Expert Systems With Applications 255 (2024) 124775

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

ECGencode: Compact and computationally efficient deep learning feature encoder for ECG signals[☆]

Lennert Bontinck [*], Karel Fonteyn, Tom Dhaene, Dirk Deschrijver

IDLab, Ghent University - imec, Technologiepark-Zwijnaarde 126, Ghent, 9052, East-Flanders, Belgium

ARTICLE INFO

Keywords:
Electrocardiography
Deep learning
Feature encoder
Computational efficiency
Arrhythmia detection
Predictive modelling

ABSTRACT

The visual interpretation of electrocardiogram (ECG) data is driven by human pattern recognition and requires in-depth medical knowledge. Although state-of-the-art deep learning models can automate and improve ECG feature extraction and analysis, they face deployment challenges, particularly on medical edge devices, due to their extensive computational demands and large parameter counts. To address these limitations, this work introduces ECGencode, a novel deep learning feature encoder optimised for ECG data. ECGencode is characterised by its intuitive, compact, and expert-inspired architecture, drawing from the Filter Bank Common Spatial Patterns method traditionally used in EEG signal analysis. It leverages depthwise and separable convolutions to provide state-of-the-art analysis performance at a fraction of the computational cost. Designed for intuitive model configuration and providing a latent space that retains the structure of an ECG, ECGencode can be incorporated into a wide variety of ECG analysis models. Furthermore, a novel spatial Gaussian noise regularisation technique is introduced, promoting the learning of more generalisable features. ECGencode stands out for its reduced computational requirements, using only 3.79% of the trainable parameters and 12.39% of the FLOPs compared to the benchmark model for normal sinus rhythm atrial fibrillation detection and new-onset prediction. Furthermore, an LSTM-extended ECGencode model matches the performance of leading multi-label classification models with a tenfold reduction in parameters. These attributes position ECGencode as a highly efficient tool for ECG analysis, with the potential to facilitate its adaptation in resource constrained cardiac diagnostics and monitoring settings.

1. Introduction

The electrocardiogram (ECG) is a fundamental diagnostic tool in clinical practice, favoured for its cost-effectiveness, non-invasive nature, and straightforward data acquisition process (Faruk et al., 2021). A standard 12-lead ECG, providing a 10-second recording at high temporal resolution, offers an intricate temporal and spatial portrait of cardiac electrophysiology. This diagnostic modality is integral for the early detection and management of cardiac anomalies, playing a vital role in the timely initiation of therapeutic interventions.

However, the visual interpretation of ECGs is a demanding task, requiring significant medical expertise and is inherently limited by human pattern recognition capacity. These limitations have motivated the development of automated ECG analysis methods, with machine learning algorithms increasingly becoming the method of choice (Gilon et al., 2023; Mincholé et al., 2019; Petmezas et al., 2022; Sau & Ng, 2023; Somani et al., 2021). Owing to their data-driven approach, machine learning algorithms, and deep learning (DL) models in particular, can learn complex features from raw ECG data. This has allowed them to surpass traditional methods in tasks such as arrhythmia classification and rhythm analysis (Mincholé et al., 2019; Petmezas et al., 2022; Sau & Ng, 2023; Somani et al., 2021).

A key benefit of DL models is the fact that they can be run fully autonomously on both stored and live ECG recordings. This makes them ideal for integration in clinical decision support systems, where they can help clinicians in their complex decision-making processes (Mincholé et al., 2019; Petmezas et al., 2022; Sau & Ng, 2023). These models can perform multi-label classification on a variety of diagnostic ECG statements, including rare arrhythmia, offering potentially valuable second opinions for clinical diagnostics. Furthermore, DL models have shown promise in enhancing patient screening and risk stratification, challenging conventional risk scores like the CHARGE-AF score for atrial fibrillation (AFib), which primarily rely on tabular data from the Electronic Health Record (EHR, Alonso et al., 2013). This follows from

[☆] Funding: This work was supported by the Flemish Government via the AI Research Program.

[*] Corresponding author.

E-mail addresses: lennert.bontinck@ugent.be (L. Bontinck), karel.fonteyn@ugent.be (K. Fonteyn), tom.dhaene@ugent.be (T. Dhaene), dirk.deschrijver@ugent.be (D. Deschrijver).

https://doi.org/10.1016/j.eswa.2024.124775
Received 13 December 2023; Received in revised form 19 June 2024; Accepted 8 July 2024
Available online 14 July 2024

the initial assumption that no indicative patterns of AFib are observable on the ECG before AFib onset or during normal sinus rhythm (NSR). However, recent studies reveal that DL can predict new-onset AFib or identify AFib from NSR ECGs with greater accuracy than traditional methods, suggesting their utility as risk score and in selecting patient subgroups for extended screening (Attia, Noseworthy, et al., 2019; Christopoulos et al., 2020; Gruwez et al., 2023; Raghunath et al., 2021; Sau & Ng, 2023). Moreover, DL has been utilised to detect markers of systemic diseases such as COVID-19 in ECG data, showcasing its ability to extend beyond traditional cardiac diagnostics (Sakr et al., 2023).

Nevertheless, widespread clinical application of advanced DL models for ECG analysis is still limited, not least due to their "black-box" nature, which complicates their direct use as standalone diagnostic tools (Mincholé et al., 2019; Petmezas et al., 2022; Sau & Ng, 2023; Somani et al., 2021). While acknowledging this restriction, DL models, when used appropriately, can still significantly enhance decision support systems by augmenting clinical judgement with previously unavailable risk scores and insights into potential, overlooked diagnoses (Attia, Noseworthy, et al., 2019; Christopoulos et al., 2020; Gruwez et al., 2023; Raghunath et al., 2021; Sau & Ng, 2023; Strodthoff et al., 2021). However, other deployment challenges persist, in part due to the reliance on complex, parameter-heavy models like residual networks (ResNets) which are not inherently optimised for ECG signal analysis.

Firstly, the number of computational operations these complex state-of-the-art (SOTA) models need to perform during inference, expressed in floating-point operations (FLOPs), is enormous. This makes them significantly resource-intensive, rendering them impractical for use in resource-constrained environments, such as inference on low-powered medical edge devices without GPUs (Phukan et al., 2023). Secondly, the high parameter counts of these models combined with the limited and imbalanced availability of (public) ECG datasets increases the risk for various unfavourable training behaviours. This includes the risk of overfitting and bias learning, requirement of a significant amount of costly VRAM for GPU training, slow gradient calculation due to the many trainable parameters and thus a slower training process, and others (Buber & Diri, 2018; Gyawali, 2023; Liao et al., 2022; Mincholé et al., 2019; Phukan et al., 2023; Somani et al., 2021). Thirdly, as these models are adopted from other fields and not specifically tailored to ECG analysis, there is no intuitive meaning as to what the model configuration parameters mean in relation to the ECG analysis. This results in the choice of a default, non-optimised model configuration or the need for computationally very expensive and time-consuming configuration tuning, a process which can increase the risk of overfitting, effectively decreasing generalisation performance (Liao et al., 2022). Fourthly, the latent space representation of the ECG signal using these models has no intuitive meaning and is often large in size, can capture unnecessary redundancies and makes modifications or extensions to these architectures a challenging task (Mincholé et al., 2019; Somani et al., 2021). Finally, given the nature of deep SOTA models used, interpretability of the learned weights and resulting predictions is hard and often limited to general post hoc methods such as gradient-based class activation maps (Chattopadhay et al., 2018; Jiang et al., 2021; Selvaraju et al., 2020; Wang et al., 2020) and saliency maps (Simonyan et al., 2014; Smilkov et al., 2017).

In response to these challenges, this paper introduces ECGencode: a compact and computationally efficient DL feature encoder made specifically for ECG signals. ECGencode serves as a building block for the creation of various model architectures to perform ECG-specific tasks. It can be used to transform a high dimensional raw input ECG to a smaller-sized latent space with learned features relevant to the task, whilst offering the following benefits:

ECG Specific, Compact, and Expert-Inspired Architecture: ECGencode efficiently transforms high-dimensional raw ECG data into a compact, information-rich latent space, preserving the structure of an ECG. The novel DL model architecture, inspired by the expert-approved Filter Bank Common Spatial Patterns (FBCSP, Ang et al., 2008) technique, introduces a novel Spatial Gaussian Noise layer for regularisation across both lead and channel dimensions.

Computationally Efficient and Low-Parameter Design: Characterised by its low parameter count and the utilisation of depthwise and depthwise separable convolutions, ECGencode delivers SOTA-level performance with significantly reduced computational demands. A binary classification model incorporating ECGencode, designed for NSR AFib detection and new-onset AFib prediction, achieves comparable results to SOTA models while requiring over twenty times fewer parameters and reducing FLOPs by tenfold. Such efficiency facilitates deployment on resource-constrained edge devices, without reducing the classification performance.

Intuitive Model Configuration and Interpretable Architecture: ECGencode's DL architecture supports intuitive model configuration tuning, closely aligned with ECG signal characteristics, and produces an interpretable latent space mirroring the ECG leads over time structure. Each layer is configured for a distinct, visually interpretable task, from temporal frequency filtering to spatial reduction into augmented leads, enriching model transparency and understanding.

Versatile By Design: Deliberately proposed as a feature encoder rather than a complete model, ECGencode is highly adaptable, supporting diverse configurations and extensions tailored to specific ECG analysis tasks. Its flexible and easy-to-adopt nature makes ECGencode an ideal building block for developing clinically viable DL models, closing the gap between the performance of computationally expensive models and the resource constraints found in medical edge devices.

Many of these points relate to the computational efficiency of a model, a key benefit of ECGencode, which encompasses several aspects. Computational efficiency is primarily measured by the FLOPs count, indicating CPU cost when GPUs are unavailable, as is often the case for medical edge devices. Additionally, the parameter count denotes the number of weights that need to be stored, impacting memory cost. A model with fewer trainable parameters is also less susceptible to overfitting, as it requires fewer weights to be learned, potentially leading to better performance with less data and shorter training sessions, both of which enhance computational efficiency in terms of training cost. ECGencode's intuitive parameterisation avoids the need for computationally expensive grid searches to determine optimal hyperparameters, further improving training efficiency. Finally, extensions using ECGencode benefit from the computationally efficient reduction of a high-dimensional raw ECG input to a compact latent space, ensuring lower computational cost in subsequent processing stages.

The structure of the paper is as follows. Section 2 provides an overview of the related work. First, a brief history of ECG signals and the transition towards automated ECG analysis is provided. Next, the most common DL architectures used for ECG analysis and available model interpretability techniques are discussed. Afterwards, the issue of computational efficiency for these complex models is explained in more detail and existing literature on improved computational efficiency is highlighted. Finally, based on these topics and the identified gaps, ECGencode is positioned as a tool to intuitively build computationally efficient DL ECG analysis models. In Section 3, the architecture of ECGencode is detailed, highlighting its computational efficiency and the novel ECG-specific normalisation layers that enhance model generalisability. Furthermore, the analogy with the traditional FBCSP framework is explored, providing a comprehensive understanding of the model's layers and learned weights. This visual interpretation of the architecture is further elaborated as well. Section 4 presents a thorough evaluation of ECGencode through two models incorporating ECGencode. Three binary ECG classification tasks are evaluated using ECGencode model 1: detection of AFib-related patients, detection of AFib during normal sinus rhythm (NSR) and prediction of AFib before its first onset. Additionally, a fourth task employs ECGencode model

2, extended with LSTM capabilities, for multi-label ECG classification. For these evaluations, the PTB-XL (Goldberger et al., 2000; Strodthoff et al., 2021; Wagner et al., 2020, 2022) and CODE-15% (Lima et al., 2021; Ribeiro et al., 2021, 2020) open-source data sets are utilised. The results underscore ECGencode's computational efficiency, its adaptability for various tasks, and its ability to match the performance of SOTA models with orders of magnitude fewer parameters and FLOPs. Section 5 reflects on the wider impact of ECGencode for automated ECG analysis. It discusses the role of ECGencode as a highly suitable DL feature encoder for diverse ECG analysis applications, motivated by the obtained results. The potential for its integration in ECG analysis is explored, primarily due to its computational efficiency and minimal resource demands. Finally, Section 6 summarises the main conclusions of this research and discusses interesting future work.

2. Related work

Automated ECG analysis has evolved significantly since its inception in the early 1960s, leading to widespread adoption in both clinical and consumer-grade devices (Macfarlane & Kennedy, 2021; Petmezas et al., 2022). Modern medical ECG acquisition devices and even smart wearables now commonly feature automated analysis, employing a variety of rule-based, expert-derived algorithms to diagnose heart conditions (Faruk et al., 2021; Macfarlane & Kennedy, 2021; Musa et al., 2023).

The recent growth of the Internet of Medical Things has led to a rapidly growing availability of medical data, including ECG recordings linked with patients' EHRs and other metadata (Musa et al., 2023). This increase in available data has caused the rapid development of new automated ECG analysis methods, with deep learning (DL) becoming the preferred approach, outperforming traditional methods in various aspects (Jaworski et al., 2022; Mincholé et al., 2019; Musa et al., 2023; Petmezas et al., 2022; Sau & Ng, 2023; Somani et al., 2021). DL's main advantage is its ability to automatically extract features, eliminating the need for predetermined expert diagnosis rules and manual feature selection (Macfarlane & Kennedy, 2021; Sau & Ng, 2023). This capability of automatic feature learning helps to uncover medical conditions or their precursors not visible through conventional analysis, expanding diagnostic capabilities and potentially enabling early disease detection and screening (Attia, Noseworthy, et al., 2019; Sau & Ng, 2023).

Whilst the field is rapidly evolving, some general DL issues as well as specific medical adaptation issues are still present. This section presents some of the most relevant issues and how they are currently handled in literature.

2.1. Interpretability of deep learning in ECG analysis

DL's capability to automatically learn features from raw ECG data is both its biggest strength and its biggest weakness, as it introduces the black box problem (Ayano et al., 2023; Hicks et al., 2021; Musa et al., 2023; Petmezas et al., 2022). The black box problem refers to the decision-making process of these DL models, which with their hundreds of thousands of parameters, remains opaque, lacking a clear interpretation or explanation. In healthcare, where such explanations for diagnoses are crucial, this opacity raises concerns and limits widespread adoption, especially for the use of these models as a stand-alone diagnostic tool (Hicks et al., 2021).

In response, various post hoc methods to provide some form of interpretability to a trained DL model have been proposed. Among the most popular in ECG research is post hoc visual explanations, including saliency maps (e.g., vanilla saliency, SmoothGrad) and gradient-based class activation maps (e.g., ScoreCAM, LayerCAM, GradCAM, GradCAM++), where a heatmap highlights areas in the input ECG that influence the prediction most (Chattopadhay et al., 2018; Jiang et al., 2021; Selvaraju et al., 2020; Simonyan et al., 2014; Smilkov et al., 2017; Wang et al., 2020). This allows clinicians to focus their review on a specific portion of the ECG and potentially uncover new diagnostic markers. Some of these general DL techniques have seen adaptation specifically for ECG analysis models, with ECGradCAM by Hicks et al. (2021) being a popular example. For more information on interpretability techniques in ECG analysis, see the review article by Ayano et al. (2023).

It is important to note that even without specifically addressing the black box issue, deep learning models can still offer valuable support in clinical decision-making. They can predict the likelihood of conditions such as arrhythmias or future cardiac events, guiding closer monitoring for patients at risk in a setting where no traditional risk scores are fit for determining a group of patients that should be screened (Christopoulos et al., 2020; Raghunath et al., 2021; Sau & Ng, 2023). Furthermore, they can simply bring potentially missed arrhythmia to a clinician's attention, serving as a valuable second opinion or helping prioritise patient data for review, without providing a definitive and final diagnosis (Ayano et al., 2023).

2.2. Common model architectures for ECG analysis

Convolutional Neural Networks (CNNs) are the most commonly used type of architecture for DL ECG analysis. For the majority of works, simple CNN architectures with alternating convolution and pooling layers are used, although the more sophisticated ResNets which were designed to tackle the vanishing gradient problem in deep convolutional networks are becoming more popular (Jaworski et al., 2022; Petmezas et al., 2022; Sau & Ng, 2023; Somani et al., 2021).

Given the sequential nature of ECG data, Recurrent Neural Networks (RNNs) have been proposed for complex ECG analysis tasks (Bozyigit et al., 2020; Petmezas et al., 2022). In particular, Long Short-Term Memory (LSTM) units and Gated Recurrent Units (GRUs) are popular RNN options for ECG analysis. However, their complexity often leads to overfitting, challenging their superiority over ResNets despite theoretical benefits for time-series data (Bozyigit et al., 2020; Petmezas et al., 2022). CNN-LSTM combinations, which first compress ECG data into a manageable latent space via a CNN architecture before applying RNN analysis via LSTM units have proven beneficial in some applications (Abdullah & Al-ani, 2020; Alamatsaz et al., 2024).

These model architectures are adopted from other fields, predominately the field of image processing, meaning they are not explicitly tailored for ECG analysis which introduces several issues. The lack of intuitive correlation between model configuration parameters and ECG signal complexity, such as the amount of residual blocks in a ResNet or the kernel size of a CNN, often necessitates empirical, computationally intensive model tuning. This complicates the optimisation process, which often leads to the use of non-optimised default parameters or an optimisation process which increases the risk for overfitting or bias learning (Jaworski et al., 2022; Liao et al., 2022). For medical professionals, who may be less familiar with deep learning intricacies but have proven significant contributions to the field, this issue is even more pronounced. Petmezas et al. (2022) provide a more detailed overview of commonly used DL architectures for ECG analysis.

2.3. Computational efficiency optimisation

Besides the challenges with model architectures not being optimised for ECG analysis, these models also tend to be computationally demanding. They often have hundreds of thousands of parameters and require significant amounts of FLOPs, impacting both the training and inference phases. During training, especially with imbalanced datasets common in ECG analysis, a large number of parameters increases the risk of overfitting. It also makes the training process longer and requires more storage and VRAM during GPU training, which could cause an economic barrier to entry for smaller research groups or clinical settings (Sharir et al., 2020; Xu & Du, 2023). For inference, these models require more resources and lead to longer processing times and increased battery usage, limiting their use in resource-constrained environments like medical edge devices (Phukan et al., 2023).

Current approaches to enhance computational efficiency in deep learning models often employ post-training operations, focusing on reducing FLOPs and parameter count for the final inference model, but starting from a trained model of considerable complexity with all risks and requirements associated. Knowledge distillation, or teacher–student modelling, is one such strategy, having been applied to ECG analysis to reduce a 12-lead model to a more computationally manageable single-lead model (Qin et al., 2023; Sepahvand & Abdali-Mohammadi, 2022). Similarly, multistage pruning has effectively reduced the complexity of trained ECG models (Xiaolin et al., 2021). However, these methods inherit the limitations of the initial complex models they are based on. This includes the potential of mimicking unwanted behaviour which results from overfitting whilst requiring many computational resources and enough data for training the initial complex model as well as a computationally expensive hyperparameter grid search to find optimal model configuration for this initial model. Other approaches start directly from a single-lead or even a single heartbeat as input to learn a more compact model, but the found computational efficiency gains largely result from alterations to the input data rather than intrinsic architectural innovations (Alfaras et al., 2019; Dubatovka & Buhmann, 2022; Khan et al., 2023).

Architectural design for improved computational efficiency has been explored in the general field of deep learning and has seen some interest in the field of ECG analysis. Densely connected convolutional networks (DenseNets), for example, require fewer parameters compared to traditional ResNet-based models, mitigating overfitting risks (Huang et al., 2017). In the context of AFib detection, a DenseNet-based model has demonstrated comparable performance to SOTA models with only 69,087 parameters (Cai et al., 2020). While the parameter reduction is notable, DenseNet-based models typically still entail high computational costs in terms of FLOPs due to their deep layered structure. In contrast, the binary ECG classification model for AFib detection during NSR and new-onset AFib prediction presented in this paper (ECGencode model 1) achieves similar performance to SOTA with merely 8242 parameters and further reduces computational demands by minimising layer count and using specialised convolutions for a lower FLOPs count. Recently, a study by Phukan et al. (2023) explored the use of simpler CNN architectures to reduce computational demands for deployment on edge devices. However, as discussed in their work and revealed in the evaluation of Section 4.3, while these architectures might lower FLOPs, they still maintain a substantial number of parameters and fail to match the performance of SOTA models in various tasks.

The quest for computational efficiency in ECG analysis has also led to the development of custom chips for running traditional convolutional-based neural networks (Gu et al., 2023). While these offer impressive energy and time efficiency during inference, their requirement for specialised hardware, which only supports specific types of operations, limits their general applicability.

2.4. Positioning of ECGencode in literature

The commonly used DL models for ECG analysis, dominated by traditional CNNs and ResNets, present several challenges. First, as these models were not originally designed for the high dimensional ECG signal analysis nor deployment on resource-constrained environments such as medical edge devices, they require high computational resources. This includes high FLOPs counts which necessitate more CPU power for inference and high parameter counts which require more memory to store the trained model and increase the risk of overfitting. Existing solutions for improving this computational efficiency and allowing them to be run on medical edge devices either compromise performance for efficiency, necessitate specialised hardware, start from a complex model with its associated drawbacks, or fail to scale across different ECG analysis tasks (Cai et al., 2020; Gu et al., 2023; Phukan et al., 2023). Second, while models like ResNets offer some control over complexity, such as adjusting the number of residual blocks, these configurations often lack a direct relationship to the ECG signals or the specific task at hand. This results in a default, overly complex configuration, or the requirement of time-consuming, empirical, and computationally very expensive model optimisation through a hyperparameter grid search, which increases the risk of overfitting and bias learning (Liao et al., 2022). Third, the latent space representations of the input ECG from these models are large and unstructured with no straightforward way of being extended. This latent space representation combined with the deep structure of the commonly used models and high parameter counts also makes intrinsic interpretation of the learned parameters hard, if not impossible.

To bridge these gaps, ECGencode is introduced as a versatile and computationally efficient DL feature encoder, serving as a building block for a wide possibility of DL ECG analysis models. Based on the expert-inspired FBCSP approach, it transforms complex ECG inputs into a manageable latent space that retains the ECG's structure, suitable for use and extension in a variety of DL ECG analysis models. Its compact and computationally very efficient architecture, employing depthwise and depthwise separable convolutions, allows for the creation of models with minimal parameters and FLOPs which are applicable for deployment on edge devices without compromising performance. The introduction of a novel, ECG-specific, Spatial Gaussian Noise regularisation technique provides satisfactory generalisation without impact on inference speed. Furthermore, ECGencode supports intuitive model configuration and offers interpretability at both the architectural and parameter levels. Not only does this facilitate researchers to configure a custom DL model, but it also allows for model-specific visualisations and some intrinsic parameter interpretation besides existing post hoc visualisation techniques.

It is noted that unlike pre-trained, general ECG feature encoders, such as those using self-supervised learning or auto-encoders, ECGencode is crafted as a trainable component for supervised learning models. This ensures the extraction of task-relevant features, shown to be generalisable to data sets from other clinics, enabling performance that matches or surpasses SOTA models while maintaining efficiency and interpretability (Christ et al., 2018; Del Pup & Atzori, 2023; Gedon et al., 2021; Jang et al., 2021; Kuznetsov et al., 2021; Liu et al., 2021).

3. ECGencode feature encoder

The challenges outlined in the previous section highlight the need for a deep learning feature encoder that prioritises computational efficiency without sacrificing representation capability, while also remaining versatile enough through intuitive model configuration parameterisation for use in various tasks based on a raw ECG input. To meet these requirements, this section introduces ECGencode, a compact deep learning feature encoder designed for standard 12-lead ECG signals. Despite its minimal use of learning parameters and FLOPs, ECGencode retains crucial information in its latent space, enabling it to perform competitively with far more complex models across different problem settings, as evidenced by the evaluation performed in Section 4.

This section offers a comprehensive overview of the ECGencode architecture. Section 3.1 discusses the model's temporal, spatial, and feature convolutions. To enhance feature generalisability, ECGencode incorporates ECG-specific normalisation and an ECG-specific novel Spatial Gaussian Noise regularisation technique, which are detailed in Section 3.2. Section 3.3 presents an analysis of the model's computational efficiency, revealing a significant reduction in FLOPs and trainable parameters. The architecture provides options for increasing model complexity and for including extensions, such as a CNN-LSTM extension, through intuitive model configuration parameterisation and a latent space that retains the ECG structure as discussed in Section 3.4. Lastly, Section 3.5 explores various ECGencode-specific visualisation techniques, both intrinsic and post hoc, made possible by the architecture's novel design.

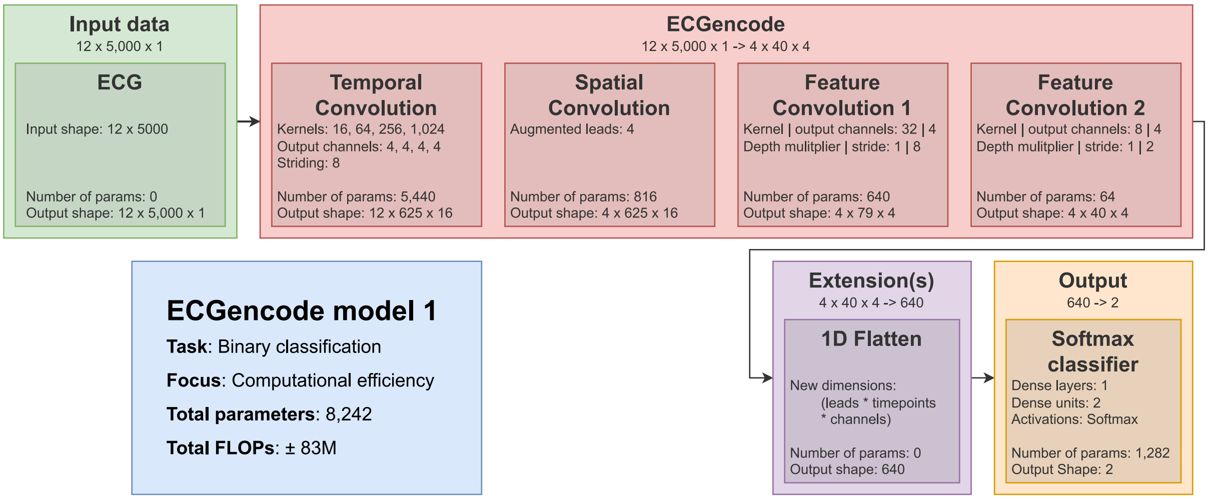


Fig. 1. A high-level overview of ECGencode model 1 used for binary ECG classification. This model, and the ECGencode configuration it uses, focus on achieving the highest computational efficiency possible without significant classification performance loss. The 2D input consists of a standard 10-second 12-lead ECG sampled at 500 Hz, represented as a 12×5000 matrix. ECGencode outputs a compact 3D latent space with dimensions 4 × 40 × 4. Binary ECG classification is achieved using a fully connected layer with softmax activation, applied to the 1D flattened latent space. This model's total parameter count is 8242 and FLOPs count is estimated to be ±83M.

3.1. Compact convolutional architecture for automated ECG feature encoding

Fig. 1 presents a high-level overview of the ECGencode architecture as configured in a model for binary ECG classification (ECGencode model 1). The input consists of a standard 10-second 12-lead ECG sampled at 500 Hz, represented as a 12 × 5000 2D matrix. ECGencode will convert this input to a 3D matrix by adding an additional channel dimension, resulting in a 12 × 5000 × 1 3D matrix that can be interpreted as leads × time points × channels. This allows ECGencode to explicitly retain the 2D structure of the ECG signal throughout its different layers, resulting in a final latent space of shape 4 × 40 × 4 which can be interpreted as a signal of 4 augmented leads, 40 time points and 4 channels. The final layer of the first ECGencode model links this latent space generated by ECGencode to the binary ECG classification output through a fully connected layer with softmax activation. In this configuration, the latent space is simply flattened into a one-dimensional vector before softmax activation, resulting in a compact model with a total of only 8242 parameters.

ECGencode comprises four sequential components: a temporal convolution, a spatial convolution, and two feature convolutions. These components are inspired by the FBCSP method of Ang et al. (2008), a well-established approach in feature engineering for EEG signals. In FBCSP, temporal filters partition the signal into multiple frequency banks, followed by spatial filters using the Common Spatial Patterns (CSP) algorithm by Koles et al. (1990) to maximise inter-class variance for each frequency bank. Subsequent feature selection methods like mutual information reduce dimensionality and redundancy, paving the way for classifiers such as linear discriminant analysis (LDA, Izenman, 2008).

Previous efforts to adapt the FBCSP method for deep learning have primarily focused on EEG analysis in brain–computer interface applications, such as the EEGNet model by Lawhern et al. (2018) commonly used as benchmark. In contrast, ECGencode is explicitly optimised for standard 12-lead ECG signals rather than EEG signals and high computational efficiency, offering a more compact and efficient feature encoder compared to EEGNet and its variants (Huang et al., 2020; Lawhern et al., 2018; Riyad et al., 2020; Roots et al., 2020; Wang, 2023; Zhang et al., 2022).

3.1.1. Temporal convolution

The temporal convolution component, as shown in Fig. 2, is designed to capture a range of temporal dependencies in the ECG signal. Inspired by the filter banking stage of FBCSP, this convolution features kernels of varying lengths that process each lead independently to produce an output comparable to that of a frequency filter. These kernels, with dimensions 1 × 16 × 1, 1 × 64 × 1, 1 × 256 × 1, and 1 × 1024 × 1, capture information at multiple temporal scales: 0.03, 0.1, 0.5, and 2 seconds, respectively. Employing kernels of diverse temporal lengths allows ECGencode to capture both high-frequency and low-frequency information. Kernels with a shorter temporal axis focus on high-frequency details, whereas those with a longer temporal axis help in smoothing the signal and capturing low-frequency traits. Whilst the longer temporal kernels can provide valuable feature extraction, their use should be considered keeping the desired inference device in mind, as they can easily blow up the FLOPs count of the model. For example, the four 1 × 1024 × 1 kernel in ECGencode model 1 are responsible for more than 60M out of the total ±83M FLOPs. Due to the high input sampling rate of 500 Hz, a stride of 8 is applied to downscale the temporal axis.

Each of these kernel convolutions creates four output channels, which are all merged along the channel axis, resulting in an ECG-like signal of size 12 × 625 × 16 with temporal alterations based on frequency filtering along the channel axis and a considerably down-scaled temporal axis. Given that the input ECG only has one input channel, using 2D depthwise or 2D depthwise separable convolutions, as explained in Section 3.3, would not yield any computational or parameter efficiency gains, explaining the use of standard 2D convolutions in the temporal convolution component.

3.1.2. Spatial convolution

The spatial convolution component, represented in Fig. 3, takes as its input the output from the preceding temporal convolution component. It aims to capture spatial correlations across all twelve ECG leads, similar to the CSP stage in FBCSP (Ang et al., 2008; Koles et al., 1990). Four independent depthwise 2D convolutions are utilised, each with a kernel of size 12 × 1 × 1. This results in four augmented leads, each synthesised from all twelve original leads. The selection of four augmented leads serves dual purposes: it both reduces spatial

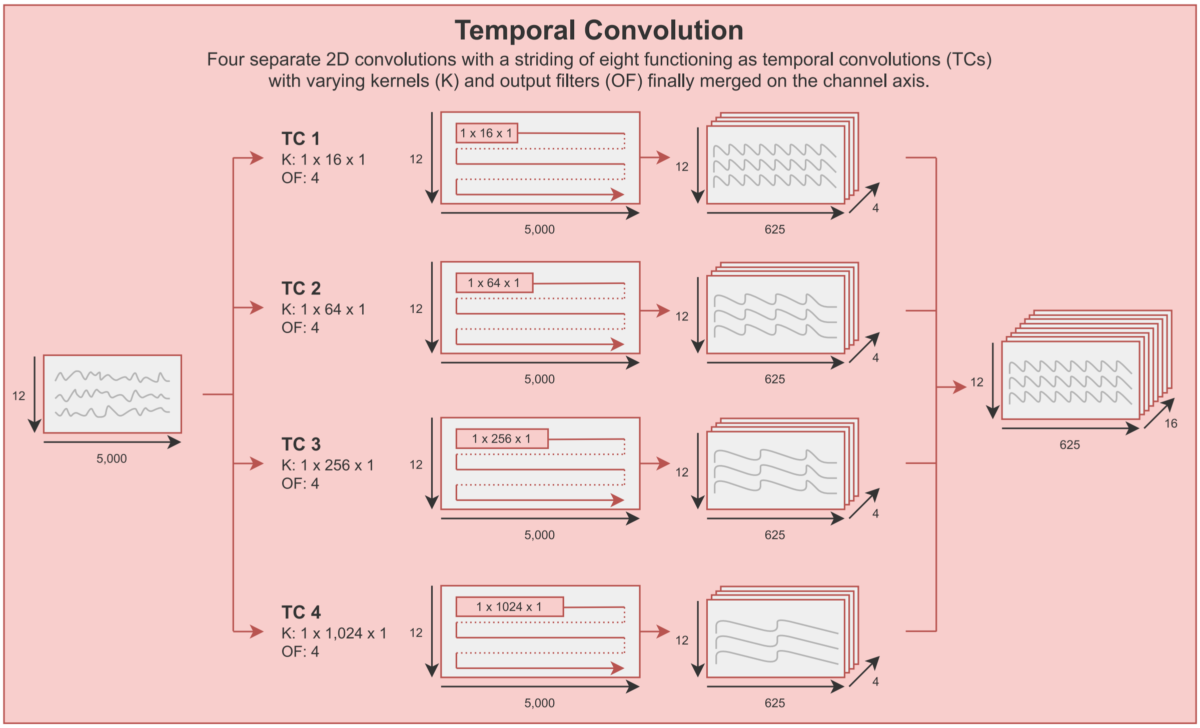

Temporal Convolution
Four separate 2D convolutions with a striding of eight functioning as temporal convolutions (TCs) with varying kernels (K) and output filters (OF) finally merged on the channel axis.
12
5,000
TC 1
K: 1 x 16 x 1
OF: 4
1 x 16 x 1
TC 2
K: 1 x 64 x 1
OF: 4
1 x 64 x 1
TC 3
K: 1 x 256 x 1
OF: 4
1 x 256 x 1
TC 4
K: 1 x 1,024 x 1
OF: 4
1 x 1024 x 1
625
4
16

Fig. 2. In-depth view of the temporal convolution component within ECGencode, configured per ECGencode model 1 specified in Fig. 1. Four distinct standard 2D convolutions are employed, each with a stride of eight. Kernel sizes vary along the temporal axis: 1 × 16 × 1, 1 × 64 × 1, 1 × 256 × 1, and 1 × 1024 × 1. The outputs of these convolutions are combined along the channel axis to form the final output.

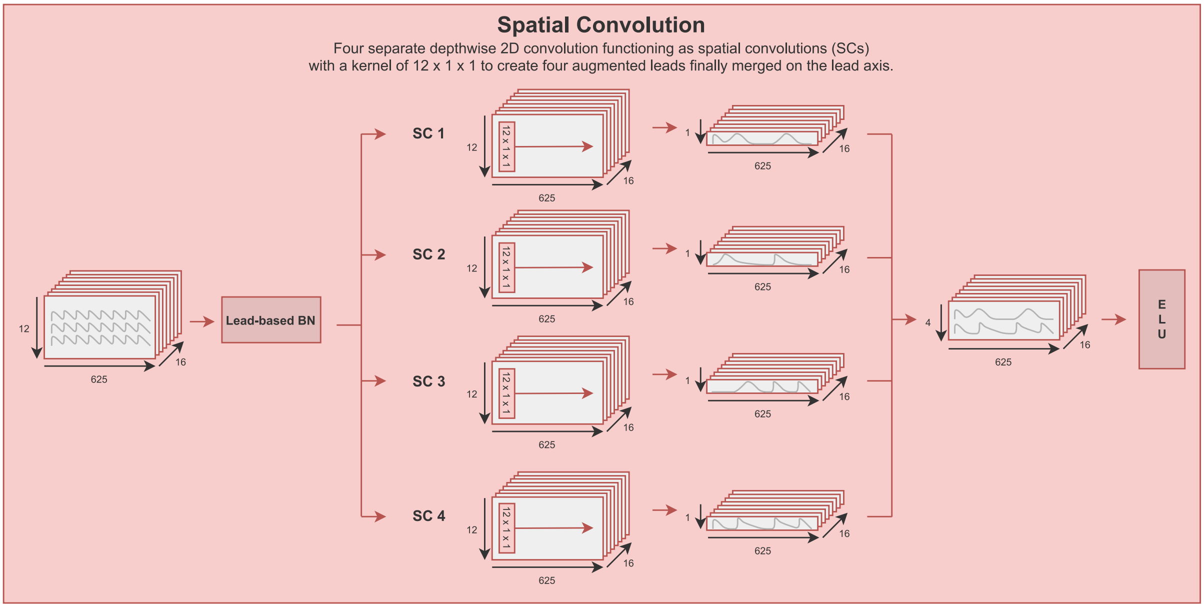

Spatial Convolution
Four separate depthwise 2D convolution functioning as spatial convolutions (SCs) with a kernel of 12 x 1 x 1 to create four augmented leads finally merged on the lead axis.
12
625
16
Lead-based BN
SC 1
SC 2
SC 3
SC 4
12 x 1 x 1
1
4
E L U

Fig. 3. In-depth view of the spatial convolution component within ECGencode, configured per ECGencode model 1 specified in Fig. 1. The input for this stage is the output from the temporal convolution shown in Fig. 2. Four separate depthwise 2D convolutions are employed, each with a kernel size of 12 × 1 × 1, covering all leads. The outputs, termed augmented leads based on the original twelve leads, are combined along the lead axis to produce the final output.

complexity and aligns with the optimal number of leads for neural network training found by Lai et al. (2021).

Unlike standard 2D convolutions, which would employ a kernel of 12 × 1 × 16, each depthwise 2D convolution applies 16 distinct 12 × 1 × 1 kernels to each input channel. Without the use of a depth multiplier, this yields 16 unique output channels for each convolution. Combining these outputs along the lead axis produces a final output of size 4 × 625 × 16. This output maintains the frequency-based temporal variations from the previous stage while reducing the original 12 leads to four augmented ones. Besides allowing explicit retention of the input channels, depthwise convolutions also offer computational efficiency benefits as further explained in Section 3.3.

Before the spatial convolution, lead-based batch normalisation (BN, Ioffe & Szegedy, 2015) is applied to the output of the temporal convolution. This normalisation facilitates the training of ECGencode and enhances the intrinsic interpretability of the spatial convolution's learned

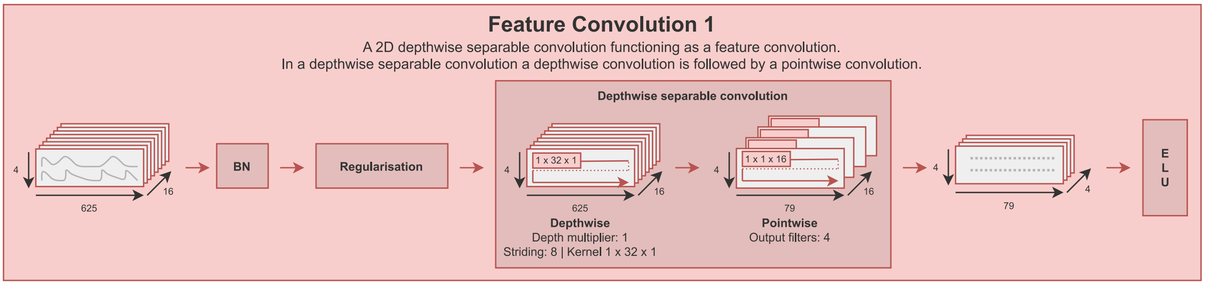

Feature Convolution 1
A 2D depthwise separable convolution functioning as a feature convolution.
In a depthwise separable convolution a depthwise convolution is followed by a pointwise convolution.
Depthwise separable convolution
4
625
16
BN
Regularisation
1 x 32 x 1
625
Depthwise
Depth multiplier: 1
Striding: 8 | Kernel 1 x 32 x 1
1 x 1 x 16
79
Pointwise
Output filters: 4
79
E L U

Fig. 4. In-depth view of the first feature convolution within ECGencode, configured per ECGencode model 1 specified in Fig. 1. The input originates from the spatial convolution shown in Fig. 3. This stage employs a depthwise separable convolution, with the depthwise convolution using a kernel of 1 × 32 × 1 and a stride of eight. The pointwise convolution has a kernel of 1 × 1 × 16 and produces four output filters.

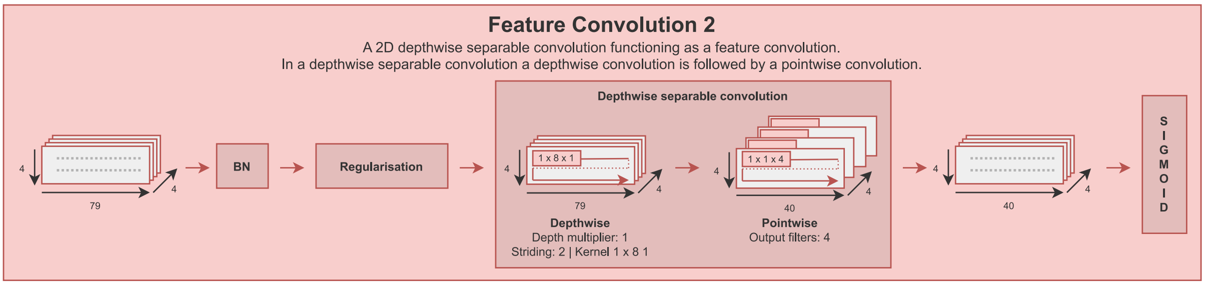

Feature Convolution 2
A 2D depthwise separable convolution functioning as a feature convolution.
In a depthwise separable convolution a depthwise convolution is followed by a pointwise convolution.
Depthwise separable convolution
4
79
BN
Regularisation
1 x 8 x 1
79
Depthwise
Depth multiplier: 1
Striding: 2 | Kernel 1 x 8 1
1 x 1 x 4
40
Pointwise
Output filters: 4
40
S I G M O I D

Fig. 5. In-depth view of the second feature convolution within ECGencode, configured per ECGencode model 1 specified in Fig. 1. The input is the output of the first feature convolution depicted in Fig. 4. A depthwise separable convolution with a kernel of 1 × 8 × 1 and a stride of two is used. The pointwise convolution has a kernel of 1 × 1 × 4 and four output filters.

weights, as detailed in Section 3.2. The output from the spatial convolution block is activated using the Exponential Linear Unit (ELU) function, which introduces non-linearity into the network and addresses the vanishing gradient problem (Clevert et al., 2016). It is noteworthy that the preceding temporal convolution intentionally omits non-linearity as it increases the computational complexity without improving performance, aligning with the design decisions of the EEGNet architecture (Lawhern et al., 2018).

3.1.3. *Feature convolutions*

The feature convolutions in ECGencode serve to further refine and compact the latent space. This stage automates the traditional process of manual feature extraction that follows the application of FBCSP, learning a final latent space directly from the data. Fig. 4 provides details on the first feature convolution, which uses a depthwise separable convolution to achieve computational efficiency while generating a more compact feature representation. Specifically, the depthwise convolution employs a kernel of 1 × 32 × 1 with a stride of eight, resulting in an output of dimensions 4 × 79 × 16. A subsequent pointwise convolution with a kernel of 1 × 1 × 16 produces four output filters, leading to an output of size 4 × 79 × 4.

The second feature convolution, shown in Fig. 5, builds upon the output of the first. It also employs a depthwise separable convolution but with a kernel of 1 × 8 × 1 and a stride of two. This convolution retains four output filters, yielding a final latent space of dimensions 4 × 40 × 4.

Depthwise separable convolutions, as further explained in Section 3.3, contribute to the computational efficiency of ECGencode while leveraging cross-channel information to construct the final latent space. Both feature convolutions also incorporate channel-based BN followed by an ECG-specific novel Spatial Gaussian Noise regularisation technique, which is further discussed in Section 3.2. This regularisation improves both the stability and generalisability of ECGencode during training. While the first feature convolution continues to use the ELU activation function, the second employs a sigmoid activation function. The sigmoid activation ensures that all features in the final latent space lie within the 0 to 1 range, beneficial for contexts requiring a probabilistic interpretation of these features.

3.2. *ECG specific normalisation and regularisation*

ECGencode incorporates two techniques to enhance training stability and performance: Batch normalisation (BN) on different axes and a novel spatial Gaussian noise regularisation technique. Additionally, the data used in this study has undergone minimal preprocessing through ECG-device-specific normalisation, as discussed in Section 4.1.

3.2.1. *Different axis batch normalisation*

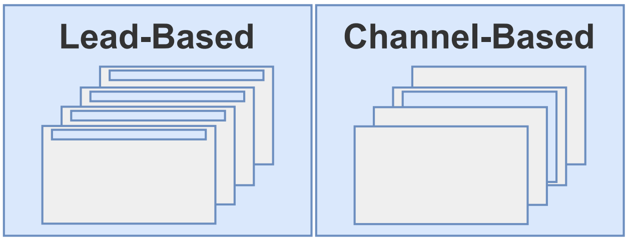
BN plays a significant role in the spatial and feature convolution blocks of ECGencode. During training, BN normalises its output over a specified axis using the mean and variance statistics computed over a mini-batch of training samples. During inference, the model uses a moving average of these statistics, obtained during training, instead (Ioffe & Szegedy, 2015).

By mitigating the internal covariate shift problem, BN facilitates faster and more stable learning, while also reducing reliance on specific weight initialisation choices. Additionally, it provides a form of implicit regularisation, thus limiting the risk of overfitting (Luo et al., 2019). Due to its benefits and low computational cost during both training and inference, BN is integrated into ECGencode where applicable.

In the feature convolution blocks, BN is performed on the channel axis (feature maps) as is conventional in literature. However, in the spatial convolution block, where each channel undergoes independent processing through the use of depthwise convolution, a lead axis-based BN is employed. This allows a more intrinsic interpretation of the learned kernel weights for the spatial convolution, which can aid the interpretability.

3.2.2. *Novel spatial Gaussian noise regularisation*

To explicitly enhance regularisation and prevent overfitting while having minimal impact on the training speed, ECGencode introduces a novel technique which has been named Spatial Gaussian Noise.



(a) Lead 1 considered

(b) Channel 3 considered

Fig. 6. Visualisation of data considered for lead-axis-based and channel-axis-based normalisation and regularisation.

Spatial Gaussian Noise, used in the feature convolutions of ECGencode, combines concepts from spatial dropout (Tompson et al., 2015) and Gaussian noise regularisation.

In spatial dropout, a complete slice of the channel axis in the input data is zeroed out with a given probability. This technique is favoured for time series data like ECG, where neighbouring data points exhibit strong correlation, making it more effective than regular dropout which would randomly zero out individual data points rather than a complete slice.

However, due to ECGencode's compact latent space, fully nullifying an entire slice during spatial dropout can lead to excessive regularisation, negatively impacting the learning speed and overall model performance. To address this, a custom spatial Gaussian noise regularisation technique is employed. This technique applies multiplicative Gaussian noise with a user-defined mean and standard deviation to values from a slice in the specified axis with a given probability. Empirical findings demonstrate that combining regular spatial dropout using a low probability followed by the custom spatial Gaussian noise with a higher probability yields the most effective regularisation and generalisation performance for ECGencode. The spatial Gaussian noise regularisation for the first feature convolution (Fig. 4) happens on a lead-axis basis whereas the spatial Gaussian noise regularisation in the second feature convolution (Fig. 5) happens on a channel-axis basis. The difference between which data is considered for the different axis-based regularisation is shown in Fig. 6. Notably, as this regularisation is only performed during training, it has no impact on computational efficiency during inference.

3.3. Computational efficiency through depthwise and depthwise separable convolutions

To efficiently extract hierarchical features from input ECG signals, ECGencode employs convolutional layers. Traditional 2D convolutions, although effective, are computationally demanding in terms of FLOPs, and can require many trainable parameters. To mitigate this computational burden, ECGencode incorporates depthwise and depthwise separable convolutions, offering a more efficient computational profile where applicable.

Figs. 7–9 provide a visual demonstration of the FLOPs required for these three types of convolutions: standard 2D, depthwise 2D, and depthwise separable 2D. For comparative clarity, each convolutional type is applied to the same input data (12 × 5000 × 16) and configured to produce the same output shape (1 × 5000 × 64). It should be noted that the FLOPs calculations presented are theoretical estimates based on a straightforward CPU implementation of these algorithms without padding. It is also mentioned that ECGencode does not use bias terms for its various convolutions. Not only does this save additional parameters and FLOPs, but the use of BN after the convolutions makes the use of a bias term in the convolution redundant (Ioffe & Szegedy, 2015).

3.3.1. Standard 2D convolution

As depicted in Fig. 7, a standard 2D convolution effectively involves a 3D convolutional operation, as it incorporates both the 2D signal dimensions and the input channels as the third dimension. The kernel used in this type of convolution performs element-wise multiplications and additions on this 3D input to produce each output channel.

Given an input with dimensions $H \times W \times C_{\text{in}}$, a kernel of size $K_1 \times K_2 \times C_{\text{in}}$, a stride S, and C_{out} output channels, the FLOPs for this operation can be calculated using Eq. (1). Here, H', W', K'_{standard}, and $\text{Bias}_{\text{standard}}$ represent the output height, output width, FLOPs for each kernel pass, and the FLOPs for bias addition, respectively. These terms are defined in Eq. (2). Note that K'_{standard} is multiplied by two to account for both multiplication and addition for each weight in the kernel.

Using Eq. (1) for the standard convolution visualised in Fig. 7, the total FLOPs is found to be 123,200,000.

$$\text{FLOPs}_{\text{standard}} = (K'_{\text{standard}} \times H' \times W' \times C_{\text{out}}) + \text{Bias}_{\text{standard}} \tag{1}$$

$$\begin{aligned} H' &= \left\lceil \frac{H - K_1 + 1}{S} \right\rceil \\ W' &= \left\lceil \frac{W - K_2 + 1}{S} \right\rceil \\ K'_{\text{standard}} &= 2 \times (K_1 \times K_2 \times C_{\text{in}}) \\ \text{Bias}_{\text{standard}} &= C_{\text{out}} \times H' \times W' \end{aligned} \tag{2}$$

The number of parameters for a standard 2D convolution is given by Eq. (3), which accounts for both the kernel weights and the bias terms for each output channel. For the standard 2D convolution in Fig. 7 with 64 output kernels, the total number of parameters is 12,352.

$$\text{Params}_{\text{standard}} = ((K_1 \times K_2 \times C_{\text{in}}) + 1) \times C_{\text{out}} \tag{3}$$

3.3.2. Depthwise 2D convolution

Fig. 8 showcases depthwise 2D convolution, a more computationally efficient variant that processes each input channel separately. This channel-wise operation eliminates the fusion of information across different input channels, substantially reducing both FLOPs and the number of parameters.

A depth multiplier D is introduced to control the output channel count, allowing it to be a multiple of the input channels. The multiplier D determines the number of output channels generated per input channel, effectively specifying the number of distinct kernels per input channel.

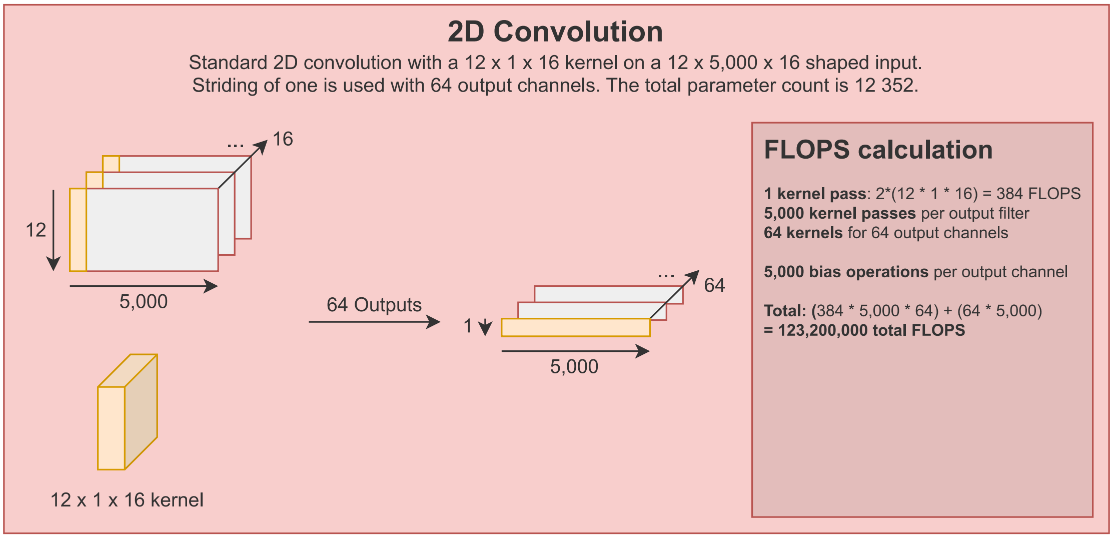
Given an input with dimensions $H \times W \times C_{\text{in}}$, kernel dimensions $K_1 \times K_2 \times C_{\text{in}}$, a stride S, and a depth multiplier D, the FLOPs for this operation are governed by Eq. (4). Here, H' and W' are as defined in Eq. (2), and K'_{depth} and $\text{Bias}_{\text{depth}}$ are outlined in Eq. (5).

Application of Eq. (4) to the depthwise convolution in Fig. 8 yields a total of 8,000,000 FLOPs. Remarkably, this constitutes just 6.49% of the FLOPs required for a standard 2D convolution with identical input and output dimensions. The ratio $\frac{\text{FLOPs}_{\text{depthwise}}}{\text{FLOPs}_{\text{standard}}}$ can be roughly approximated as $\frac{D}{C_{\text{out}}}$, highlighting the computational advantages of depthwise 2D convolutions when D is significantly smaller than C_{out}.

$$\text{FLOPs}_{\text{depthwise}} = (K'_{\text{depth}} \times H' \times W' \times C_{\text{in}} \times D) + \text{Bias}_{\text{depth}} \tag{4}$$

$$\begin{aligned} K'_{\text{depth}} &= 2 \times (K_1 \times K_2 \times 1) \\ \text{Bias}_{\text{depth}} &= C_{\text{in}} \times D \times H' \times W' \end{aligned} \tag{5}$$

The parameter count for depthwise 2D convolutions is calculated using Eq. (6). For the instance in Fig. 8 with 16 input channels and



2D Convolution

Standard 2D convolution with a 12 x 1 x 16 kernel on a 12 x 5,000 x 16 shaped input.
Striding of one is used with 64 output channels. The total parameter count is 12 352.

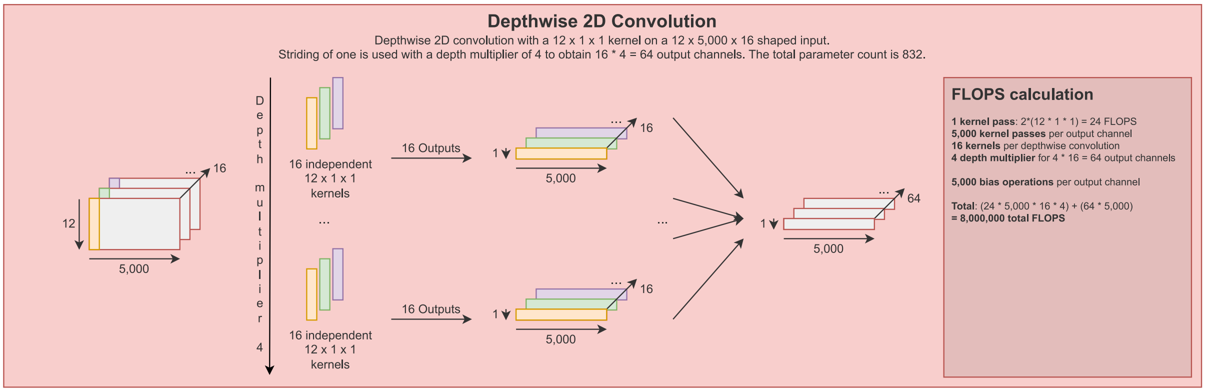
FLOPS calculation

1 kernel pass: 2*(12 * 1 * 16) = 384 FLOPS
5,000 kernel passes per output filter
64 kernels for 64 output channels

5,000 bias operations per output channel

Total: (384 * 5,000 * 64) + (64 * 5,000)
= 123,200,000 total FLOPS

Fig. 7. Operational steps, parameters and FLOPs analysis for a standard 2D convolution.



Depthwise 2D Convolution

Depthwise 2D convolution with a 12 x 1 x 1 kernel on a 12 x 5,000 x 16 shaped input.
Striding of one is used with a depth multiplier of 4 to obtain 16 * 4 = 64 output channels. The total parameter count is 832.

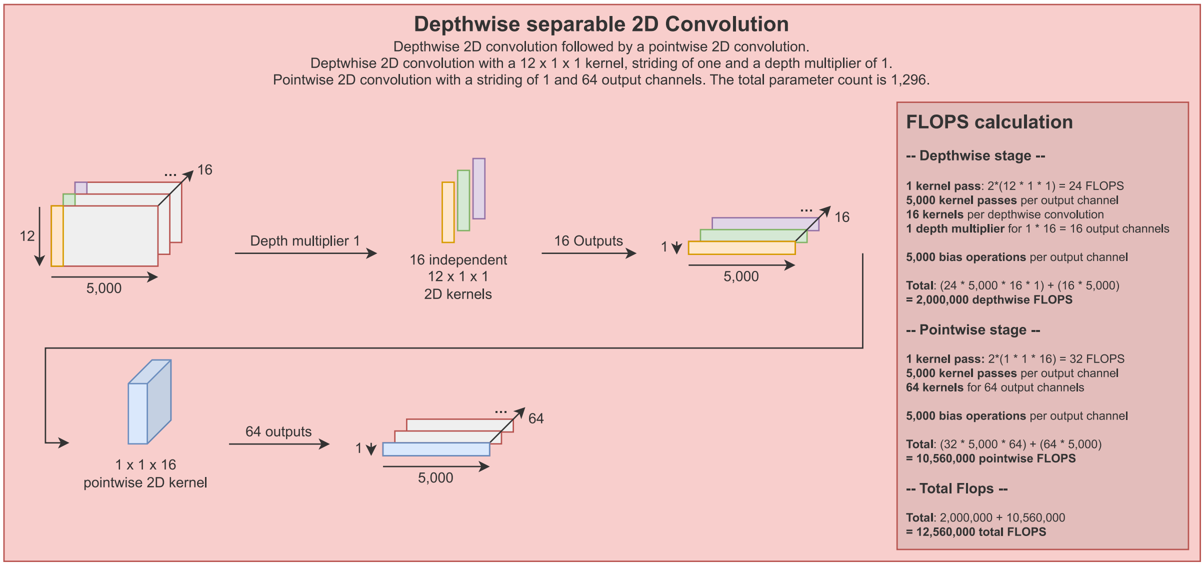
FLOPS calculation

1 kernel pass: 2*(12 * 1 * 1) = 24 FLOPS
5,000 kernel passes per output channel
16 kernels per depthwise convolution
4 depth multiplier for 4 * 16 = 64 output channels

5,000 bias operations per output channel

Total: (24 * 5,000 * 16 * 4) + (64 * 5,000)
= 8,000,000 total FLOPS

Fig. 8. Operational steps, parameters and FLOPs analysis for a depthwise 2D convolution.



Depthwise separable 2D Convolution

Depthwise 2D convolution followed by a pointwise 2D convolution.
Depthwise 2D convolution with a 12 x 1 x 1 kernel, striding of one and a depth multiplier of 1.
Pointwise 2D convolution with a striding of 1 and 64 output channels. The total parameter count is 1,296.

FLOPS calculation

-- Depthwise stage --

1 kernel pass: 2*(12 * 1 * 1) = 24 FLOPS
5,000 kernel passes per output channel
16 kernels per depthwise convolution
1 depth multiplier for 1 * 16 = 16 output channels

5,000 bias operations per output channel

Total: (24 * 5,000 * 16 * 1) + (16 * 5,000)
= 2,000,000 depthwise FLOPS

-- Pointwise stage --

1 kernel pass: 2*(1 * 1 * 16) = 32 FLOPS
5,000 kernel passes per output channel
64 kernels for 64 output channels

5,000 bias operations per output channel

Total: (32 * 5,000 * 64) + (64 * 5,000)
= 10,560,000 pointwise FLOPS

-- Total Flops --

Total: 2,000,000 + 10,560,000
= 12,560,000 total FLOPS

Fig. 9. Operational steps, parameters and FLOPs analysis for a depthwise separable 2D convolution.

a depth multiplier of 4, the total is 832 parameters. This accounts for merely 6.74% of the parameters needed for a standard 2D convolution.

$$\text{Params}_{\text{depthwise}} = ((K_1 \times K_2) + 1) \times (C_{\text{in}} \times D) \tag{6}$$

3.3.3. Depthwise separable 2D convolution

As illustrated in Fig. 9, depthwise separable 2D convolution builds upon the efficiency of depthwise 2D convolution but reintegrates cross-channel information. This two-step process consists of an initial depthwise 2D convolution followed by a pointwise 2D convolution. The former operates identically to the previously described depthwise convolution, whereas the latter employs a standard convolution with a kernel size of $1 \times 1 \times C_{\text{in}}$, facilitating the merging of information from the depthwise output channels.

Employing Eq. (7), the FLOPs for this convolution type can be calculated. Here, H' and W' align with those in Eq. (2), and K'_{depth} and $\text{Bias}_{\text{depth}}$ are consistent with those in Eq. (5). K'_{point} and $\text{Bias}_{\text{point}}$ are formulated in Eq. (8).

For the depthwise separable convolution presented in Fig. 9, Eq. (7) yields a total of 12,560,000 FLOPs. This constitutes merely 10.19% of the FLOPs required for a standard convolution of identical dimensions, yet is 1.57 times greater than that of a depthwise convolution. Therefore, depthwise separable convolutions offer a nuanced balance between computational efficiency and channel mixing, particularly beneficial in the feature convolution stage of ECGencode.

$$\begin{aligned}\text{FLOPs}_{\text{separable}} &= \text{FLOPs}_{\text{depthwise}} + \text{FLOPs}^{\text{pointwise}}_{\text{standard}} \\ &= \langle (K'_{\text{depth}} \times H' \times W' \times C_{\text{in}} \times D) \\ &\quad + \text{Bias}_{\text{depth}} \rangle \\ &\quad + \langle (K'_{\text{point}} \times H' \times W' \times C_{\text{out}}) \\ &\quad + \text{Bias}_{\text{point}} \rangle \end{aligned} \tag{7}$$

$$\begin{aligned} K'_{\text{point}} &= 2 \times (1 \times 1 \times C_{\text{in}} \times D) \\ \text{Bias}_{\text{point}} &= C_{\text{out}} \times H' \times W' \end{aligned} \tag{8}$$

Employing Eq. (6) for the depthwise stage yields 208 parameters, whilst Eq. (3) for the pointwise stage (standard 2D convolution with a kernel of size $1{\times}1{\times}C_{\text{in}}$) results in 1088 parameters, totalling 1296. This is a mere 10.49% of the parameters required for a standard convolution.

3.3.4. Striding over pooling

Conventional convolutional neural networks often employ a combination of convolutional layers for feature extraction and pooling layers for latent space downscaling. However, it has been demonstrated that substituting pooling layers with convolutional layers that use increased striding can enhance model performance, as this allows the network to effectively learn a downscaling strategy (Springenberg et al., 2015). Although employing striding in place of pooling preserves the FLOPs count, it increases the model's parameter count due to the learnable nature of the downscaling convolution. To achieve computational efficiency while maintaining compactness, ECGencode integrates striding directly into its primary convolutional layers, thereby eliminating the need for separate downscaling convolutions and consequently decreasing both FLOPs and parameters count. Empirical findings show that the integration of these two convolutions into one results in a negligible performance decrease for ECGencode whilst requiring considerably fewer parameters and FLOPs.

3.4. Controllable and extendable feature complexity

Designed for versatility, ECGencode offers a rich configuration space through intuitive configuration parameters, facilitating adaptation to various latent space complexities and problem settings. In addition to this inherent flexibility, ECGencode also easily supports extensions, such as the incorporation of LSTM units. This section discusses these two primary directions for customising ECGencode to address a diverse range of applications.

3.4.1. Parameter-driven control of latent space complexity

ECGencode enables control over its complexity via multiple intuitive configuration parameters, with the number of augmented leads, time points, and output channels being the most important.

Augmented Leads and Time Point Regulation

The number of parallel spatial convolutions directly correlates with the number of augmented leads, which primarily encode spatial information. Although there is no explicit upper limit on the number of augmented leads, a practical upper bound is suggested to be eight. This number aligns with the eight physical leads used in 12-lead ECG recordings, where aVR, aVL, aVF, and III are derived as linear functions of leads I and II (Attia, Noseworthy, et al., 2019).

Control over the amount of time points retained in the latent space is achieved through the manipulation of striding parameters. Early-stage striding is recommended for significant FLOP reduction in later stages, especially when working with a high-resolution input of 500 Hz. This makes the increase of striding favourable in early stages whilst the decrease of striding is most computationally efficient in later stages.

Adjustment of Output Channels

ECGencode's internal and external complexity can be further fine-tuned through the number of output channels in its temporal and feature convolution components. The amount of different length kernels and output channel counts in the temporal convolution stage influences internal complexity. The final pointwise step in the last feature convolution is what ultimately decides the amount of latent space output channels and thus the external complexity. Increasing the depth multiplier in the depthwise step of the feature convolutions offers an additional lever for the internal complexity of ECGencode. In practice, aligning internal and external complexities yields optimal performance.

3.4.2. Incorporating advanced extensions

While the ECGencode configuration presented in Fig. 1 serves as a highly compact and computationally efficient setup for binary ECG classification, featuring a flattened latent space and a singular softmax-activated dense layer, ECGencode is designed to support sophisticated extensions. These extensions are enabled by the convolutional nature of ECGencode which retains the sequential patterns present in ECG signals.

LSTM for Sequential Modelling

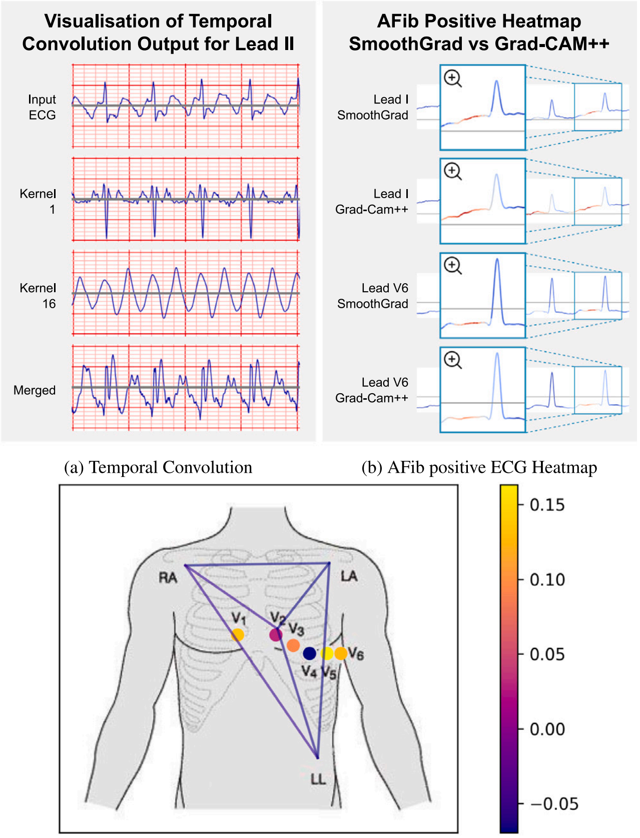
Transforming the ECGencode output from its initial 3D format (augmented leads × time points × channels) to a 2D configuration (augmented time points × (augmented leads $*$ channels)) enables the integration of LSTM units. This results in a hybrid CNN-LSTM model capable of exploiting the inherent temporal dynamics of ECG signals. Although beneficial for context-sensitive feature extraction, this extension demands significant additional computational resources in terms of both parameters and FLOPs and as such is not suitable for all applications.

Complex Output Models

Task-specific requirements may necessitate more intricate output models. Incorporating a fully connected dense layer before the softmax layer can be advantageous for complex ECG classification tasks, including multi-class and multi-label scenarios. While the experiments in this paper are limited to ECG classification, ECGencode is versatile enough for a range of applications, including ECG regression tasks which also require effective ECG feature encoding.

3.5. ECGencode-specific visualisation

Inspired by the proven FBCSP technique (Ang et al., 2008), ECGencode has been designed such that each layer fulfils a distinct role, as detailed in Section 3.1. This structured design enhances both the intrinsic interpretability of the learned parameters and post hoc visualisation capabilities. A preliminary examination of both these



(a) Temporal Convolution

(b) AFib positive ECG Heatmap

(c) Spatial Convolution Lead Weights

Fig. 10. Various visualisation techniques applied to the first ECGencode model for binary ECG classification. Fig. 10(a) depicts the initial segment of Lead II from an input ECG, both in its raw form and as processed by temporal convolutions using 2D kernels of dimensions 1 × 16 × 1 (Kernel 1) and 1 × 1024 × 1 (Kernel 16), along with the combined output from all 16 temporal convolution kernels. Fig. 10(b) presents both a SmoothGrad saliency map and a Grad-Cam++ class activation map for the same segments on a correctly classified AFib-positive ECG. Fig. 10(c) reveals the relative importance of each lead in generating a single augmented lead in the spatial convolution, based on the intrinsic evaluation of the learned kernel weights.

aspects is presented below, demonstrating that the layers within ECGencode exhibit the anticipated behaviour, and confirming the effectiveness of the architectural design. These preliminary visualisations and interpretations reveal ECGencode's potential to facilitate advanced medical interpretability, positioning ECGencode as a promising tool for further, medically validated, investigative research towards model explainability.

3.5.1. *Insights into the temporal convolution*

Designed to emulate the frequency filtering stage of the FBCSP technique, the temporal convolution component within ECGencode is structured to capture both high-frequency and low-frequency attributes from the input ECG signal. Visualisation of individual channels of this component reveals the varying length kernels behave as expected. As illustrated in Fig. 10(a), shorter kernels (i.e., Kernel 1) predominantly capture high-frequency details, whereas longer kernels (i.e., Kernel 16) emphasise the lower-frequency elements of the input ECG.

3.5.2. *Interpretation of spatial convolution*

The spatial convolution component in ECGencode utilises its learned kernel weights to determine the significance of corresponding input leads for generating the augmented leads. Each kernel in this component, responsible for generating one of the output augmented leads, consists of 12 weights, one for each input lead. Due to the preceding lead-specific BN, these learned weights directly indicate the extent of influence each lead has in the creation of the augmented lead. Leads with weights close to zero contribute minimally to the final augmented lead and thus to the final prediction. Likewise, leads with weights that have a larger absolute value contribute more to the final augmented lead.

This intrinsic interpretation of the learned weights enables a topographic visualisation, where the weights assigned to individual leads can be spatially mapped. This visualisation technique, illustrated in Fig. 10(c), can assist medical staff by highlighting which lead in the original 12-lead input signal contains the most prominent information relevant to the diagnosis.

3.5.3. *Utilisation of conventional visualisation techniques*

Conventional visualisation methodologies from the broader domain of deep learning offer additional ways for interpreting models which employ ECGencode. This includes gradient-based class activation maps, such as GradCAM (Selvaraju et al., 2020), GradCAM++ (Chattopadhay et al., 2018), ScoreCAM (Wang et al., 2020), LayerCAM (Jiang et al., 2021), as well as saliency maps, such as vanilla saliency (Simonyan et al., 2014) and SmoothGrad (Smilkov et al., 2017). These techniques are commonly used for ECG analysis models to provide heatmaps that indicate critical regions influencing the model's decisions (Jahmunah et al., 2022; Kim et al., 2022; Tohyama et al., 2023).

Given ECGencode's custom layers are disabled during inference, these existing techniques can be applied directly to models incorporating ECGencode. An application of both a SmoothGrad saliency map and a Grad-Cam++ class activation map for the same segments on a correctly classified AFib ECG by the first ECGencode model, is depicted in Fig. 10(b). The heatmaps generated by these methods highlight the P-wave regions in the ECG signal, an area recognised to exhibit diminished or absent activity in AFib-positive patients (Goodacre & Irons, 2002).

4. Evaluation and results

This section evaluates ECGencode's versatility, computational efficiency, and performance across four distinct ECG classification tasks, employing ECGencode as a feature encoder in two separate deep learning models. Both configurations are assessed using open-source data sets and benchmarked against SOTA techniques, highlighting ECGencode's potential in real-world applications.

4.1. *Available data sets*

ECGencode is evaluated using the PTB-XL (Goldberger et al., 2000; Wagner et al., 2020, 2022) and CODE-15% (Lima et al., 2021; Ribeiro et al., 2021, 2020) open-source data sets. These data sets are among the most extensive in the public domain, offering a reliable platform for assessing generalisability and benchmarking against SOTA methodologies. While they provide valuable insights into real-world performance, it should be noted that they cannot fully replicate the breadth of data typically available in private clinical settings.

Both data sets are utilised in their original formats, except for per-device normalisation and the upsampling of CODE-15% to 500 Hz to align with PTB-XL. These preprocessing steps aim to improve the models' robustness and transferability, mitigating device-specific biases and enhancing generalisability across different ECG recording devices and data sets.

4.1.1. PTB-XL data set

The PTB-XL data set comprises 21,837 standardised 10-second 12-lead ECG recordings (Goldberger et al., 2000; Wagner et al., 2020, 2022). Sourced from 18,885 distinct patients between October 1989 and June 1996, these recordings employ various Schiller AG ECG devices and exhibit a data shape of 12 × 5000 due to a 500 Hz sampling rate. On average, 1.16 ECGs per patient are present in the data set.

Being a multi-label data set, PTB-XL assigns one or more labels to each ECG, reflecting the large variety of cardiac conditions and possible combinations of them found in real-world data. The PTB-XL data set features 71 diverse labels, which are different types of diagnostic statements, ranging from rhythm to form statements, with a distribution that approximates actual clinical prevalence rates. For instance, it encompasses 9528 normal ECGs (43.63%), contrasted with rare diagnostic categories like second-degree AV block which only has 14 samples (0.06%). The median age of the data set's patients is 62, with an interquartile range of 22. Additional metadata, such as the ECG device and recording date, are also available.

For a comprehensive overview and download details of the PTB-XL data set, refer to the work by Wagner et al. (2020). Benchmarking information and performance metrics are available in the work of Strodthoff et al. (2021).

4.1.2. CODE-15% data set

The CODE-15% data set contains 345,779 12-lead ECG exams, each lasting either 7 or 10 seconds (Lima et al., 2021; Ribeiro et al., 2021, 2020). These exams were collected between 2010 and 2016 by the Telehealth Network of Minas Gerais (TNMG) in Brazil and originate from 233,770 distinct patients. Representing a stratified 15% subset of the larger, non-publicly available CODE data set, CODE-15% averages 1.48 ECGs per patient. To have a uniform data shape of 12 × 4096 for both 7 and 10-second ECGs in the CODE-15% data set, zero padding is pre-applied to the 400 Hz signals. For compatibility with PTB-XL, this zero padding is removed from the CODE-15% ECGs such that they can be upsampled to 500 Hz and re-padded in case of the 7-second ECGs to have a final, shared with PTB-XL, data shape of 12 × 5000.

CODE-15% includes seven diagnostic labels such as first-degree AV block and AFib. In contrast to PTB-XL, this data set is less rich in metadata; it lacks ECG device details, necessitating a generalised normalisation process and only the age of the patient at the time of recording is available, limiting the ability to determine the exact recording date.

For further insights into the CODE-15% data set, consult the works by Ribeiro et al. (2021, 2020) and Lima et al. (2021).

4.2. Experimental setup and evaluation metrics

To assess the performance of the ECGencode feature encoder whilst demonstrating its intuitive model configuration parameterisation and versatility, two custom deep learning models incorporating ECGencode have been constructed for four distinct ECG classification tasks. The first model prioritises computational efficiency without loss of classification performance, targeting three binary ECG classification tasks related to AFib presence on the ECG. The second model features a more complex architecture that supplements the ECGencode latent space with Long Short-Term Memory (LSTM) units, designed for multi-label ECG classification. This more complex ECGencode model 2 aims to demonstrate how an extension to ECGencode can be made using intuitive reasoning over the model configuration parameters and latent space, whilst maintaining a low parameter count and SOTA matching performance. These models undergo evaluation both within an isolated test partition of the originating data set and on an entirely separate, previously unseen data set for the binary ECG classification tasks. This provides a comprehensive performance assessment through multiple reported metrics.

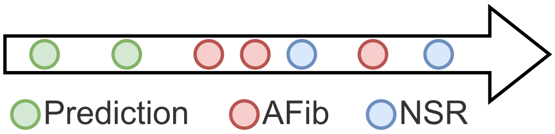


Fig. 11. Temporal alignment of all ECGs from an AFib-positive patient categorised as prediction, AFib, and NSR ECGs. Prediction ECGs precede the first AFib-labelled ECG, meaning the patient was not known to be AFib positive yet. NSR ECGs follow the first AFib-labelled ECG but are not labelled as AFib themselves.

4.2.1. ECGencode model 1: Binary ECG classification

ECGencode model 1, illustrated in Fig. 1 and detailed in Section 3.1, is optimised for binary ECG classification with a focus on achieving high computational efficiency, measured both in terms of parameters and FLOPs, without loss of classification performance. The configuration of model 1 is as follows:

- **Temporal convolution**: Kernels of temporal length 16, 64, 256 and 1024 spanning 0.03, 0.1, 0.5, and 2 seconds, respectively. Striding of eight for significant temporal resolution and FLOPs reduction. Each kernel has 4 output channels, totalling a computationally manageable 16.
- **Spatial convolution**: 4 augmented leads for significant temporal downscaling, based on the optimal found four number of leads by Lai et al. (2021).
- **Feature convolution 1**: Depthwise kernel of temporal length 32 with 4 pointwise output channels. A depth multiplier of 1 and a stride of 8 is used.
- **Feature convolution 2**: Depthwise kernel of temporal length 8 with 4 pointwise output channels. Depth multiplier of 1 and a stride of 2. These parameters were chosen to obtain a compact output latent space.
- **ECGencode output shape**: 4 × 40 × 4.
- **ECGencode parameters**: 6960.
- **Extension**: Simple 1D flatten.
- **Classification**: softmax activated dense layer of 2 units.
- **Total model parameters**: 8242.
- **Total model FLOPs**: ±83M.

ECGencode model 1 is trained using the CODE-15% data set, partitioned into a training (80%), validation (10%), and test (10%) set through a stratified strategy, ensuring that the label distribution is maintained and no patients overlap exists between the sets. All ECGs from a patient with at least one AFib-positive ECG are considered a positive sample, this includes the prediction, AFib and NSR ECGs as depicted in Fig. 11. All ECGs from patients without any AFib association are considered negative samples, which is inspired by the experimental setup of Attia, Noseworthy, et al. (2019) for similar experiments. It is important to note that due to the limited metadata of CODE-15%, which does not include the exact recording date of an ECG, the temporal ordering of ECGs from a patient is based on the patient's age. Thus, for an ECG to be considered a prediction sample of AFib, the patient's age must be lower than on their first AFib-diagnosed ECG. In situations where the first AFib-diagnosed ECG is close to their birthday (e.g., one day before their birthday), this means that an ECG has to be taken at least 1 year before the first AFib-diagnosed ECG to be considered as a prediction sample. Given this already severely limits the amount of prediction samples, no upper limit for age difference to be considered a prediction sample is set. This means the prediction samples have a long time horizon, making the prediction evaluation task a very difficult one. Likewise, an ECG without AFib diagnosis is considered an NSR sample when the patient's age is identical or higher than on their first AFib-diagnosed ECG. This means that some samples considered NSR in the NSR test set could be recorded before the first AFib-diagnosed ECG, as ordering them in time is impossible when the patient's age is

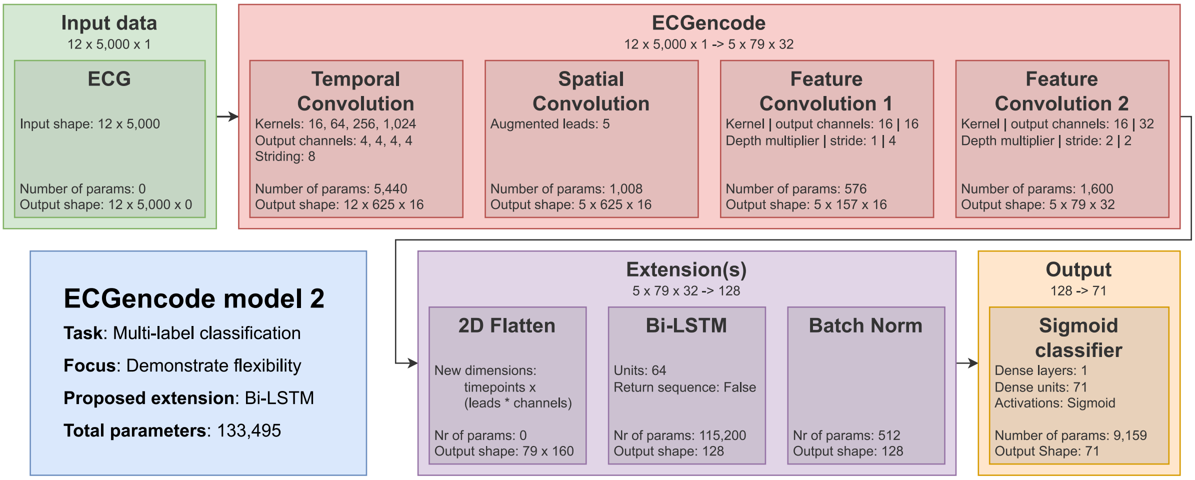


Fig. 12. A high-level overview of ECGencode model 2 used for multi-label ECG classification of 71 classes. This model, and the ECGencode configuration it uses, focus on demonstrating ECGencode's flexibility and extendibility. The 2D input consists of a standard 10-second 12-lead ECG sampled at 500 Hz, represented as a 12×5000 matrix. ECGencode outputs a 3D latent space with dimensions $5 \times 79 \times 32$ which is flattened into 2D by merging the channel and lead dimensions. A bi-directional LSTM layer with 64 units in each direction is used to create a CNN-LSTM model. A fully connected layer with sigmoid activation accomplishes the multi-label ECG classification. ECGencode Model 2 has 133,495 parameters in total.

identical. In this sense, the NSR detection test task likely contains what is intuitively considered "prediction samples", but due to the limited metadata cannot be labelled as such.

For the training set, this corresponds to 7865 positive samples, of which only 1677 are NSR samples and 983 are prediction samples, contrasted to the large set of negatives which consists of 271,643 ECGs. For the validation set this results in 407 NSR samples and 238 prediction samples in the positive set of 1432 samples and 29,547 negative samples. The model selected for further use is the one obtained at the epoch where the validation sensitivity is at its highest. This choice is based on wanting to train a model that correctly predicts as many of the positive samples given the heavy class imbalance. Training is conducted for 2500 epochs using an AdamW optimizer and an alpha-balanced categorical focal cross-entropy loss function to take into account the heavy class imbalance of the training data, typical for ECG analysis tasks (Lin et al., 2020; Loshchilov & Hutter, 2019; Romdhane et al., 2020). The thresholds used for converting the continuous values to labels are optimised for achieving the highest validation F1 score.

4.2.2. ECGencode model 2: Multi-label ECG classification

To demonstrate ECGencode's intuitive model configuration, versatility, and ECG-structured latent space, a second ECGencode model targeting multi-label classification is proposed. ECGencode model 2, visualised in Fig. 12, has been designed for classifying 71 class labels in a multi-label task and is an extension of the binary model proposed in Section 4.2.1. The configuration of ECGencode has been modified to boast more complex internal and external representations and an LSTM extension has been added following the guidelines discussed in Section 3.4. These modifications have been done by reasoning over the ECGencode configuration parameters and generated latent space directly, without optimising the model architecture through computationally expensive methods. The proposed LSTM extension demonstrates the possibility of sequential modelling using ECGencode as elaborated in Section 3.4. The resulting CNN-LSTM architecture is thus a more complex model compared to the first ECGencode model but one that is capable of achieving performance on par with the PTB-XL benchmark models proposed by Strodthoff et al. (2021), as discussed in Section 4.4. The exact configuration of ECGencode model 2 as compared to ECGencode model 1 is given below:

- **Temporal convolution**: Kernels of temporal length 16, 64, 256 and 1024. Striding of eight. Each kernel has 4 output channels, totalling 16. This is unchanged compared to ECGencode model 1 to maintain a high temporal reduction early on, helping save significant FLOPs later in the model.
- **Spatial convolution**: 5 augmented leads, a slight increase from ECGencode model 1's four augmented leads, preserving a higher spatial resolution.
- **Feature convolution 1**: Depthwise kernel of temporal length 16 with 16 pointwise output channels. Depth multiplier of 1 and a stride of 4. This uses a lower striding compared to ECGencode model 1, preserving a higher temporal resolution, and more channels for a more complex internal representation.
- **Feature convolution 2**: Depthwise kernel of temporal length 16 with 32 pointwise output channels. Depth multiplier of 2 and a stride of 2. This uses a lower striding compared to ECGencode model 1, preserving a higher temporal resolution, and more channels for a more complex external representation.
- **ECGencode output shape**: $5 \times 79 \times 32$, contrasted to ECGencode model 1's more compact latent space of $4 \times 40 \times 4$.
- **ECGencode parameters**: 8624, contrasted to ECGencode model 1's 6960.
- **Extension**: 2D flatten followed by a bi-directional LSTM layer with 64 units per direction and finally a channel-based BN layer.
- **Classification**: sigmoid activated dense layer of 71 units.
- **Total model parameters**: 133,495, contrasted to ECGencode model 1's 8242.

ECGencode model 2 is trained using the PTB-XL data set for classifying all of the 71 diagnostic statements available. The PTB-XL data set is partitioned into training (folds 1–8), validation (fold 9), and testing (fold 10) sets, following an identical setup to the one proposed by Strodthoff et al. (2021) in their PTB-XL benchmark paper. Following epoch-based experimentation using the training and validation sets, the model undergoes a final training phase for 1500 epochs using both the training and validation set as training data. An AdamW optimiser and an alpha-balanced binary focal cross-entropy loss function are utilised in this training phase (Lin et al., 2020; Loshchilov & Hutter, 2019; Romdhane et al., 2020). The thresholds used for converting the continuous values to labels are optimised for achieving the highest validation F1 score.

4.2.3. Evaluation metrics

To evaluate the performance of the devised models, multiple evaluation metrics are reported. These metrics are selected to account for the heavy class imbalance in ECG data sets. Consequently, conventional metrics such as accuracy are consciously excluded to preclude misleadingly high scores from naive models that predict solely the majority class.

Eq. (9) defines the Area Under the Receiver Operating Characteristic Curve (AUC), which serves to quantify the model's capacity for discriminating between positive and negative instances, independent of the threshold used for converting the continuous values to labels. It is defined in terms of the True Positive Rate (TPR) and the False Positive Rate (FPR). A high AUC generally means multiple good thresholds exist for achieving both a good TPR as well as a good FPR, and the threshold can be adjusted in favour of any of these rates. For the multi-label task, a macro-averaged AUC is reported.

$$\text{AUC} = \int_{x=0}^{1} \text{TPR}(\text{FPR}^{-1}(x)), dx \tag{9}$$

Besides AUC, some threshold dependent metrics, based on the number of True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN) are also reported. The precision metric, specified in Eq. (10), denotes the positive predictive value, the proportion of true positives over all of the positive predictions. The sensitivity metric, specified in Eq. (11), denotes the recall, the proportion of true positives over all the positive samples. As precision and sensitivity are correlated to each other, and an improvement in one metric often causes a reduction in the other, the F1 score is often considered to summarise these two metrics. The F1 score, defined in Eq. (12), serves as a harmonic mean of precision and sensitivity. As these threshold-dependent metrics focus on the positive samples, the specificity metric is also provided, which denotes the true negative rate, as depicted in Eq. (13). For the multi-label task, these metrics are macro-averaged across all 71 classes.

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{10}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{11}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}} \tag{12}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN}+\text{FP}} \tag{13}$$

Specific to the multi-label ECG classification task, the Hamming loss is also reported. As illustrated in Eq. (14) where $y_{i,j}$ is the target, $\hat{y}_{i,j}$ is the model output, N is the total number of samples and L is the total number of labels, this metric quantifies the fraction of erroneously predicted labels to the total number of labels.

$$\text{Hamming Loss} = \frac{1}{N \cdot L} \sum_{i=1}^{N} \sum_{j=1}^{L} \text{xor}(y_{i,j}, \hat{y}_{i,j}) \tag{14}$$

These metrics are presented as point estimates derived from the complete test set, complemented by a 95% confidence interval for the AUC and F1 metric. This confidence interval is calculated through empirical bootstrapping on the test set, encompassing 10,000 iterations. The bootstrapping methodology employed here involves sampling the test set with replacement, creating a distinct test set of the same size for each of the 10,000 iterations. This approach is consistent with the one used in the PTB-XL benchmarking paper by Strodthoff et al. (2021), enabling a direct comparison with their findings. Under this framework, a model's better performance for a specific metric is deemed statistically significant if the confidence intervals for the point estimates do not overlap.

4.3. Binary ECG classification results

The methodology employed for training ECGencode model 1 is elaborated in Section 4.2.1. Besides ECGencode model 1, the model by Attia, Noseworthy, et al. (2019), now referred to as Attia's model, and the most compact model by Phukan et al. (2023) for 10-second ECG classification, now referred to as AFibri-Net 3, were trained using the same methodology and serve as benchmark models. Attia's model, inspired by a ResNet-9 architecture (He et al., 2016), is chosen as benchmark model as it has received varied levels of clinical validation for its SOTA performance in NSR AFib detection and new-onset AFib prediction, among other ECG classification tasks (Attia, Kapa, et al., 2019; Attia, Noseworthy, et al., 2019; Christopoulos et al., 2020; Gruwez et al., 2023; Noseworthy et al., 2020; Raghunath et al., 2021). AFibri-Net 3 was chosen as it was recently proposed as a computationally efficient model for AFib detection, suitable for use on edge devices. Given the limited available metadata, traditional AFib risk scores such as CHARGE-AF by Alonso et al. (2013) were not applicable for comparison.

Evaluation of these binary models encompasses three tasks with the first being the detection of AFib-related ECGs (Related - all ECGs from AFib-positive patients are considered positive). This corresponds to the task and labelling scheme used for training ECGencode model 1. Besides this task, two more complex sub-tasks are also evaluated: NSR AFib detection (NSR) and new-onset AFib prediction (Prediction). These sub-tasks only use the NSR or Prediction ECGs from AFib-positive patients, as illustrated in Fig. 11, in the positive set and omit the other ECGs from AFib-positive patients from the test set. The first task was chosen to demonstrate how well the model learned the training task, whereas the other two tasks represent detecting a subgroup of the positives which cannot be detected through traditional methods, highlighting the benefit of DL and its use as risk prediction and screening selection tool (Attia, Noseworthy, et al., 2019; Christopoulos et al., 2020; Gruwez et al., 2023; Raghunath et al., 2021). As noted in Section 4.2.1, the limited metadata in the used data sets and the choice for no fixed maximum time-to-onset delta for the prediction samples result in the NSR detection and new-onset prediction tasks being very hard tasks.

Given the limited number of positives, 1973 for the first task, 639 for NSR detection and 293 for new-onset prediction, and a high imbalance towards negatives (33,319 for all) in the CODE-15% test split, the risk of learnable data set biases is not negligible. As such, additional assessments are carried out on the previously unseen, complete PTB-XL data set (Goldberger et al., 2000; Wagner et al., 2020, 2022).

Table 1 summarises the found evaluation metrics for these models on these data sets. Given the explained relation between sensitivity, specificity and precision, in Section 4.2, the confidence intervals and significant differences are only highlighted for the AUC and F1 metric.

Table 1 reveals several insights. First, the evaluation results show that ECGencode achieves the highest AUC and sensitivity across all tasks, demonstrating its power to match or surpass SOTA performance. This is especially notable considering the used sensitivity-focused model selection procedure, where the highest validation sensitivity models were used for collecting the results. Second, Attia's model significantly outperforms the other models in F1 score for the CODE-15% test set of the "related" task, which had the same labelling procedure as the train set, but is significantly worse than ECGencode model 1 on the PTB-XL dataset. This suggests a possible data bias is learned rather than medical properties of the task in the Attia model, further highlighting to the effectiveness of ECGencode's novel Spatial Gaussian Noise regularisation for improved generalisation results. Third, AFibri-Net 3 consistently performs statistically worse in both AUC and F1 for various tasks, indicating its computational efficiency sacrifices performance. Overall, ECGencode model 1's performance, enhanced by the novel Spatial Gaussian Noise Regularisation technique, suggests it is highly effective for these binary ECG analysis tasks while being computationally far more efficient when compared to the SOTA model, as further detailed in Section 4.5.

While this exact evaluation setup has not been considered in other works, taking into account the discussed metadata restrictions and test set configuration, the CODE-15% NSR test set is comparable to the NSR AFib detection setup of Attia, Noseworthy, et al. (2019) and the short-term prediction setup of Raghunath et al. (2021), which limits the

Table 1
Summary of binary ECG classification performance for detection of AFib-related ECGs (Related - all ECGs from AFib-positive patients are considered positive), NSR AFib detection (NSR), and new-onset AFib prediction (Prediction) tasks. Models were trained on the CODE-15% train set, considering all ECGs from AFib-positive patients as positive samples. Evaluations were performed on the CODE-15% test set and the full PTB-XL data set to examine generalisation. Metrics reported include AUC, F1 score, sensitivity, specificity, and precision, with point estimates for the test sets and confidence intervals for AUC and F1 metrics derived from 10,000 bootstrapping iterations. F1, Sensitivity, specificity, and precision are based on thresholds optimised for the highest CODE-15% validation set F1 score. Bold indicates top scores per metric and dataset whilst asterisks (*) mark scores where the AUC and F1's confidence intervals do not overlap with the highest scores, indicating significant differences.

Task	Data set	Model	AUC	F1	Sensitivity	Specificity	Precision
Related	CODE-15%	ECGencode M1	**0.9127 ± 0.0079**	0.6052 ± 0.0185*	**0.5844**	0.9795	0.6277
		Attia	0.9007 ± 0.0090	**0.6627 ± 0.0188**	0.5580	**0.9925**	**0.8156**
		AFibri-Net 3	0.8558 ± 0.0092*	0.4158 ± 0.0180*	0.4891	0.9489	0.3616
	PTB-XL	ECGencode M1	**0.9430 ± 0.0070**	**0.6372 ± 0.0207**	**0.5198**	0.9900	0.8232
		Attia	0.9137 ± 0.0083*	0.5028 ± 0.0245*	0.3518	**0,9958**	**0.8811**
		AFibri-Net 3	0.8442 ± 0.0096*	0.2522 ± 0.0238*	0.1608	0.9898	0.5842
NSR	CODE-15%	ECGencode M1	**0.8636 ± 0.0166**	0.2951 ± 0.0298	**0.3584**	0.9795	0.2508
		Attia	0.8355 ± 0.0187	**0.3272 ± 0.0381**	0.2520	**0.9945**	**0.4667**
		AFibri-Net 3	0.7993 ± 0.0184*	0.1893 ± 0.0255*	0.2457	0.9741	0.1539
	PTB-XL	ECGencode M1	**0.7367 ± 0.0372**	0.0623 ± 0.0350	**0.0694**	0.9900	0.0566
		Attia	0.6614 ± 0.0421	0.0253 ± 0.0309	0.0173	0.9970	0.0469
		AFibri-Net 3	0.6942 ± 0.0414	**0.0753 ± 0.0497**	0.0520	**0.9972**	**0.1364**
Prediction	CODE-15%	ECGencode M1	**0.7652 ± 0.0301**	**0.0897 ± 0.0270**	**0.1331**	0.9839	0.0676
		Attia	0.7408 ± 0.0301	0.0734 ± 0.0308	0.0683	**0.9930**	**0.0794**
		AFibri-Net 3	0.7238 ± 0.0296	0.0445 ± 0.0222	0.0546	0.9877	0.0376
	PTB-XL	ECGencode M1	**0.7546 ± 0.0561**	**0.0756 ± 0.0437**	**0.1058**	0.9912	**0.0588**
		Attia	0.6599 ± 0.0501	0.0423 ± 0.0442	0.0385	0.9960	0.0471
		AFibri-Net 3	0.7195 ± 0.0460	0.0282 ± 0.0433	0.0192	**0.9982**	0.0526

time-to-onset delta to a maximum of one year. This makes an indirect comparison to these works possible. Raghunath et al. (2021) report a sensitivity of 0.69 and a number needed to screen (NNS) of 9 to find one new case of AFib. This NNS translates to a precision of 0.11 ($\frac{1}{9}$). Thus, the (non-reported) F1 score of their model is calculated as $2\times\frac{0.69\times0.11}{0.69+0.11} = 0.1897$. In contrast, the F1 score for the comparable NSR CODE-15% setting of this work is 0.2951 for ECGencode Model 1, which is a reasonable difference given the slightly more challenging evaluation task of Raghunath et al. (2021). Similarly, Attia, Noseworthy, et al. (2019) report a higher F1 score of 0.392, but this is expected given their easier-to-classify test set, which includes only true NSR samples. Their model (Attia), when trained and evaluated under the experimental setup of this work, results in a similar F1 score to ECGencode Model 1. Additionally, the more challenging prediction task and cross-clinic validation on the PTB-XL test set show expected performance results in comparison. It should be noted that a different model threshold optimisation technique could be used to favour a specific metric other than F1, as is currently the case.

4.4. Multi-label ECG classification results

The training approach for the ECGencode model 2 is detailed in Section 4.2.2. The evaluation results, shown in Table 2, compare the performance of ECGencode model 2 with models from the PTB-XL benchmark study by Strodthoff et al. (2021), which uses the same evaluation framework. The xresnet1d101 model was selected for comparison due to its highest AUC value in the benchmark. Additionally, the lstm_bidir is included for its use of Bidirectional LSTM layers, similar to ECGencode model 2. For the same reasons as Strodthoff et al. (2021), the Wavelet+NN model, which employs manual feature extraction, is reported to provide contrast with more conventional, non-DL, methods.

While the PTB-XL benchmark study reports only the AUC metric and its confidence interval, ECGencode model 2's results are expanded with the additional metrics that were discussed in Section 4.2.3. Considering the overlapping AUC confidence intervals among the top-performing models, including ECGencode model 2, no significant differences were observed. However, when compared to the more traditional Wavelet+NN model, a significant difference in performance is seen. This demonstrates the effectiveness of the ECGencode model 2's architecture, which was developed simply through intuitive configuration and latent space analysis, yielding a satisfactory model without extensive tuning. Moreover, as outlined in Section 4.5, ECGencode model 2 operates with considerably fewer parameters compared to these other models.

4.5. Computational efficiency analysis

One of the main advantages of using ECGencode as a deep learning feature encoder resides in its ability to transform complex ECG inputs into a compact latent space with remarkable computational efficiency. As shown in Sections 4.3 and 4.4, this latent space serves as the foundation for ECG classification models that are competitive with SOTA alternatives in both binary and multi-label ECG classification scenarios. This section aims to quantify the computational efficiency in terms of FLOPs and model parameters. The computational metrics are calculated using the formulas delineated in Section 3.3 and further validated by the keras-FLOPs library.[1]

4.5.1. Computational efficiency of binary ECG classification models

Table 3 presents a comparative analysis of the computational demands for the evaluated binary ECG classification models in terms of the number of parameters and FLOPs. ECGencode model 1 stands out for its efficiency, requiring only 8242 parameters, a fraction (3.79%) of what is needed by the Attia model. It also operates with approximately 83 million FLOPs, just 12.39% of the Attia model's requirements. This reduction in parameters, along with the introduction of Spatial Gaussian Noise Regularisation, helps prevent overfitting, supporting the better-found model's generalisation capabilities compared to Attia's model, as discussed in Section 4.3.

The smaller number of parameters, offering lower storage requirements and reduced risk of overfitting, combined with the manageable number of FLOPs, offering lower CPU demands, make ECGencode model 1 well-suited for use on edge devices with limited computational resources. Although AFibri-Net 3 has the lowest FLOPs count, it does so

[1] https://pypi.org/project/keras-FLOPs/

Table 2
Performance of various multi-label ECG classification models on identifying all 71 diagnostic labels from the PTB-XL test set (fold 10). ECGencode model 2 underwent training across PTB-XL's folds 1–9 for 1500 epochs. Metrics reported include AUC, Hamming loss and macro-averaged F1 score, sensitivity, specificity, and precision, with point estimates for the test sets and confidence intervals for AUC and F1 metrics derived from 10,000 bootstrapping iterations. Thresholds for calculating Hamming loss, F1, sensitivity, specificity, and precision were obtained through optimisation for the highest F1 score on the validation set (fold 9). Bold indicates the top AUC score, and asterisks (*) denote significantly different scores based on non-overlapping confidence intervals with the highest AUC score. Benchmark models and their AUC values are sourced from the PTB-XL benchmark study by Strodthoff et al. (2021), which only reports AUC and their confidence intervals.

Model	AUC	F1	Sensitivity	Specificity	Precision	Hamming
ECGencode M2	0.9181 ± 0.0097	0.3265 ± 0.0214	0.3555	0.9779	0.3484	0.0276
xresnet1d101	**0.9250 ± 0.0070**	–	–	–	–	–
lstm_bidir	0.9140 ± 0.0080	–	–	–	–	–
Wavelet+NN	0.8490 ± 0.0130*	–	–	–	–	–

Table 3
Comparative analysis of computational efficiency for the evaluated binary ECG classification models, highlighting ECGencode model 1's minimal parameter count and manageable FLOPs. The relative size of both parameter count and FLOPs from the benchmark models compared to ECGencode is also provided.

Model	Parameters		FLOPs	
ECGencode M1	**8,242**	(1x)	± 83M	(1×)
Attia	217,350	(26.5x)	± 670M	(8×)
AFibri-Net 3	191,106	(23x)	± **12M**	(.15×)

Table 4
Efficiency comparison of parameter counts among multi-label ECG classification models, showcasing ECGencode model 2's low parameter footprint amidst complex tasks. Relative parameter sizes for benchmark models are presented as multiples of ECGencode's metrics. The "Wavelet+NN" model employs a hybrid approach combining wavelet transforms with neural networks, making a direct parameter count comparison not applicable.

Model	Parameters	
ECGencode M2	**133,495**	(1×)
xresnet1d101	1,880,775	(14×)
lstm_bidir	2,330,629	(17.5×)
Wavelet+NN	–	(-)

at the cost of significantly poorer classification performance, as detailed in Section 4.3. Its parameter count, while less than the Attia model, is still 23 times higher than ECGencode model 1. Notably, a significant portion of the FLOPs in ECGencode model 1 is due to the large kernel in the initial convolution layer, which is responsible for nearly 80M of the 83M FLOPs. Optimising this layer, especially by reducing the kernel sizes, could drastically decrease the FLOPs if desired, although the risk of significantly worse performance, like the AFibri-Net 3 model, should be considered.

4.5.2. *Computational efficiency of multi-label ECG classification models*

Due to the lack of publicly available libraries that support FLOPs calculations for LSTM layers and complex models such as those in the PTB-XL benchmark paper (Strodthoff et al., 2021), this computational efficiency analysis is restricted to parameter counts.

Table 4 summarises these parameter requirements of the second ECGencode model in contrast to the xresnet1d101 and lstm_bidir models from the PTB-XL benchmark paper (Strodthoff et al., 2021). Despite incorporating a computationally demanding LSTM layer, the second ECGencode model necessitates only 7.13% of the parameters as compared to the xresnet1d101 model. When compared to the lstm_bidir model, the parameter count for the second ECGencode drops to only 5.73%. This significant reduction in parameters is attributed to the LSTM layer requiring less complexity as its input complexity is already significantly reduced by ECGencode. Given the Wavelet+NN model employs a hybrid approach combining wavelet transforms with neural networks, a direct parameter count comparison is not applicable.

5. Discussion

SOTA models in ECG analysis face various training and inference challenges due to their large parameter counts and high computational demands. These challenges are particularly pronounced in environments with limited resources, such as medical edge devices, limiting their practical deployment. While some task-specific models have been developed to significantly reduce computational efficiency and allow for inference on edge devices, they are prone to significantly worse performance, as shown for the AFibri-Net 3 model in this work. These task-specific models also lack the versatility required for adopting them to other tasks and hardware settings.

Moreover, whilst some complex models allow for adjusting model complexity, such as controlling the amount of residual blocks in ResNets, this process is not straightforward due to the absence of an intuitive relation between the model configuration parameters and the specific ECG analysis tasks. This either results in the need for a computationally expensive optimisation process, which increases the risk of overfitting and bias learning, or more often, the use of default and overly complex configurations. These deep models also offer limited intrinsic interpretability in their learned parameters, requiring post hoc visualisations for some basic model interpretability.

This work proposes ECGencode as a solution to these limitations, offering a compact and computationally efficient deep learning feature encoder specifically designed to be used as a building block for DL ECG analysis models. Inspired by the FBCSP approach, ECGencode enables intuitive model configuration and provides some intrinsic interpretability of model parameters and decisions, as shown in Fig. 10 and discussed in Section 3.5. ECGencode also maintains the ECG structure within its 2D latent space representation which allows for intuitive complexity configuration and lends itself to be used in various model architectures. Minimal computational load for the feature extraction is guaranteed through the use of depthwise and depthwise separable convolutions in the compact ECGencode architecture. Additionally, ECGencode's novel Spatial Gaussian Noise regularisation technique enhances generalisation performance, positioning it as a versatile tool for various ECG analysis tasks without the trade-offs commonly associated with models optimised for computational efficiency.

These claims in favour of ECGencode are validated by incorporating it into two distinct ECGencode models. ECGencode model 1 is configured for a low parameter and FLOPs count while achieving performance on par with SOTA models for three distinct binary ECG classification tasks: detecting AFib-related patients, NSR AFib detection, and new-onset AFib prediction. The results from Section 4.3 demonstrate ECGencode power of matching or even outperforming the complex SOTA model by Attia, Noseworthy, et al. (2019) whilst significantly outperforming the computationally efficient AFibri-Net 3 model by Phukan et al. (2023). A computational efficiency analysis of the trainable parameters and FLOPs revealed a tenfold saving in FLOPs and over 20 times saving in parameters compared to the model by Attia, Noseworthy, et al. (2019). When compared to the AFibri-Net 3 model by Phukan et al. (2023), the trainable parameters remained significantly reduced, but the FLOPs count was higher. It was highlighted this is due to the contributions of the large kernels in the temporal filter of ECGencode and that it is possible to adjust this configuration to match the AFibri-Net 3 FLOPs count, although this could result in sub-par performance, like the AFibri-Net 3 model. A preliminary intrinsic interpretation of the temporal and spatial filters from ECGencode reveals they perform as expected, and common post hoc visualisation tools highlight the model predictions are based on areas of the ECG known to be representative of the task.

ECGencode model 2 showcases that a complex model incorporating ECGencode can be built through intuitive reasoning of the model configuration and 2D latent space representation of ECGencode. In particular, ECGencode model 2 is a CNN-LSTM model which performs on par with the benchmark for PTB-XL multi-label classification of 71 classes without requiring complex configuration or optimisation. Additionally, even with this complex LSTM functionality added, it still boasts a significant reduction in model parameters compared to the benchmark models.

6. Conclusion

This work introduced ECGencode, an innovative deep learning feature encoder designed for computationally efficient extraction of compact and informative feature vectors from raw ECG data. ECGencode tackles the challenges found in the complex, resource-intensive deep learning models prevalent in ECG signal analysis through various ECG-specific optimisations. First, ECGencode is an ECG-specific, expert-inspired and compact architecture, based on the FBCSP method. This, in combination with a novel Spatial Gaussian Noise layer for regularisation across both lead and channel dimensions, results in SOTA matching performance across various ECG analysis tasks. Secondly, ECGencode boasts an over tenfold reduction in FLOPs when compared to SOTA performing models in these tasks. This makes ECGencode models particularly suitable for inference on resource-constraint medical edge devices and provides them with favourable training behaviour. Thirdly, ECGencode's architectural parameters provide an intuitive relation with the ECG analysis task and its latent space retains the familiar 2D ECG structure. This enables easy configuration of the feature encoder for various ECG analysis model architectures. Lastly, as a compact architecture with minimal parameters based on the FBCSP approach, ECGencode lends itself to intrinsic learned parameter interpretations and integrates effectively with existing post hoc model visualisations. The temporal component of ECGencode can be visualised to confirm both high-frequency and low-frequency alternations are derived, while the learned kernel weights of the spatial component allow for a topographic visualisation of the input lead importance in generating augmented leads. Post hoc model visualisation techniques, such as gradient-based class activation maps and saliency maps, highlight the P-wave focus for positive AFib predictions. Although these preliminary visualisations are promising, additional medical research is required for comprehensive validation.

More specifically, in binary ECG classification tasks like AFib detection during NSR and pre-onset prediction, ECGencode model 1 efficiently operates with only 3.79% of the parameters and 12.39% of the FLOPs required by the SOTA model of Attia, Noseworthy, et al. (2019), while delivering comparable, or even improved, performance. For multi-label ECG classification of 71 diagnostic statements, the discussed ECGencode model 2 with added LSTM functionality matched the top PTB-XL benchmark models (Strodthoff et al., 2021) in performance while using less than a tenth of the parameters.

Whilst the experiments from this work validate the claimed benefits of ECGencode, and the efficient ECGencode model 1 can prove valuable as an AFib risk predictor and screening selection tool for use on medical edge devices, ECGencode can prove even more valuable in future work. One such research direction includes pre-training the feature encoder through self-supervised learning for learning general ECG features. This general feature encoder can then serve as a robust initialisation for tasks with such limited data that traditional training is not possible. More interesting studies can be done on the intrinsic model interpretation techniques ECGencode offers, by incorporating them into a live timeline scrubbing tool validating its relevance by medical clinicians, and potentially revealing new medical insights. Furthermore, ECGencode model 1 results, with the best AUC and sensitivity for all tasks, suggest it can be optimised to significantly beat SOTA performance in these or other tasks whilst continuing to enjoy its other benefits. One such optimisation may be the inclusion of known biomarkers and traditionally derived features in the latent space of ECGencode to create a hybrid feature encoder.

In conclusion, ECGencode's benefits make it a valuable addition to the ECG signal analysis domain, especially suited for deployment in environments with constrained computational resources or as an easy-to-configure feature encoder for benchmark models in new ECG analysis tasks. Given its versatility, efficiency and interpretable model architecture, it positions itself as a fundamental feature extraction tool in the toolbox of ECG analysis.

CRediT authorship contribution statement

Lennert Bontinck: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Visualization. **Karel Fonteyn:** Conceptualization, Methodology, Writing – review & editing. **Tom Dhaene:** Conceptualization, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Dirk Deschrijver:** Conceptualisation, Methodology, Formal analysis, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used OpenAI's ChatGPT (GPT-4) in order to improve readability and language. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

References

Abdullah, L. A., & Al-ani, M. S. (2020). CNN-LSTM Based Model for ECG Arrhythmias and Myocardial Infarction Classification. *Advances in Science, Technology and Engineering Systems Journal*, *5*(5), 601–606.

Alamatsaz, N., Tabatabaei, L., Yazdchi, M., Payan, H., Alamatsaz, N., & Nasimi, F. (2024). A lightweight hybrid CNN-LSTM explainable model for ECG-based arrhythmia detection. *Biomedical Signal Processing and Control*, *90*, Article 105884.

Alfaras, M., Soriano, M. C., & Ortín, S. (2019). A fast machine learning model for ECG-based heartbeat classification and arrhythmia detection. *Frontiers in Physics*, *7*.

Alonso, A., Krijthe, B. P., Aspelund, T., Stepas, K. A., Pencina, M. J., Moser, C. B., Sinner, M. F., Sotoodehnia, N., Fontes, J. D., Janssens, A. C. J. W., Kronmal, R. A., Magnani, J. W., Witteman, J. C., Chamberlain, A. M., Lubitz, S. A., Schnabel, R. B., Agarwal, S. K., McManus, D. D., Ellinor, P. T., ... Benjamin, E. J. (2013). Simple risk model predicts incidence of atrial fibrillation in a racially and geographically diverse population: the charge&#x2010;af consortium. *Journal of the American Heart Association*, *2*(2), Article e000102.

Ang, K. K., Chin, Z. Y., Zhang, H., & Guan, C. (2008). Filter bank common spatial pattern (FBCSP) in brain-computer interface. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)* (pp. 2390–2397).

Attia, Z. I., Kapa, S., Lopez-Jimenez, F., McKie, P. M., Ladewig, D. J., Satam, G., Pellikka, P. A., Enriquez-Sarano, M., Noseworthy, P. A., Munger, T. M., Asirvatham, S. J., Scott, C. G., Carter, R. E., & Friedman, P. A. (2019). Screening for cardiac contractile dysfunction using an artificial intelligence–enabled electrocardiogram. *Nature Medicine*, *25*(1), 70–74.

Attia, Z. I., Noseworthy, P. A., Lopez-Jimenez, F., Asirvatham, S. J., Deshmukh, A. J., Gersh, B. J., Carter, R. E., Yao, X., Rabinstein, A. A., Erickson, B. J., Kapa, S., & Friedman, P. A. (2019). An artificial intelligence-enabled ECG algorithm for the identification of patients with atrial fibrillation during sinus rhythm: A retrospective analysis of outcome prediction. *The Lancet*, *394*(10201), 861–867.

Ayano, Y. M., Schwenker, F., Dufera, B. D., & Debelee, T. G. (2023). Interpretable machine learning techniques in ECG-based heart disease classification: A systematic review. *Diagnostics*, *13*(1).

Bozyigit, F., Erdemir, F., Sahin, M., & Kilinc, D. (2020). Classification of electrocardiogram (ECG) data using deep learning methods. In *2020 4th international symposium on multidisciplinary studies and innovative technologies* (pp. 1–5).

Buber, E., & Diri, B. (2018). Performance analysis and CPU vs GPU comparison for deep learning. In *2018 6th international conference on control engineering & information technology* (pp. 1–6).

Cai, W., Chen, Y., Guo, J., Han, B., Shi, Y., Ji, L., Wang, J., Zhang, G., & Luo, J. (2020). Accurate detection of atrial fibrillation from 12-lead ECG using deep neural network. *Computers in Biology and Medicine*, *116*, Article 103378.

Chattopadhay, A., Sarkar, A., Howlader, P., & Balasubramanian, V. N. (2018). Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE winter conference on applications of computer vision* (pp. 839–847).

Christ, M., Braun, N., Neuffer, J., & Kempa-Liehr, A. W. (2018). Time series feature extraction on basis of scalable hypothesis tests (tsfresh – A Python package). *Neurocomputing*, *307*, 72–77.

Christopoulos, G., Graff-Radford, J., Lopez, C. L., Yao, X., Attia, Z. I., Rabinstein, A. A., Petersen, R. C., Knopman, D. S., Mielke, M. M., Kremers, W., Vemuri, P., Siontis, K. C., Friedman, P. A., & Noseworthy, P. A. (2020). Artificial intelligence–electrocardiography to predict incident atrial fibrillation. *Circulation: Arrhythmia and Electrophysiology*, *13*(12), Article e009355.

Clevert, D., Unterthiner, T., & Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (ELUs). In Y. Bengio, & Y. LeCun (Eds.), *4th international conference on learning representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, conference track proceedings*.

Del Pup, F., & Atzori, M. (2023). Applications of self-supervised learning to biomedical signals: Where are we now. *Authorea Preprints*.

Dubatovka, A., & Buhmann, J. M. (2022). Automatic detection of atrial fibrillation from single-lead ECG using deep learning of the cardiac cycle. *BME Frontiers*, *2022*, Article 9813062.

Faruk, N., Abdulkarim, A., Emmanuel, I., Folawiyo, Y. Y., Adewole, K. S., Mojeed, H. A., Oloyede, A. A., Olawoyin, L. A., Sikiru, I. A., Nehemiah, M., Ya'u Gital, A., Chiroma, H., Ogunmodede, J. A., Almutairi, M., & Katibi, I. A. (2021). A comprehensive survey on low-cost ECG acquisition systems: Advances on design specifications, challenges and future direction. *Biocybernetics and Biomedical Engineering*, *41*(2), 474–502.

Gedon, D., Ribeiro, A. H., Wahlström, N., & Schön, T. B. (2021). First steps towards self-supervised pretraining of the 12-lead ECG. In *2021 computing in cardiology (CinC): vol. 48*, (pp. 1–4).

Gilon, C., Grégoire, J.-M., Mathieu, M., Carlier, S., & Bersini, H. (2023). IRIDIA-AF, a large paroxysmal atrial fibrillation long-term electrocardiogram monitoring database. *Scientific Data*, *10*(1), 714.

Goldberger, A. L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet. *Circulation*, *101*(23), e215–e220.

Goodacre, S., & Irons, R. (2002). ABC of clinical electrocardiography: Atrial arrhythmias. *BMJ*, *324*(7337), 594–597.

Gruwez, H., Barthels, M., Haemers, P., Verbrugge, F. H., Dhont, S., Meekers, E., Wouters, F., Nuyens, D., Pison, L., Vandervoort, P., & Pierlet, N. (2023). Detecting paroxysmal atrial fibrillation from an electrocardiogram in sinus rhythm: External validation of the AI approach. *JACC: Clinical Electrophysiology*, *9*(8, Part 3), 1771–1782.

Gu, M., Zhang, Y., Wen, Y., Ai, G., Zhang, H., Wang, P., & Wang, G. (2023). A lightweight convolutional neural network hardware implementation for wearable heart rate anomaly detection. *Computers in Biology and Medicine*, *155*, Article 106623.

Gyawali, D. (2023). Comparative analysis of CPU and GPU profiling for deep learning models. ArXiv arXiv:2309.02521.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hicks, S. A., Isaksen, J. L., Thambawita, V., Ghouse, J., Ahlberg, G., Linneberg, A., Grarup, N., Strümke, I., Ellervik, C., Olesen, M. S., Hansen, T., Graff, C., Holstein-Rathlou, N.-H., Halvorsen, P., Maleckar, M. M., Riegler, M. A., & Kanters, J. K. (2021). Explaining deep neural networks for knowledge discovery in electrocardiogram analysis. *Scientific Reports*, *11*(1), 10949.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE conference on computer vision and pattern recognition* (pp. 2261–2269).

Huang, W., Xue, Y., Hu, L., & Liuli, H. (2020). S-EEGNet: Electroencephalogram signal classification based on a separable convolution neural network with bilinear interpolation. *IEEE Access*, *8*, 131636–131646.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. Bach, & D. Blei (Eds.), *Proceedings of machine learning research*: *vol. 37*, *Proceedings of the 32nd international conference on machine learning* (pp. 448–456). Lille, France: PMLR.

Izenman, A. J. (2008). Linear discriminant analysis. In *Modern multivariate statistical techniques: regression, classification, and manifold learning* (pp. 237–280). New York, NY: Springer New York.

Jahmunah, V., Ng, E., Tan, R.-S., Oh, S. L., & Acharya, U. R. (2022). Explainable detection of myocardial infarction using deep learning models with grad-CAM technique on ECG signals. *Computers in Biology and Medicine*, *146*, Article 105550.

Jang, J.-H., Kim, T. Y., Lim, H.-S., & Yoon, D. (2021). Unsupervised feature learning for electrocardiogram data using the convolutional variational autoencoder. *PLoS One*, *16*(12), 1–16.

Jaworski, M., Duraj, A., & Szczepaniak, P. (2022). Evaluation of deep machine learning methods for analysis of ECG stream data. *Procedia Computer Science*, *207*, 1212–1221.

Jiang, P.-T., Zhang, C.-B., Hou, Q., Cheng, M.-M., & Wei, Y. (2021). LayerCAM: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, *30*, 5875–5888.

Khan, F., Yu, X., Yuan, Z., & Rehman, A. u. (2023). ECG classification using 1-D convolutional deep residual neural network. *PLoS One*, *18*(4), 1–22.

Kim, J.-K., Jung, S., Park, J., & Han, S. W. (2022). Arrhythmia detection model using modified DenseNet for comprehensible grad-CAM visualization. *Biomedical Signal Processing and Control*, *73*, Article 103408.

Koles, Z. J., Lazar, M. S., & Zhou, S. Z. (1990). Spatial patterns underlying population differences in the background EEG. *Brain Topogr.*, *2*(4), 275–284.

Kuznetsov, V. V., Moskalenko, V. A., Gribanov, D. V., & Zolotykh, N. Y. (2021). Interpretable feature generation in ECG using a variational autoencoder. *Frontiers in Genetics*, *12*.

Lai, C., Zhou, S., & Trayanova, N. A. (2021). Optimal ECG-lead selection increases generalizability of deep learning on ECG abnormality classification. *Philosophical Transactions of the Royal Society of London A (Mathematical and Physical Sciences)*, *379*(2212), Article 20200258, arXiv:https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2020.0258.

Lawhern, V. J., Solon, A. J., Waytowich, N. R., Gordon, S. M., Hung, C. P., & Lance, B. J. (2018). EEGNet: A compact convolutional neural network for EEG-based brain–computer interfaces. *Journal of Neural Engineering*, *15*(5), Article 056013.

Liao, L., Li, H., Shang, W., & Ma, L. (2022). An empirical study of the impact of hyperparameter tuning and model optimization on the performance properties of deep neural networks. *ACM Transactions on Software Engineering and Methodology*, *31*(3).

Lima, E. M., Ribeiro, A. H., Paixão, G. M., Ribeiro, M. H., Filho, M. M. P., Gomes, P. R., Oliveira, D. M., Sabino, E. C., Duncan, B. B., Giatti, L., Barreto, S. M., Meira, W., Schön, T. B., & Ribeiro, A. L. P. (2021). Deep neural network estimated electrocardiographic-age as a mortality predictor. *medRxiv*.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2020). Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(2), 318–327.

Liu, H., Zhao, Z., & She, Q. (2021). Self-supervised ECG pre-training. *Biomedical Signal Processing and Control*, *70*, Article 103010.

Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *International conference on learning representations*.

Luo, P., Wang, X., Shao, W., & Peng, Z. (2019). Towards understanding regularization in batch normalization. In *International conference on learning representations*.

Macfarlane, P. W., & Kennedy, J. (2021). Automated ECG interpretation—A brief history from high expectations to deepest networks. *Hearts*, *2*(4), 433–448.

Mincholé, A., Camps, J., Lyon, A., & Rodríguez, B. (2019). Machine learning in the electrocardiogram. *Journal of Electrocardiology*, *57*, S61–S64.

Musa, N., Gital, A. Y., Aljojo, N., Chiroma, H., Adewole, K. S., Mojeed, H. A., Faruk, N., Abdulkarim, A., Emmanuel, I., Folawiyo, Y. Y., Ogunmodede, J. A., Oloyede, A. A., Olawoyin, L. A., Sikiru, I. A., & Katb, I. (2023). A systematic review and meta-data analysis on the applications of deep learning in electrocardiogram. *Journal of Ambient Intelligence and Humanized Computing*, *14*(7), 9677–9750.

Noseworthy, P. A., Attia, Z. I., Brewer, L. C., Hayes, S. N., Yao, X., Kapa, S., Friedman, P. A., & Lopez-Jimenez, F. (2020). Assessing and mitigating bias in medical artificial intelligence. *Circulation: Arrhythmia and Electrophysiology*, *13*(3), Article e007988.

Petmezas, G., Stefanopoulos, L., Kilintzis, V., Tzavelis, A., Rogers, J. A., Katsaggelos, A. K., & Maglaveras, N. (2022). State-of-the-art deep learning methods on electrocardiogram data: Systematic review. *JMIR Medical Informatics*, *10*(8), Article e38454.

Phukan, N., Manikandan, M. S., & Pachori, R. B. (2023). Afibri-net: A lightweight convolution neural network based atrial fibrillation detector. *IEEE Transactions on Circuits and Systems. I. Regular Papers*, *70*(12), 4962–4974.

Qin, Y., Sun, L., Chen, H., Yang, W., Zhang, W.-Q., Fei, J., & Wang, G. (2023). MVKT-ECG: Efficient single-lead ECG classification for multi-label arrhythmia by multi-view knowledge transferring. *Computers in Biology and Medicine*, *166*, Article 107503.

Raghunath, S., Pfeifer, J. M., Ulloa-Cerna, A. E., Nemani, A., Carbonati, T., Jing, L., vanMaanen, D. P., Hartzel, D. N., Ruhl, J. A., Lagerman, B. F., Rocha, D. B., Stoudt, N. J., Schneider, G., Johnson, K. W., Zimmerman, N., Leader, J. B., Kirchner, H. L., Griessenauer, C. J., Hafez, A., ... Haggerty, C. M. (2021). Deep neural networks can predict new-onset atrial fibrillation from the 12-lead ECG and help identify those at risk of atrial fibrillation–related stroke. *Circulation*, *143*(13), 1287–1298.

Ribeiro, A. H., Paixao, G. M., Lima, E. M., Horta Ribeiro, M., Pinto Filho, M. M., Gomes, P. R., Oliveira, D. M., Meira, W., Jr., Schon, T. B., & Ribeiro, A. L. P. (2021). CODE-15%: A large scale annotated dataset of 12-lead ECGs.

Ribeiro, A. H., Ribeiro, M. H., Paixão, G. M. M., Oliveira, D. M., Gomes, P. R., Canazart, J. A., Ferreira, M. P. S., Andersson, C. R., Macfarlane, P. W., Meira Jr., W., Schön, T. B., & Ribeiro, A. L. P. (2020). Automatic diagnosis of the 12-lead ECG using a deep neural network. *Nature Communications*, *11*(1), 1760.

Riyad, M., Khalil, M., & Adib, A. (2020). MI-EEGNET: A novel convolutional neural network for motor imagery classification. *Journal of Neuroscience Methods*, *353*, Article 109037.

Romdhane, T. F., Alhichri, H., Ouni, R., & Atri, M. (2020). Electrocardiogram heartbeat classification based on a deep convolutional neural network and focal loss. *Computers in Biology and Medicine*, *123*, Article 103866.

Roots, K., Muhammad, Y., & Muhammad, N. (2020). Fusion convolutional neural network for cross-subject EEG motor imagery classification. *Computers*, *9*(3).

Sakr, A. S., Pławiak, P., Tadeusiewicz, R., Pławiak, J., Sakr, M., & Hammad, M. (2023). ECG-COVID: An end-to-end deep model based on electrocardiogram for COVID-19 detection. *Information Sciences*, *619*, 324–339.

Sau, A., & Ng, F. S. (2023). –The emerging role of artificial intelligence enabled electrocardiograms in healthcare. *BMJ Medicine*, *2*(1).

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2020). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, *128*(2), 336–359.

Sepahvand, M., & Abdali-Mohammadi, F. (2022). A novel method for reducing arrhythmia classification from 12-lead ECG signals to single-lead ECG with minimal loss of accuracy through teacher-student knowledge distillation. *Information Sciences*, *593*, 64–77.

Sharir, O., Peleg, B., & Shoham, Y. (2020). The cost of training NLP models: A concise overview. ArXiv arXiv:2004.08900.

Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at international conference on learning representations*.

Smilkov, D., Thorat, N., Kim, B., Viégas, F. B., & Wattenberg, M. (2017). SmoothGrad: Removing noise by adding noise. CoRR abs/1706.03825.

Somani, S., Russak, A. J., Richter, F., Zhao, S., Vaid, A., Chaudhry, F., De Freitas, J. K., Naik, N., Miotto, R., Nadkarni, G. N., Narula, J., Argulian, E., & Glicksberg, B. S. (2021). Deep learning and the electrocardiogram: review of the current state-of-the-art. *EP Europace*, *23*(8), 1179–1191.

Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. A. (2015). Striving for simplicity: The all convolutional net. In Y. Bengio, & Y. LeCun (Eds.), *3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, workshop track proceedings*.

Strodthoff, N., Wagner, P., Schaeffter, T., & Samek, W. (2021). Deep learning for ECG analysis: Benchmarks and insights from PTB-XL. *IEEE Journal of Biomedical and Health Informatics*, *25*(5), 1519–1528.

Tohyama, T., Ide, T., Ikeda, M., Nagata, T., Tagawa, K., Hirose, M., Funakoshi, K., Sakamoto, K., Kishimoto, J., Todaka, K., Nakashima, N., & Tsutsui, H. (2023). Deep learning of ECG for the prediction of postoperative atrial fibrillation. *Circulation: Arrhythmia and Electrophysiology*, *16*(2), Article e011579.

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., & Bregler, C. (2015). Efficient object localization using convolutional networks. In *2015 IEEE conference on computer vision and pattern recognition* (pp. 648–656).

Wagner, P., Strodthoff, N., Bousseljot, R.-D., Kreiseler, D., Lunze, F. I., Samek, W., & Schaeffter, T. (2020). PTB-XL, a large publicly available electrocardiography dataset. *Scientific Data*, *7*(1), 154.

Wagner, P., Strodthoff, N., Bousseljot, R.-D., Samek, W., & Schaeffter, T. (2022). PTB-XL, a large publicly available electrocardiography dataset.

Wang, M. (2023). A modified motor imagery classification method based on EEGNet. In *Proceedings of the 2022 6th international conference on electronic information technology and computer engineering* (pp. 427–431). New York, NY, USA: Association for Computing Machinery.

Wang, H., Wang, Z., Du, M., Yang, F., Zhang, Z., Ding, S., Mardziel, P., & Hu, X. (2020). Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *2020 IEEE/CVF conference on computer vision and pattern recognition workshops* (pp. 111–119). Los Alamitos, CA, USA: IEEE Computer Society.

Xiaolin, L., Panicker, R. C., Cardiff, B., & John, D. (2021). Multistage pruning of CNN based ECG classifiers for edge devices. In *2021 43rd annual international conference of the IEEE engineering in medicine & biology society* (pp. 1965–1968). IEEE.

Xu, W., & Du, S. S. (2023). Over-parameterization exponentially slows down gradient descent for learning a single neuron. ArXiv arXiv:2302.10034.

Zhang, H., Wang, Z., Yu, Y., Yin, H., Chen, C., & Wang, H. (2022). An improved EEGNet for single-trial EEG classification in rapid serial visual presentation task. *Brain Science Advances*, *8*(2), 111–126.