normalization, dropout, or SE (Squeeze-

and-Excitation) modules are removed, researchers can determine essential vs. secondary structures. This informs model design for efficiency as well as for robustness, especially in high-stakes domains like healthcare.

2.4 The Role of Squeeze-and-Excitation (SE) Layers in Deep Networks

Hu et al. [23] presented squeeze-and-excitation (SE) blocks as a lightweight attention method that explicitly models interdependencies between channels to adaptively recalibrate channel-wise feature responses. SE blocks have been demonstrated to increase classification accuracy with a slight increase in computational cost when smoothly integrated into pre-existing architectures like ResNet.

The SE block operates in two main stages: a squeeze operation that aggregates global spatial information into a channel descriptor and an excitation operation that fully captures channel-wise dependencies. Although SE modules typically add only a small number of additional parameters relative to the backbone network, their cumulative impact on model size and computational burden becomes significant in resource-limited settings [24].

Consequently, SE layer ablation may result in a reduction in model size, as has been witnessed in this thesis. Several studies have been carried out examining the trade-off between the improvement in performance provided by SE blocks and their increased overhead. For instance, Hu et al. [23] showed that although SE blocks enhance performance on various models (ResNet, Inception, etc.), it is acceptable to eliminate them when slight drops in accuracy can be tolerated to

achieve significant gains in efficiency, especially for deployment on the edge [21].

With ECG classification, where noisy datasets and subtle fluctuations exist, channel-wise recalibration would be beneficial. Nevertheless, in real-world scenarios where lightweight models are emphasized, eliminating the SE layer is a valuable technique.

2.5 Depth-wise Separable Convolutions for Model Efficiency

Depth-wise separable convolutions, originally introduced in MobileNet [6] and Xception [8] structures, are a powerful mechanism for reducing the model size and computation complexity at relatively minor performance cost. While common convolutions filter and add inputs simultaneously, depth-wise separable convolutions separate them into two single, simpler operations: a depth-wise convolution using one filter across all input channels and a pointwise convolution (1x1 convolution) that sums the output.

Chollet [8] demonstrated how replacing with depth-wise separable convolutions could reduce the number of parameters and operations by close to an order of magnitude. That insight paved the way for a wave of interest in transferring similar strategies to domains beyond vision, including sequential and medical data analysis.

In clinical AI applications such as ECG classification, where input signals have strong spatial (temporal) patterns with comparatively low visual complexity compared to natural images, depth-wise separable convolutions can maintain adequate

feature extraction ability with much less resource requirement [25].

2.6 ResNet Architecture and its Application/Compression in ECG Analysis

Residual Networks, or ResNets, are a pivotal architecture in the evolution of deep learning, especially for computer vision tasks. Their impact has been seen in a wide range of fields, including the interpretation of physiological data such as ECG. Introduced by He et al., ResNets were designed to address the degradation problem observed in very deep neural networks, where adding more layers paradoxically led to higher training (and test) error [26]. The major innovation in ResNet is incorporating "residual blocks" with identity shortcut connections (or skip connections). Such connections allow the network to skip one or several layers and learn residual functions in relation to layer inputs, rather than directly learn unreferenced functions [27]. These residual blocks are stacked to create standard ResNet architectures (e.g., ResNet-18, ResNet-34, ResNet-50), which usually include convolutional layers, batch normalization, and ReLU activations [28].

However, as with other deep models, the standard ResNet models are potentially computationally expensive and parameter-heavy, particularly the deeper ones (ResNet-50, ResNet-101, etc.) [16]. This is the motivation for compressing and simplifying ResNet architectures, especially for ECG analysis, without sacrificing their high accuracy to make them deployable in resource-limited settings. Numerous approaches that are in alignment with general model compression techniques have been investigated. To reduce the network's width, structured pruning

techniques have been used, such as channel pruning or filter pruning in leftover blocks [29]. ResNets can also be quantized using PTQ or QAT to lower the activation precision and weight [30]. Knowledge distillation can be utilized to train small ResNet variants (e.g., ResNet-18) from a large one (e.g., ResNet-50) as a teacher [5]. A review of these efforts contextualizes the specific modifications (block reduction, feature map reduction, SE/BN elimination, convolution replacement) explored in the thesis, putting them within the wider framework of efficient ResNet architecture for ECG analysis.

2.7 Feature Map Reduction for Lightweight Networks

Feature map reduction, or channel reduction, is one of the simplest and most efficient methods employed to scale down the memory and computational requirements of CNNs [31]. In this method, the number of channels (or filters) employed in the convolutional layers is decreased, lowering the number of parameters, and it can reduce network redundancy, storage space, and network complexity up to 70% [31]. Reducing these filters provides immediate efficiency gains because the number of filters in convolutional layers increases the model size and runtime memory footprint [26].

To capture increasingly abstract patterns in the data, feature maps act as intermediary representations. A larger model size and slower inference are the results of having many channels, even though they enable the model to learn more intricate or detailed patterns [31]. The slight performance benefits provided by

deeper or wider models are not always worth the extra computational expense in many real-world applications, particularly in healthcare settings like mobile ECG monitoring [32]. Compact devices with fewer channels, on the other hand, can frequently provide almost identical performance while conserving a substantial amount of energy and memory [32].

But there are also trade-offs. Significant reduction of feature maps can behave as an information bottleneck, particularly in deeper layers where high-level representations are built [33]. This may lead to issues like underfitting, reduced generalization capability, noise sensitivity, or class imbalance, commonly present in clinical ECG databases [9], [34]. Feature map reduction needs to be informed by domain-specific expert knowledge and extensive testing.

Feature map reduction can also be synergistically combined with other architectural simplifications. For example, when paired with depth-wise separable convolutions, the combined architecture yielded a model as small as 2.2 MB while maintaining functional accuracy, demonstrating the effectiveness of multi-faceted simplification approaches [9], [35]. These strategies together can significantly aid in deploying ECG models in environments with limited processing power or storage capacity.

In summary, feature map reduction is a great and interpretable model compression technique for CNNs in medical signal processing. Used wisely, it can provide lightweight networks suitable for real-world deployment, particularly in edge and wearable healthcare applications [35].

2.8 Synthesis and Research Gaps

Deep learning for ECG classification has been shown to achieve enormous progress, allowing automatic systems to be precise in identifying an extensive set of cardiac anomalies. Models such as CNNs, RNNs (LSTMs/GRUs), and more recently, Transformers, which are usually applied in synergy in hybrid models, are capable of extracting significant features and temporal information from multidimensional ECG signals. But the pursuit of higher accuracy has been to deliver computationally costly models, which have become a primary bottleneck toward adoption in poor-resource settings such as point-of-care or wearables. The highlight of this review is to present the call for research to close the gap through effective network design and compression of models.

There is a wide range of methods available to address model efficiency. Weight pruning, parameter quantization, knowledge distillation, and low-rank decomposition are powerful methods to reduce the size and computational cost of existing large models. At the same time, innovation in natively light architectures, e.g., MobileNets, SqueezeNet, and others leveraging advancements such as depthwise separable convolutions and channel attention mechanisms (e.g., SE blocks), provide alternative paths to efficiency. ResNet, a high-performance building block architecture, has been a frequent casualty of these performance-enhancing modifications in the ECG domain, whose methods range from depth/width factorization to the employment of efficient convolutional blocks.

Ablation studies emerge as a central methodology throughout this domain, serving both analytic and constructive functions. Analytically, they allow complex models to be deconstructed, revealing the contribution of each element and

identifying redundancies. Constructively, ablation provides a simple method of architectural simplification through the ability to systematically remove or modify layers and assess the resulting models' performance-efficiency trade-offs, illustrated in the thesis experiments for architectural simplification of a ResNet model for ECG classification.

While significant progress has been made, some research gaps and challenges remain. While a very broad range of compression techniques exists, their optimal combination and application for ECG analysis models, particularly sophisticated ones like ResNets designed for 1D signals, require investigation. How different compression methods impact the noise or domain shift robustness of models in ECG is not known. Developing standardized benchmarks and protocols for evaluating efficiency alongside accuracy in the ECG context is crucial for fair comparison. Furthermore, whereas simplification through ablation simplifies models, rigorous methods to determine the optimal simplification approach (e.g., which layers or blocks to eliminate/multiply by to best balance accuracy-efficiency-size) remain improving. There is also a need for further research confirming the real-world clinical utility and robustness of compressed/efficient models executed on real-world edge devices, concerning factors other than offline accuracy metrics, e.g., real-time performance, energy consumption, and integration into clinical workflows.

Positioning the thesis research in this context, the investigation of layer ablation (SE layers, convolutional layers, residual blocks) and architectural modifications (feature map reduction, replacement of convolution) in a ResNet-based ECG classifier directly addresses the imperative need for effective deep learning models in this domain. Through evaluating the impact of these specific simplifications on the performance (accuracy, Area Under Receiver-operating characteristic or AU-

ROC/AUC) and efficiency (size of the model) for individual simplifications, the study contributes practical information to the understanding of ResNet redundancy in the analysis of ECG and practical knowledge on designing more efficient yet effective diagnostic models.

3 Methodology

This chapter outlines the methodology used to inspect the effect of layer ablation and architectural simplification on a ResNet-based deep learning classifier for ECG classification. The primary objective of this work is not to achieve the best accuracy but to systematically investigate the trade-offs between model size and performance. By conducting a series of controlled experiments on the deletion, reduction, or modification of various architectural components of a baseline ResNet-18 model, this work aims to provide practical insights into which aspects have the largest effect on model complexity and how their manipulation impacts diagnostic accuracy on ECG data.

3.1 Datasets

The selection of proper datasets is crucial for model training and performance evaluation of ECG classification models, both in the range of conditions included and in relation to real-world clinical cases. This work utilizes two publicly available large-scale 12-lead ECG datasets, as mentioned in the introduction to the thesis, to provide a good foundation for experimental studies.

The primary dataset used is the PhysioNet/Computing in Cardiology Chal-

lenge 2020 (CinC 2020) dataset [9]. It is a large dataset comprised of 12-lead ECG recordings of a heterogeneous population of over 43,000 patients acquired from multiple hospitals that are spread across different geographic locations. It is this heterogeneity that is important for developing models with the ability to generalize over heterogeneous populations of patients and recording conditions. Each tracing is annotated with SNOMED CT codes, which provide standardized diagnostic terms covering a wide variety of cardiac arrhythmias, conduction defects, and structural heart disease. The ECG signals themselves vary in length from 6 to 60 seconds and were sampled at a rate of 500 Hz. The richness and diversity of this dataset, with its varied signal lengths and complete annotations, make it a challenging and realistic test for the analysis of ECG.

In addition, the Shandong Provincial Hospital (SPH) Large-Scale Multi-Label 12-Lead Electrocardiogram Database [10] is used to enhance the training set and enhance the model's ability to identify a variety of heart diseases. By offering more examples of 12-lead ECGs, the data enhances the CinC 2020 data and expands the training set for arrhythmia diagnosis and other cardiac labels. 25770 ECG records from 24666 patients are included in the collection; they were obtained between 2019/08 and 2020/08. The duration of the record ranges from 10 to 60 seconds. Every ECG record's diagnostic statement complies completely with the AHA/ACC/HRS guidelines.

3.2 Data Preprocessing

Proper preprocessing of the ECG signals is important to remove noise, standardize formats, and prepare the data for optimal input into deep learning. Preprocessing

in this study was done through a Python script with the following significant steps applied to the merged PhysioNet Challenge 2020 and SPH dataset. It can read ECG data in MATLAB v4 (.mat) and HDF5 (.h5) files as well as their corresponding metadata in header (.hea) or comma-separated value (.csv) formats. The native sampling rate of a recording is extracted from its metadata.

A band-pass filter was applied to all ECG signals. This step is crucial in removing unwanted frequency components, for instance, baseline wander (low-frequency noise) and muscle artifacts or powerline interference (high-frequency noise), enhancing the signal-to-noise ratio and accentuating the physiologically relevant frequency ranges of the ECG signal. Following filtering, the ECG signals were resampled to a regular sampling frequency of 250 Hz. Cubic spline interpolation was used in carrying out this step. Having all recordings with a standard sampling rate is a common requirement for deep learning models that anticipate fixed-size input or comparable temporal resolution.

3.3 Data Splitting

The core of the splitting strategy depended on multi-label stratification. This is crucial in datasets where there are multiple potential labels per sample (such as common in ECG diagnoses), since it ensures the ratio of each label set is, to the extent possible, preserved throughout the subsets generated. This helps prevent situations where some rare conditions or label combinations end up getting over- or underrepresented in some specified set, possibly leading to skewed model training or testing. The script supported the creation of various dataset splits, which were saved as .csv files. These CSVs contained metadata like file paths to the pre-

processed ECG recordings and their corresponding multi-label annotations.

The primary test set was defined by the train_test_splits dictionary within the splitting code [11]. For example, the first split designated the CPSC and CPSC-Extra database as the test set, while databases like Georgia (G12EC), PTB, PTBXL, Chapman Shaoxing and Ningbo formed the training pool for this split configuration. This ensures evaluation on data from sources entirely unseen during the training of models for a particular split.

3.4 Baseline Model Architecture

The foundation for all ablation and architectural modification experiments in this thesis is a ResNet-based deep neural network, in the guise of an adaptation of the ResNet-18 architecture, taken from a publicly available ECG classifier developed at the University of Turku [11]. The ResNet-18 model, originally suggested by He et al. [26] for image classification, is characterized by its use of residual blocks that allow highly deep networks to be trained through prevention of the vanishing gradient issue. The baseline model adapts this architecture for 1D ECG signal classification. Key features and components of the baseline model used are:

- **Input Layer:** The model starts with an initial 1D convolutional layer of 64 output channels, a kernel of size 15, a stride of 2, and a padding of 7. It is then followed by Batch Normalization, ReLU activation, and a Max Pooling layer of a kernel size of 3 and a stride of 2.

- **Residual Blocks (BasicBlock):** The core of the network is formed by four consecutive layers, each constructed using BasicBlock modules. Two

copies of the same BasicBlock are employed per layer in the ResNet-18 case, thereby forming the [2, 2, 2, 2] pattern of ResNet-18. A **BasicBlock** has:

  - Two 1D convolutional layers (which use kernel size 7 and padding 3).
  - Batch Normalization after each convolutional layer.
  - A ReLU activation after the first Batch Normalization.
  - A Squeeze-and-Excitation (SE) layer followed by the second Batch Normalization in the block. The SE layer reweights channel-wise feature responses dynamically.
  - A dropout layer after the first ReLU activation.
  - A skip (residual) connection that adds to the block's input (identity) to its output. In case of non-matching dimensions due to striding or change in number of channels, a down-sampling operation (1x1 convolution with Batch Normalization) is applied on the identity path.

- **Channel Progression:** The number of output channels of the four major residual layers develops as follows: Layer 1 (64 channels), Layer 2 (128 channels), Layer 3 (256 channels), and Layer 4 (512 channels). Strides of 2 have been used at the beginning of Layer 2, Layer 3, and Layer 4 to reduce the temporal dimension.

- **Feature Fusion and Classification Head:** After the last residual layer, an Adaptive Average Pooling layer reduces each channel down to one feature. The model includes age and gender information as side input information. The demographic features are passed through an independent small fully

connected layer of 10 units as outputs. The summed ECG features and processed demographic features (10 units) are then combined and a final fully connected layer maps these concatenated features to the 'out_channel' dimension, i.e., the number of target ECG classes for multi-label classification. The Sigmoid activation function, necessary for multi-label classification, is applied in the training loop rather than being part of the model definition itself.

- **Initialization:** Convolutional layers are initialized using Kaiming Normal initialization, and Batch Normalization weights are initialized to 1 and biases to 0. The option for zero_init_residual is available, which, if true, initializes the last Batch Normalization layer in each residual block to zero, a technique suggested to improve training for very deep networks.

3.5 Training Protocol and Implementation

To allow for fair comparison between baseline ResNet-18 and all its ablated or altered variants, the training regime was the same across all experiments. This covers the training hyperparameters, the impact of training time (epochs), and the entire process of the experiments, including training implementation details.

The experiments were run using Python software package 3.10.4 with Torch version 1.13.1. Models were trained using a script, executed via Slurm batch jobs on the Puhti supercomputer provided by the CSC – IT Centre for Science Ltd.. A typical Slurm script specifies job parameters such as project account, time, CPU cores, memory, and GPU resources (e.g., one NVIDIA V100 GPU). The script saves the trained model and logs the results after parsing the given training

setup, instantiating the model, loading data, and running the training loop for the predetermined number of epochs. The experiments ran on 30 epochs. All the models were trained using the following key hyperparameters as shown in the table below:

Table 3.1: Training Hyperparameters

Category	**Details**
Training Data	CSV file defined by the data splitting process
Validation Data	CSV file derived from the training split for in-distribution validation.
Batch Size	256
Number of Workers	10
Epochs	30
Learning Rate	0.003
Weight Decay	0.00001
Optimizer	Adam (Defined in training script)
Loss Function	BCEWithLogitsLoss (suitable for multi-label classification tasks)
Output Activation	Sigmoid is applied in the training loop
Classification Threshold	0.5

3.6 Experimental Design: Architectural Modifications and Ablation Studies

The core of this thesis is the investigation of the effects of different architecture adjustments to the baseline ResNet-18 model on its performance and size for ECG classification. We created and conducted a set of experiments, each derived from a different network feature or component.

3.6.1 Layer Removal and Component Modification

Examples of such a set of experiments included the removal or replacement of standard layers in the ResNet-18 model to test their value to model size as well as performance:

- **Removal of SE Layer:** The `BasicBlock` includes an SE layer in the baseline. Removing this SE layer from all residual blocks was experimented with to evaluate its contribution to accuracy and model complexity.
- **Removal of Batch Normalization (BN) Layers:** BN layers are integral to ResNet. Experiments involved removing specific BN layers to observe the consequences on training stability and performance.
- **Removal of Convolutional Layers:** Some convolutional layers, such as `BasicBlock`'s `conv2`, were removed to analyze their impact on feature learning and accuracy.
- **Combined Removal:** Several experiments involved the combined removal of two or more components, e.g., removing the SE layer together with BN

and convolutional layers, to understand compound effects.

3.6.2 Architectural Simplification

These experiments were aimed at reducing the overall size and complexity of the model by simplifying the underlying structure.

- **Reduction in the Number of Residual Blocks:** The number of instances of `BasicBlock` in each of the four main layers of the ResNet was reduced. The experiments considered several configurations, for example, [1, 1, 2, 2], [2, 2, 3, 3], etc. These configurations were contrasted with the baseline ResNet-18 setup: **[2, 2, 2, 2]**.
- **Reduction in Feature Map Channels:** The number of feature maps (channels) in the convolutional layers was decreased, once by halving the number of feature maps and then reducing them by 75%.
- **Reduction in Kernel Size:** The kernel size in the first convolutional layer was reduced from 15 to 7.

3.6.3 Convolution Type Modification

This experiment focused on increasing computational efficiency through the use of alternative convolution.It involved using a depth-wise convolution followed by a point-wise (1x1) convolution. They were applied in the residual blocks in place of standard 3x3 convolutions (used in `BasicBlock` with a kernel size of 7).

All these experiments are based on recognized model compression literature to evaluate which parts of the ResNet model have the greatest potential for minimiza-

tion in the field of ECG classification without significantly compromising diagnostic value. The experiments table below provides a general overview of these tests and the results.

4 Results

A total of 18 distinct model configurations were evaluated, starting with the baseline ResNet-18 architecture (E1) and applying various modifications as described in the Methodology chapter. Table 4.1 provides a comprehensive reference of all experiments, including their assigned IDs, descriptions, model sizes, and performance metrics.

4.1 Relationship Between Model Size and Performance

A key finding of our experiments is that smaller models often maintained, or even improved performance compared to the baseline. Figure 4.1 illustrates the relationship between model size and micro AUROC.

Table 4.1: Experiment Results Table

Experiment ID	Description	Model Size (MB)	Micro AUROC	Macro AUROC	Micro Precision	Macro Precision	Size Reduction (%)
E1	Baseline Model	33.78	0.860	0.900	0.390	0.390	0.0
E2	Remove SE Layer (Squeeze-and-Excitation)	33.40	0.850	0.870	0.410	0.480	1.1
E3	Remove bn1	33.70	0.860	0.890	0.430	0.500	0.2
E4	Remove bn2	33.70	0.850	0.900	0.430	0.450	0.2
E5	Remove conv2	15.10	0.640	0.800	0.140	0.310	55.3
E6	Remove bn2 and conv2	15.10	0.700	0.860	0.220	0.420	55.3
E7	Remove bn2, conv2 and SE Layer	14.80	0.780	0.840	0.300	0.340	56.2
E8	Reduce residual blocks to [1, 1, 2, 2]	32.60	0.870	0.880	0.360	0.380	3.5
E9	Reduce residual blocks to [1, 1, 1, 1]	14.90	0.860	0.890	0.360	0.390	55.9
E10	Increase residual blocks to [2, 2, 3, 3]	51.40	0.800	0.890	0.320	0.430	-52.2
E11	Reduce kernel size to 7	33.70	0.850	0.890	0.390	0.460	0.2
E12	Replace 3x3 conv with Depthwise + Pointwise	5.93	0.880	0.870	0.490	0.450	82.4
E13	Halve number of feature maps	8.57	0.790	0.890	0.280	0.420	74.6
E14	Smaller Fully Connected Layer for Metadata	33.70	0.860	0.890	0.420	0.450	0.2
E15	Combine E7 + E12	2.84	0.850	0.880	0.450	0.460	91.6
E16	Combine E7 + E12 + E13	0.83	0.860	0.870	0.460	0.420	97.5
E17	Reduce feature maps by 1/4	2.24	0.880	0.900	0.450	0.470	93.4
E18	Combine E7 + E12 + E17	0.28	0.840	0.880	0.320	0.410	99.2

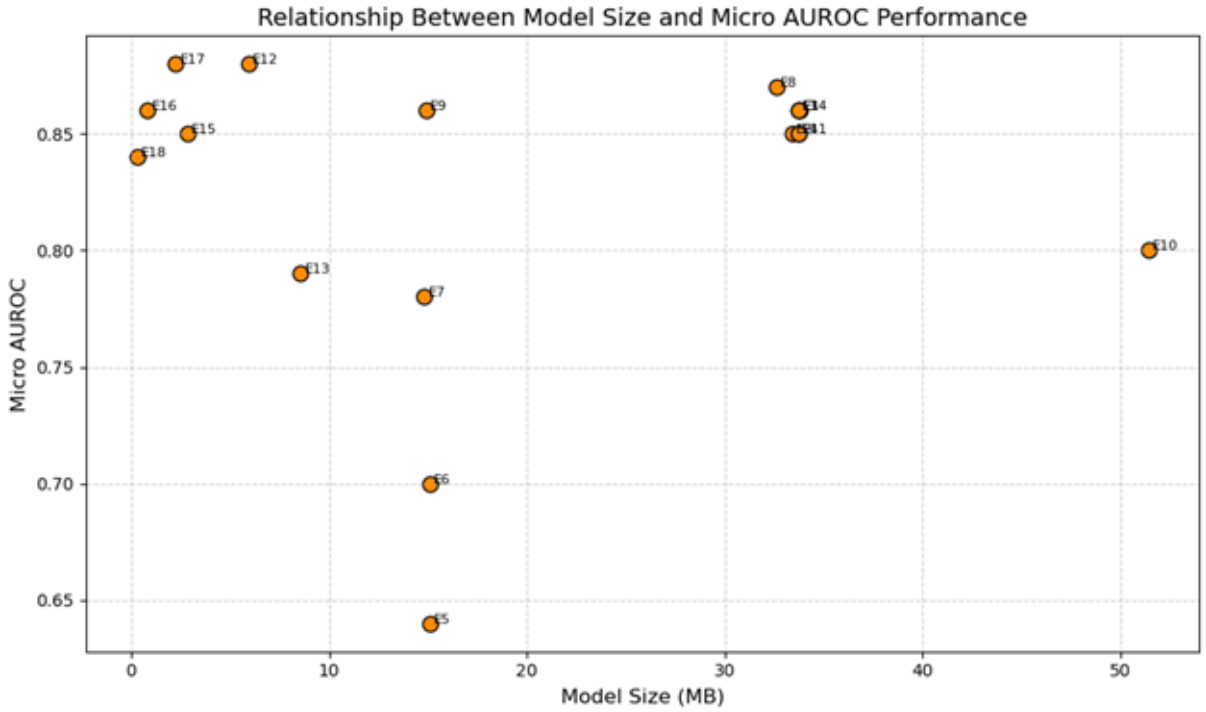

Figure 4.1: Relationship between model size and micro AUROC performance

Several interesting patterns emerge from the visualization in Figure 4.1. One major finding is the preservation of performance despite size reduction: a number of smaller models (notably E12, E16, and E17) maintained or even exceeded the baseline AUROC score while being significantly smaller. Furthermore, model size vs. performance is not an implicitly linear relationship. Some of the smallest models, such as E17 with a mere 2.24MB, recorded higher AUROC values (0.88) than the baseline (0.86). These gains are not permanent, since, there does appear to be a point of diminishing returns. To take an example, whereas E18 achieved a 99.2

The precision analysis reveals that smaller models (particularly E12, E15, E16, and E17) actually produced better results, with precision values ranging between

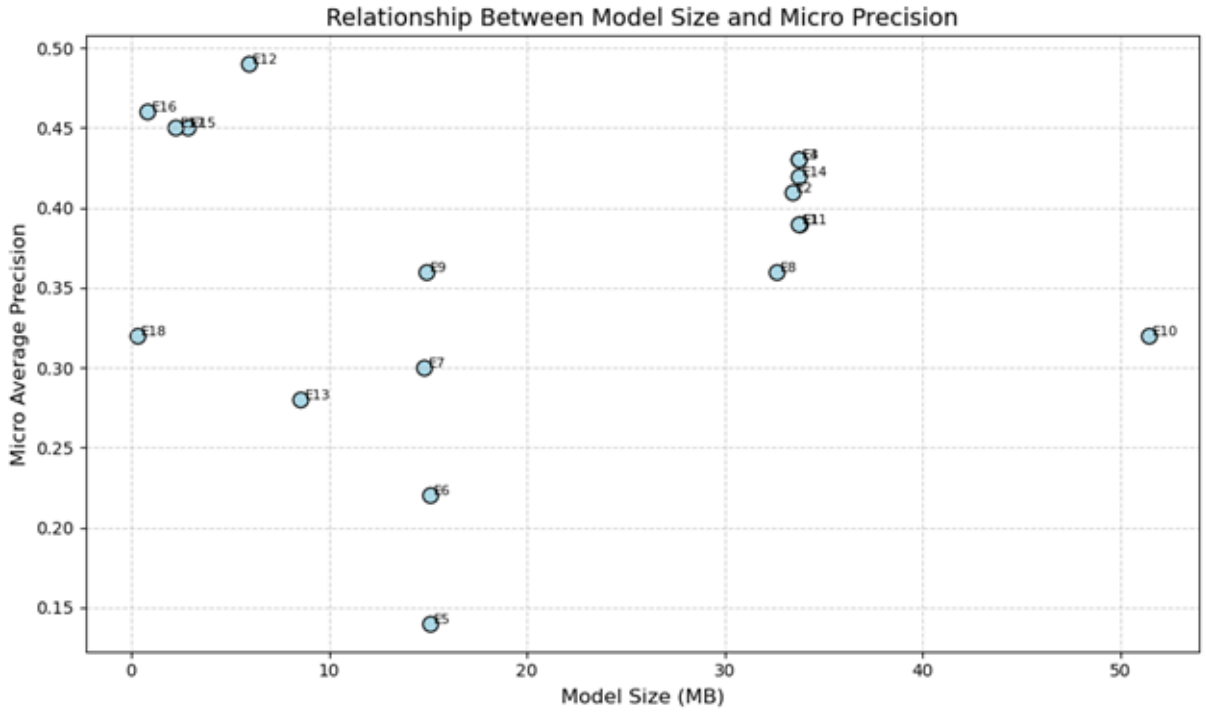

Figure 4.2: Relationship between model size and micro precision performance

0.45 and 0.49, compared to the baseline's 0.39. However, this improvement was not uniform across all models. Some trade-offs emerged—for instance, in E2, we observed an increase in precision (0.41 vs. 0.39) despite a slight drop in AUROC (0.85 vs. 0.86). Among all experiments, E12 and E17 stood out for striking an excellent balance by enhancing both AUROC and precision while drastically reducing model size.

This bar plot displays the relative behavior of micro and macro AUROC for each experiment, showing that, while the two values follow similar trends, there are minor differences, particularly for better-class balanced prediction models. For example, models E12 and E17 have somewhat improved macro AUROC behavior, indicating higher minority class performance.

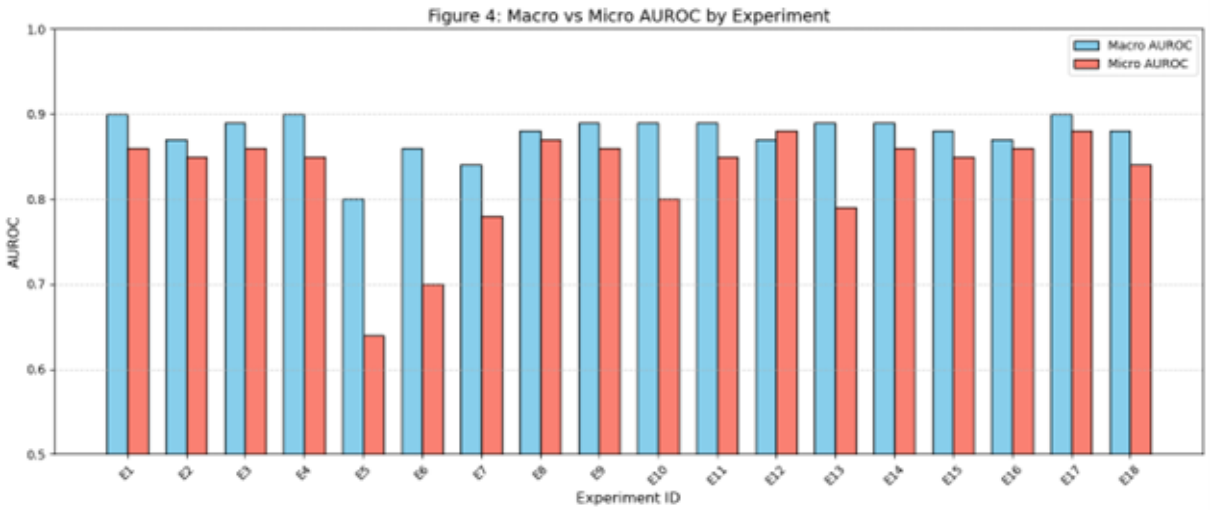

Figure 4.3: Comparison of Micro and Macro AUROC across all model configurations

4.2 Impact of Individual Layer Modifications

Our first set of experiments (E2–E7) focused on removing or modifying individual layers within the ResNet architecture. Removing batch normalization layers, as done in E3 and E4, revealed subtle yet notable differences. In E3, where bn1 was removed, AUROC remained consistent with the baseline at 0.86, but precision improved from 0.39 to 0.43. Similarly, in E4, the removal of bn2 led to a slight AUROC dip to 0.85, yet precision still increased to 0.43. Interestingly, these modifications had negligible effects on model size (less than 0.3%), suggesting that batch normalization has little effect on the model's memory footprint but can still influence specific performance metrics.

The impact of eliminating the second convolutional layer was greater. In E5, doing this alone produced a large 55.3% reduction in model size but also significantly impacted performance (AUROC: 0.64, precision: 0.14). But combining it with the elimination of the second batch normalization layer in E6 showed slightly

better performance (AUROC: 0.70, precision: 0.22), without changing the size reduction. The best outcome in these experiments was obtained by E7, in which the three layers—bn2, conv2, and the SE layer—were removed. This configuration moderately improved AUROC and precision (0.78 and 0.30, respectively) and increased reduction in size to 56.2%. These results indicate that while eliminating convolutional layers has a significant impact on model size reduction, special caution is needed while choosing sets of layers to eliminate in order to have negligible loss of performance.

In Experiment E2, we removed the Squeeze-and-Excitation (SE) layer to evaluate its impact. The change in model size was almost negligible—only about a 1.1% reduction. The AUROC dropped slightly, moving from 0.86 to 0.85. Interestingly, precision saw a modest increase, going from 0.39 to 0.41.This showed that the SE layer doesn't play a major role in determining model size but does appear to affect how the model manages attention and balances different performance metrics. The effect isn't very huge, but it's not insignificant either; the SE layer's contribution varies and it can influence specific outcomes even if the overall impact isn't overwhelming.

4.3 Impact of Architectural Simplifications

The second set of experiments (E8–E14) explored broader architectural modifications to the ResNet model.

4.3.1 Residual Block Configuration

Experiments E8, E9, and E10 explored the effects of changing the number and arrangement of residual blocks. Reducing the blocks to [1,1,2,2] (E8) led to a slight improvement in performance (AUROC increased from 0.86 to 0.87), while the model size decreased by only 3.5%. When the blocks were minimized to [1,1,1,1] (E9), the AUROC remained comparable (0.86), but this adjustment resulted in a substantial 55.9% reduction in model size.Increasing the blocks to [2,2,3,3] (E10) resulted in a lower micro AUROC and precision, even though macro performance metrics remained similar. As anticipated, this change also increased the model size by 52.2%. These findings show that, for ECG classification, using a more compact configuration of residual blocks can be just as effective, or even preferable, compared to larger architectures.

4.3.2 Kernel Size Modifications

Reducing the kernel size to 7 in E11 showed only a marginal impact. The model size decreased by 0.2%, AUROC reduced slightly to 0.85, and precision remained unchanged. This reinforces the idea that kernel size plays a limited role in model performance and efficiency in this specific context.

4.3.3 Convolution Type Modifications

Experiment E12 replaced standard 3×3 convolutions with depth-wise separable convolutions (depth-wise + point-wise). Since we modified the model structure, we achieved great results.The modification achieved a dramatic size reduction (82.4%), improving the AUROC (0.88 vs. 0.86) while at the same time increasing the preci-

sion as well (0.49 vs 0.39). This adjustment stands out as one of the most effective individual modifications. It demonstrates that depth-wise separable convolutions can simultaneously reduce model size and enhance performance in ECG classification tasks.

4.3.4 Feature Map Reductions

Experiments E13 and E17 examined the impact of feature map reductions.For E13, half reduction of feature maps reduced the model size by 74.6%, but AUROC fell to 0.79 and precision decreased to 0.28. For E17, the feature map reduction to one quarter resulted in a 93.4% decrease in model size, where both AUROC and precision were boosted to 0.88 and 0.45, respectively. Notably, the enhanced performance of E17 over that of E13 is counterintuitive. These results suggest that, in the case of ECG classification, more drastic feature map reduction may result in better generalization, perhaps by reducing overfitting.

4.3.5 Fully Connected Layer Modifications

Experiment E14 reduced the size of the fully connected layer for metadata. This modification had minimal impact on model size (0.2% reduction). The AUROC remained the same (0.86), however, the Precision improved (0.42 vs. 0.39). This suggests that the fully connected layer size has limited impact on model size but can influence precision.

4.3.6 Combined Modifications and Extreme Compression

For the final set of experiments (E15-E18), the research focused on combining the strongest individual modifications to achieve significant model compression. In Experiment E15, we combined the removal of layers (E7) with depth-wise separable convolutions (from E12). This led to a 91.6% reduction in model size, which yielded a 2.84MB model. The AUROC decreased only slightly (0.85 compared to 0.86), however, the precision increased substantially (0.45 compared to 0.39). The above findings highlight that the two modifications complement each other, as their combination maintained the performance gains while additionally accentuating the reduction in model size.

Experiment E17 explored a more aggressive feature map reduction (to 1/4 of the original). This achieved a 93.4% size reduction (to 2.24MB), improving the AUROC (0.88 vs. 0.86) and the precision at the same time (0.45 vs. 0.39). This represents another exceptional result, with both performance metrics improving despite the dramatic size reduction.

In Experiment E18, several strategies of E7, E12, and E17 were combined to realize the highest compression in this work. The strategy was highly effective, resulting in an incredible 99.2% size reduction of the model to as low as 0.28MB. While AUROC declined slightly (0.84 compared to 0.86), this decrease was very small. Accuracy also fared worse compared to other strong models, achieving 0.32, but a figure still comparable to some larger models that were tested. Generally, this test appears to set the functional limit of compression within our tests—a model under 1% the original size, but with reasonable performance.

4.4 Size Reduction Analysis

Figure 4.4 provides a comprehensive view of the size reduction percentages achieved by each modification, along with their corresponding performance metrics.

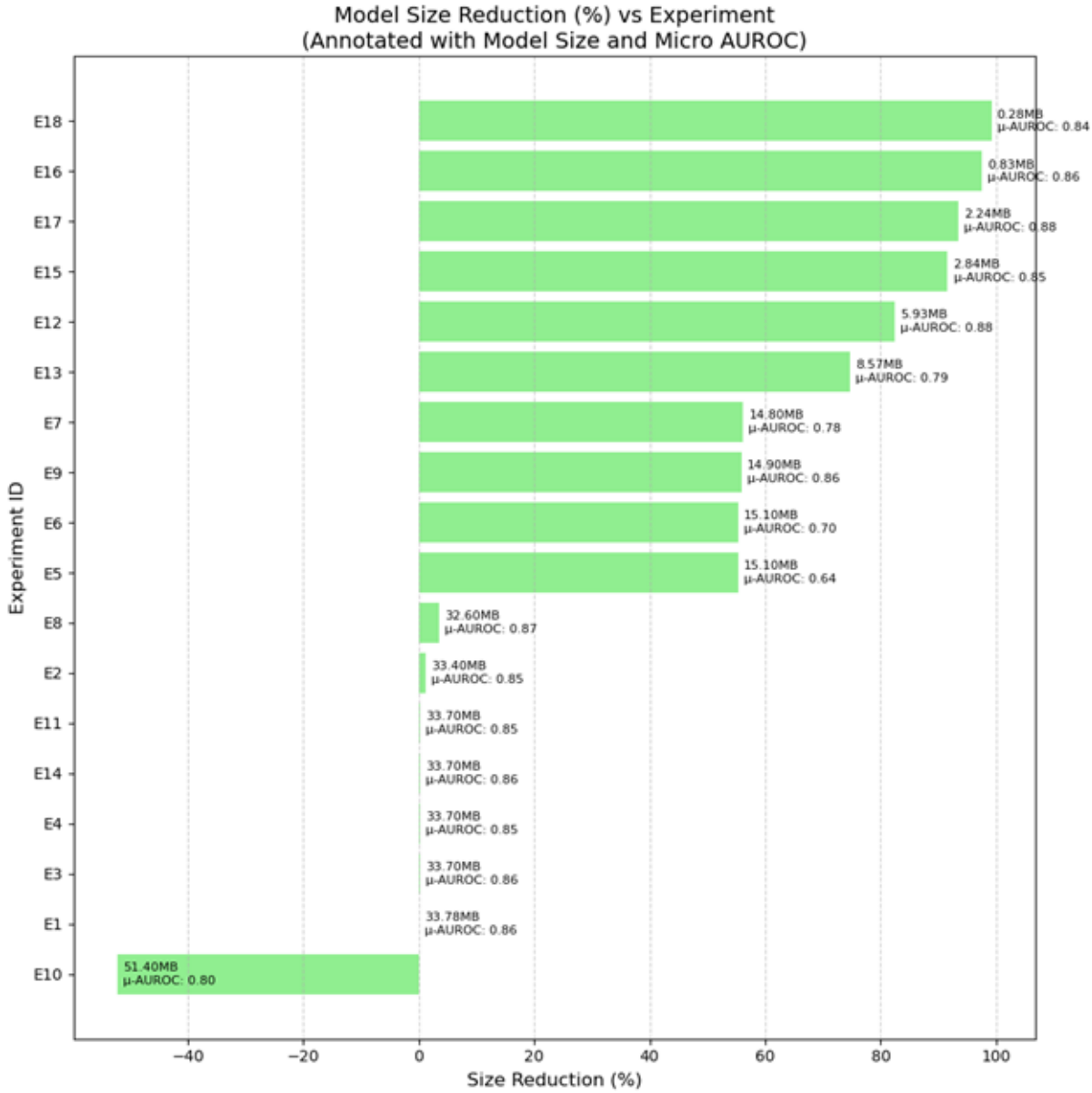

Figure 4.4: Size reduction percentage for each model modification, with annotations showing model size and performance metrics.

From the visualization we can see that there are clear, step-wise improvements in compression, a result of applying various methodological strategies. The performance metrics remain largely stable—even under significant compression pressures. We also notice that certain models, especially E12, E16, and E17, demonstrate an optimal balance between reduced size and maintained performance.

4.5 Comparison of Selected Models

To underscore the most important findings, Figure 4.5 presents a direct comparison between the baseline model and selected efficient models that best exemplify the balance of size reduction and preserved performance. Performance metrics are annotated for clarity.

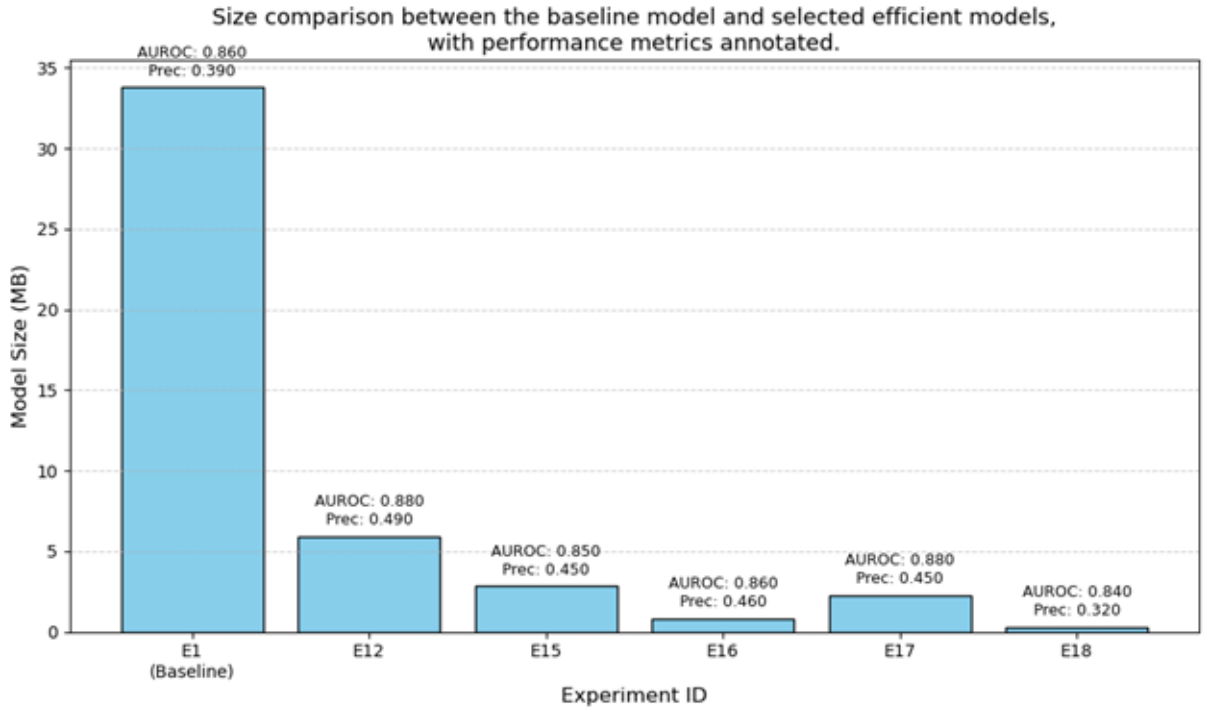

Figure 4.5: Size comparison between the baseline model and selected efficient models, with performance metrics annotated.

This comparison brings attention to the model's dramatic size differences, as model size, up to 99.2%, is visually apparent.It also illustrates that despite such reductions, the models exhibit comparable, or in some cases improved, performance metrics. E17, in particular, distinguishes itself by achieving strong performance (AUROC: 0.88, precision: 0.45) at a notably reduced size of 2.24MB, corresponding to a 93.4% reduction.

The Discussion section below interprets these results to a broader context and formulates the implications, limitations, and future directions of this study.

5 Discussion

5.1 Extreme Model Compression with Maintained Performance

Our experiments indicated that ResNet models for ECG classification could be compressed as much as 99.2% (from 33.78MB to as low as 0.28MB) with minimal loss of performance. Our finding is consistent with recent studies on model compression. For instance, Sahu et al. (2022) presented a Lottery Ticket Hypothesis-based approach (LTH-ECG) which achieved 142x times model size reduction with less than 1% loss in performance for atrial fibrillation detection [36]. Our work extends this research by examining the full range of architectural variations systematically and achieving even more extreme compression ratios.

The capacity to maintain performance despite such extreme size reduction defies the conventional wisdom in deep learning, where it is generally assumed that larger models with larger numbers of parameters would perform better. Our results suggest that standard architectures such as ResNet-18, which in fact were designed for natural image classification, are enormously redundant for the purpose of ECG classification.

5.2 Counterintuitive Performance Improvements in Smaller Models

One of the most intriguing findings of our study is that some of the smaller models (most notably E12 and E17) outperformed the baseline in both AUROC and precision. There are a variety of potential explanations for this outcome, as multiple similar studies [5], [37] have been done by researchers.

5.2.1 Reduced Overfitting

Smaller models contain fewer parameters and consequently a lower ability to memorize noise or irrelevant patterns in training data. This reduced capacity can result in greater generalization of test data, especially for medical signals such as ECGs, where the underlying patterns of interest may be relatively simple in comparison to the model's complexity. Recent work by Chen et al. (2022) supports this interpretation, showing that quantifying uncertainty in smaller models can lead to more reliable atrial fibrillation detection [38].

5.2.2 Architectural Efficiency

Some architectural modifications, such as depth-wise separable convolutions (E12), are less parameter-hungry in their very design to obtain significant patterns within ECG signals. These convolutions factorize the standard convolution operation into a depth-wise convolution followed by a point-wise convolution, dramatically reducing parameters while maintaining the ability to learn useful features. This finding is corroborated by that of Li et al. (2022), where it was found that specialized

convolutional architectures could enhance myocardial infarction detection through the use of single-lead ECGs [39]. Howard et al. similarly revealed that the use of depth-wise separable convolutions has a parameter-efficient method of learning important patterns natively [25].

5.2.3 Improved Signal-to-Noise Ratio

By removing redundant components (such as redundant batch normalization layers or excessive feature maps), the remaining network can focus more on the most discriminative features of the ECG signals. With this greater focus, there can be greater performance even with fewer parameters. Wu and Guo (2025) noted in their systematic review that personalized architectures based on ECG signal features perform better compared to generic deep learning ones [40].

5.2.4 Better Optimization Landscape

Smaller models often have smoother loss landscapes that are easier to optimize during training. This can lead to finding better local minima and thus better performance. The relationship between model size, optimization difficulty, and generalization performance has been explored in the broader deep learning literature and appears to be particularly relevant for ECG classification tasks.

5.3 Most Effective Compression Strategies

Our experiments identified several particularly effective strategies for model compression. Replacing standard convolutions with depthwise separable convolutions

(E12) achieved an 82.4% decrease in size while actually improving performance (AUROC: 0.88 vs. 0.86; precision: 0.49 vs. 0.39). This finding is consistent with the broader trend in successful deep learning models, where depthwise separable convolutions have been successfully used within models like MobileNet for a variety of tasks.

Reducing feature maps by 75% (E17) achieved reduction of size to 93.4% with improved AUROC and precision. This shows that the baseline model contained much more feature maps than needed for effective ECG classification. Improved performance of E17 compared to E13 (feature map reduction of just 50%) is quite prominent and suggests that more drastic feature map reduction at times can lead to better generalization.

Removing layers like the second batch normalization, convolutional, and SE layer (E7) achieved a 56.2% size reduction with an acceptable performance trade-off. Although the size difference was achieved by removing just the second convolutional layer, removing the others with it gave a little better performance. It still goes to show that removing a convolational layer can massively decrease the size of the model. This indicates that not all layers in the standard ResNet architecture contribute equally to ECG classification performance.

The most compression was achieved by combining multiple strategies in tandem. For example, in E18 we removed multiple layers, added depthwise separable convolutions, and reduced the feature maps to achieve a 99.2% size reduction. While this aggressive compression did result in some loss of performance, the model remained feasible with tolerable performance (AUROC: 0.84 vs. 0.86).

5.4 Comparison with Existing Literature

Our findings contribute to the growing body literature regarding successful deep learning for ECG classification. Sahu et al.'s [36] Lottery Ticket Hypothesis (LTH) approach achieved extreme compression for atrial fibrillation detection, yet our findings indicate architectural tuning can achieve even more extreme compression without requiring sophisticated pruning algorithms. Current studies by Han et al. (2020) also demonstrated the vulnerability of deep learning ECG models to adversarial attacks [41], and it was suggested that more basic models would be more resilient as well as being more efficient. Our finding that simple models sometimes achieve better results supports this assertion and suggests model simplification as having benefits in addition to efficiency.

The systematic review of Wu and Guo (2025) [40] examined 198 high-quality studies on the use of deep learning for the analysis of ECG and noted the trend towards specialized architecture taking into account the special characteristics of ECG signals. Our research adds empirical support to the trend and demonstrates how the universal architectures like ResNet-18 can be profoundly transformed in order to better suit ECG classification tasks. Similarly, Petmezas et al. (2022) surveyed state-of-the-art deep learning methods for ECG analysis and noted that, although complex models achieve strong performance, there remains considerable scope for optimization [42]. Our findings quantify this potential, demonstrating that it is possible to remove up to 99.2% of model parameters while still maintaining satisfactory performance.

Our findings are also relevant to the broader literature on effective deep learning. Our tests show that depth-wise separable convolutions perform similarly to

those used in networks such as MobileNet for broad computer vision tasks. Furthermore, the result that effective feature map reduction improves efficiency is consistent with network architecture search and effective model design concepts. The counterintuitive phenomenon of smaller models sometimes beating larger ones is one of the drivers of the controversy over the relationship between model size, complexity, and generalization for deep learning. The phenomenon has been observed in other domains but is particularly noteworthy for medical applications, including ECG classification, where model interpretability and efficiency are crucial.

5.5 Limitations

Although the promising results, our research has several limitations that need to be acknowledged.

First, we were interested in a multi-class classification problem on ECGs. Various ECG tasks, such as beat detection, waveform segmentation, or anomaly detection, may have various architectural requirements. While we anticipate that model compression principles will apply to other tasks as well, optimal trade-offs and configurations may not be directly transferable.

Second, we note that whereas we employed a consistent training procedure across all model variants, we recognize that other methods of training, such as differing optimizers, learning rates, or data augmentation strategies, would influence the mentioned performance. Additional hyperparameter search would be able to discover even better or highly performing model architectures.

Third, our evaluation primarily focused on model accuracy and size. However,

real-world deployment—particularly on implantable or wearable hardware—also needs to take into account hardware-specific metrics such as inference latency, power consumption, and memory. These can impact practical utility and hence need to be directly evaluated by follow-on work in terms of hardware performance.

Lastly, while our models were good on common metric evaluation like AUROC, precision, and recall, clinical deployment would require additional verification. Real-world impact of small variations in metrics needs to be determined in conjunction with medical professionals so model outputs actually influence clinical decision-making in substantial ways, not always the rank order of predicted classes.

5.6 Implications and Future Work

The significant model compression shown in our experiments, which reduces the size by 99.2%, is highly promising for deep learning-based ECG classification model deployment in resource-constrained environments. These lightweight models can be implemented on wearable devices, implantable monitors, and low-power edge nodes, enabling drastic access to automatic ECG interpretation in computational power and memory-constrained environments. This is highly relevant for global health environments or rural populations with little infrastructure.

In addition to accessibility, the performance efficiency of such compressed models also makes them most appropriate for real-time ECG monitoring systems. Continuous low-latency analysis of cardiosignals may potentially allow for early detection of arrhythmia or other cardiovascular disease, thereby enabling faster medical intervention and improved patient care. Real-time capabilities also enhance the

feasibility of closed-loop health monitoring systems that require instant feedback.

Energy efficiency is yet another critical advantage of compact models. Reduced computational requirements translate to reduced energy cost at inference, extended battery life for mobile applications, and reduced overall power footprint. As healthcare applications of AI become increasingly prevalent, particularly in always-on settings, sustainability and environmental impact need to be included in the costs of these models. Compact models assist in making AI more environmentally friendly through greener AI deployment.

Although interpretability was not a primary concern of this research, it must be observed that smaller models are more interpretable and more explainable than large, over-parameterized networks. More work has to investigate whether the leaner architectures that have been discussed here offer actual benefits regarding explainability, especially in high-stakes clinical decision-making. Improved interpretability would enable clinicians to rely on and utilize AI models in high-stake environments such as cardiology.

Lastly, the low weight of our models gives rise to on-device transfer learning and patient-specific adaptation. Since the computational cost is so low, it becomes feasible to perform adaptation directly on users' personal data without sending sensitive health information to a centralized server. Not only does such a distributed learning setup improve performance by adapting to individual patient status but also protects data confidentiality and integrity—two critical concerns in any actual medical application scenario.

6 Conclusion

Our exploration of successful deep learning models for the classification of ECG has demonstrated that extreme model compression (up to 99.2%) is possible while maintaining good performance. In some cases, smaller models even performed better than the baseline.

The most effective compression methods discovered in our experiments are depth-wise separable convolutions, heavy feature map reduction, and selective layer removal. These findings have practical impacts on the deployment of deep learning ECG processing in limited-resource environments, including implantable and wearable devices.

By situating our results within the broader body of work on efficient deep learning and ECG interpretation, we have situated our research and indicated its relevance. Assumptions and limitations notwithstanding, the potential for these efficient architectures to shape healthcare accessibility, real-time monitoring, and environmentally friendly AI deployment is substantial.

Future work would be aimed at verifying these findings on diverse datasets and clinical settings, examining the interpretability implications of the models, and benchmarking these effective architectures on specific hardware platforms and deployment scenarios.

References

[1] A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, *et al.*, "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network", en, *Nat. Med.*, vol. 25, no. 1, pp. 65–69, Jan. 2019.

[2] N. D. Lane, S. Bhattacharya, P. Georgiev, *et al.*, "DeepX: A software accelerator for low-power deep learning inference on mobile devices", in *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, Vienna, Austria: IEEE, Apr. 2016.

[3] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network", in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28, Curran Associates, Inc., 2015. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2015/file/ae0eb3eed39d2bcef4622b2499a05fe6-Paper.pdf`.

[4] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations", 2016. eprint: `1609.07061` (cs.NE).

[5] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network", 2015. eprint: 1503.02531 (stat.ML).

[6] A. G. Howard, M. Zhu, B. Chen, *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications", 2017. eprint: 1704.04861 (cs.CV).

[7] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks", in *Computer Vision – ECCV 2014*, ser. Lecture notes in computer science, Cham: Springer International Publishing, 2014, pp. 818–833.

[8] F. Chollet, "Xception: Deep learning with depthwise separable convolutions", in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, Jul. 2017.

[9] M. Reyna, N. Sadr, A. Gu, *et al.*, *Will two do? varying dimensions in electrocardiography: The PhysioNet/computing in cardiology challenge 2021*, 2022.

[10] H. Liu, D. Chen, D. Chen, *et al.*, "A large-scale multi-label 12-lead electrocardiogram database with standardized diagnostic statements", en, *Sci. Data*, vol. 9, no. 1, p. 272, Jun. 2022.

[11] *University of turku deep ecg classifier repository*, en. [Online]. Available: https://github.com/UTU-Health-Research/dl-ecg-classifier.

[12] Y. Ansari, O. Mourad, K. Qaraqe, and E. Serpedin, "Deep learning for ECG arrhythmia detection and classification: An overview of progress for period 2017-2023", en, *Front. Physiol.*, vol. 14, p. 1 246 746, Sep. 2023.

[13] U. Gupta, N. Paluru, D. Nankani, K. Kulkarni, and N. Awasthi, "A comprehensive review on efficient artificial intelligence models for classification of abnormal cardiac rhythms using electrocardiograms", en, *Heliyon*, vol. 10, no. 5, e26787, Mar. 2024.

[14] L. Alzubaidi, J. Zhang, A. J. Humaidi, *et al.*, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions", en, *J. Big Data*, vol. 8, no. 1, p. 53, Mar. 2021.

[15] M. Kolhar and A. M. Al Rajeh, "Deep learning hybrid model ECG classification using AlexNet and parallel dual branch fusion network model", en, *Sci. Rep.*, vol. 14, no. 1, p. 26 919, Nov. 2024.

[16] Z. Li, H. Li, and L. Meng, "Model compression for deep neural networks: A survey", en, *Computers*, vol. 12, no. 3, p. 60, Mar. 2023.

[17] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks", Oct. 2017. arXiv: 1710.09282 [cs.LG].

[18] P. Antoniadis, *Machine learning: What is ablation study?*, https://www.baeldung.com/cs/ml-ablation-study, Accessed: 2025-5-11.

[19] R. Lini, *Ablation study: What is it, in machine learning?*, en, https://medium.com/@rajilini/ablation-study-what-is-it-in-machine-learning-0a1d362b366d, Accessed: 2025-5-11, May 2024.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks", in *Computer Vision – ECCV 2016*, ser. Lecture notes in computer science, Cham: Springer International Publishing, 2016, pp. 630–645.

[21] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", Feb. 2015. arXiv: 1502.03167 [cs.LG].

[22] A. S. Morcos, D. G. T. Barrett, N. C. Rabinowitz, and M. Botvinick, "On the importance of single directions for generalization", Mar. 2018. arXiv: 1803.06959 [stat.ML].

[23] J. Hu, L. Shen, and G. Sun, "Squeeze-and-Excitation networks", in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, Jun. 2018.

[24] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module", in *Computer Vision – ECCV 2018*, ser. Lecture notes in computer science, Cham: Springer International Publishing, 2018, pp. 3–19.

[25] A. Howard, M. Sandler, B. Chen, *et al.*, "Searching for MobileNetV3", in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South): IEEE, Oct. 2019.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", 2015. eprint: 1512.03385 (cs.CV).

[27] A. Zaeemzadeh, N. Rahnavard, and M. Shah, "Norm-preservation: Why residual networks can become extremely deep?", en, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 3980–3990, Nov. 2021.

[28] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, "On the expressive power of deep neural networks", 2016. eprint: 1606.05336 (stat.ML).

[29] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets", Aug. 2016. arXiv: 1608.08710 [cs.CV].

[30] B. Jacob, S. Kligys, B. Chen, *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference", 2017. eprint: 1712.05877 (cs.LG).

[31] X.-T. Guo, X.-S. Xie, and X. Lang, "Pruning feature maps for efficient convolutional neural networks", en, *Optik (Stuttg.)*, vol. 281, no. 170809, p. 170 809, Jun. 2023.

[32] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once-for-all: Train one network and specialize it for efficient deployment", 2019. eprint: 1908.09791 (cs.LG).

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks", in *Computer Vision – ECCV 2016*, ser. Lecture notes in computer science, Cham: Springer International Publishing, 2016, pp. 630–645.

[34] Z. I. Attia, P. A. Noseworthy, F. Lopez-Jimenez, *et al.*, "An artificial intelligence-enabled ECG algorithm for the identification of patients with atrial fibrillation during sinus rhythm: A retrospective analysis of outcome prediction", en, *Lancet*, vol. 394, no. 10201, pp. 861–867, Sep. 2019.

[35] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size", 2016. eprint: 1602.07360 (cs.CV).

[36] I. Sahu, A. Ukil, S. Khandelwal, and A. Pal, "LTH-ECG: Lottery ticket hypothesis-based deep learning model compression for atrial fibrillation de-

tection from single lead ECG on wearable and implantable devices", in *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, Glasgow, Scotland, United Kingdom: IEEE, Jul. 2022.

[37] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks", *arXiv [cs.LG]*, 2018.

[38] B. Chen, G. Javadi, A. Hamilton, *et al.*, "Quantifying deep neural network uncertainty for atrial fibrillation detection with limited labels", en, *Sci. Rep.*, vol. 12, no. 1, p. 20 140, Nov. 2022.

[39] W. Li, Y. M. Tang, K. M. Yu, and S. To, "SLC-GAN: An automated myocardial infarction detection model based on generative adversarial networks and convolutional neural networks with single-lead electrocardiogram synthesis", en, *Inf. Sci. (Ny)*, vol. 589, pp. 738–750, Apr. 2022.

[40] Z. Wu and C. Guo, "Deep learning and electrocardiography: Systematic review of current techniques in cardiovascular disease diagnosis and management", en, *Biomed. Eng. Online*, vol. 24, no. 1, p. 23, Feb. 2025.

[41] X. Han, Y. Hu, L. Foschini, L. Chinitz, L. Jankelson, and R. Ranganath, "Deep learning models for electrocardiograms are susceptible to adversarial attack", en, *Nat. Med.*, vol. 26, no. 3, pp. 360–363, Mar. 2020.

[42] G. Petmezas, L. Stefanopoulos, V. Kilintzis, *et al.*, "State-of-the-art deep learning methods on electrocardiogram data: Systematic review", en, *JMIR Med. Inform.*, vol. 10, no. 8, e38454, Aug. 2022.

Appendix A Experimental Changes

This appendix details the specific architectural modifications made to the baseline ResNet-18 model for some experiments, as referenced in the main body of the thesis. These changes were implemented to investigate their impact on model size and performance for ECG classification. The code snippets provided illustrate the exact modifications made within the model architecture.

Listing 1 Experiment E11: Reduced Kernel Size

```
self.conv1 = nn.Conv1d(in_channel, 64, kernel_size=7,
stride=2, padding=7, bias=False)
```

Listing 2 Experiment E12: Depthwise Separable Convolutions

```
def conv3x1(in_planes, out_planes, stride=1):
    return nn.Sequential(
        nn.Conv1d(in_planes, in_planes, kernel_size=7, stride=stride,
        padding=3, groups=in_planes, bias=False),  # Depthwise
        nn.Conv1d(in_planes, out_planes, kernel_size=1,
        bias=False)  # Pointwise
    )
```

Listing 3 Experiment E13: Reduced Number of Feature Maps (Half)

```
self.layer1 = self._make_layer(block, 32, layers[0])
self.layer2 = self._make_layer(block, 64, layers[1], stride=2)
self.layer3 = self._make_layer(block, 128, layers[2], stride=2)
self.layer4 = self._make_layer(block, 256, layers[3], stride=2)
self.fc = nn.Linear(256 * block.expansion + 10, out_channel)
```

Listing 4 Experiment E14: Smaller Fully Connected Layer for Metadata

```
self.fc1 = nn.Linear(3, 5)
self.fc = nn.Linear(512 * block.expansion + 5, out_channel)
```

Listing 5 Experiment E17: Reduced Number of Feature Maps (Quarter)

```
self.layer1 = self._make_layer(block, 16, layers[0])
        self.layer2 = self._make_layer(block, 32, layers[1], stride=2)
        self.layer3 = self._make_layer(block, 64, layers[2], stride=2)
        self.layer4 = self._make_layer(block, 128, layers[3], stride=2)

self.fc = nn.Linear(128 * block.expansion + 10, out_channel)
```