The Scientific Temper (2025) Vol. 16 (11): 5006-5013

E-ISSN: 2231-6396, ISSN: 0976-8653

Doi: 10.58414/SCIENTIFICTEMPER.2025.16.11.04

https://scientifictemper.com/

RESEARCH ARTICLE

Performance Analysis of Deep Learning Optimizers for Arrhythmia Classification using PTB-XL ECG Dataset: Emphasis on Adam Optimizer

S Selvakumari[1*], M Durairaj[2]

Abstract

This study assesses the performance of several deep learning optimizers for arrhythmia classification via the PTB-XL ECG dataset. Deep learning (DL) based approaches, such as convolutional neural network (CNN) and recurrent neural network (RNN) have demonstrated promising results in learning discriminative feature representations of ECGs for automatic cardiac diagnosis. A CNN-LSTM based model was trained through six optimizers namely SGD, Momentum, Adagrad, Adadelta, RMSprop and Adam. The PTB-XL dataset with more than 20,000 12-lead ECGs was utilized for the classification performance comparison. Interest centred toward the Adam performance, which implies the adaptive moment estimated gradients and sets different learning rates for each learning rate. The Adam optimizer outperformed all other tested optimizers with 98.26% accuracy, 96.15% sensitivity, 97.43% specificity, 96.78% precision, and 96.46% F1-score. In comparison to other optimizers, they obtained low performance measures and reached convergence slowly. These results reveal the advantage of Adam in terms of training stability and predictive confidence for ECG-based arrhythmia classification. This research is one of the very few to systematically analyse various optimizers on PTB-XL dataset with hybrid architecture (CNN and LSTM). The experimental results confirm the superiority of Adam in ECG signal classification and provide a strong baseline for more effective deep learning model used in (cardiac) arrhythmia detection and clinical deep learning systems.

Keywords: Deep Learning, Artificial Intelligence, Adam, Deep Learning Architecture, Activation functions, Arrhythmia.

关键词：深度学习，人工智能，Adam，深度学习架构，激活函数，心律失常

Introduction

DL is a subfield of ML grounded in Artificial Neural Networks (ANN). Because neural networks aim to emulate humanoid intelligence, DL likewise functions as an imitation of human intelligence. Unlike traditional programming, not everything is explicitly coded; instead, models learn from data. The idea itself is not new—its rise in popularity reflects the recent availability of far more data and significantly greater computing power over the past decades. Over roughly twenty years, dramatic gains in processing capability have accelerated advances in both ML and DL.

DL learns layered representations of the world: concepts are organized hierarchically, with higher-level, more abstract notions defined in terms of simpler ones (Deekshith Shetty, Harshavardhan C A, M Jayanth Varma, Shrishail Navi, Mohammed Riyaz Ahmed,2020). This layered approach enables highly flexible and powerful learning. The human brain has on the order of 100 billion neurons; each connected to thousands of others. In DL, multiple layers of algorithms process data, simulate cognitive operations, and build abstractions (Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie & Laith Farhan, 2021). A network begins with an input layer and ends with an output layer; the layers in between are hidden layers. Each layer applies a basic, unified computation using an activation function. Feature extraction is part of

[1]Research Scholar, Department of Computer Science, Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.

[2]Associate Professor, Department of Computer Science, Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.

***Corresponding Author:** S Selvakumari, Research Scholar, Department of Computer Science, Bharathidasan University, Tiruchirappalli, Tamil Nadu, India, E-Mail: selvakumarisivashanmugam@gmail.com

How to cite this article: Selvakumari, S., Durairaj, M. (2025). Performance Analysis of Deep Learning Optimizers for Arrhythmia Classification using PTB-XL ECG Dataset: Emphasis on Adam Optimizer. The Scientific Temper, **16**(11):5006-5013.

Doi: 10.58414/SCIENTIFICTEMPER.2025.16.11.04

Source of support: Nil

Conflict of interest: None.

Received: 22/10/2025 **Accepted:** 14/11/2025 **Published:** 22/11/2025

this process. DL is used to detect patterns and interpret visual information, employing feature extraction to form meaningful lines from data for learning, understanding, and retention lines that are typically specified by data scientists or programmers (Iqbal H. Sarker, 2021).

The Electrocardiogram (ECG) is a crucial, non-invasive method for diagnosing cardiac arrhythmias, which account for a large share of morbidity and mortality linked to cardiovascular disease. While manual ECG review can be effective, it is labour-intensive and relies heavily on expert judgment. Against this backdrop, deep learning has become a powerful way to automate ECG interpretation by learning layered representations directly from raw signals. Models such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks have shown strong capability in capturing spatial and temporal patterns in biomedical data. In particular, hybrid CNN-LSTM architectures are well suited to time-series like ECG, combining localized feature extraction with sequential modelling (M. Durairaj and S. Selvakumari, 2025).

While architectural innovations have been extensively explored in the field of ECG classification, the impact of optimization algorithms on model performance has received considerably less attention. Most studies in this domain either default to a single optimizer (typically Adam or SGD) or fail to justify the choice of optimization strategy altogether. This presents a critical gap in research, as the choice of optimizer directly influences convergence speed, generalization ability, and classification accuracy, especially for noisy, high-dimensional biomedical signals like ECGs.

Moreover, although the PTB-XL dataset is one of the largest and most comprehensive open-access 12-lead ECG datasets, comparative studies on different optimizers using this dataset are scarce. Existing works often overlook the importance of clinically relevant metrics such as sensitivity and specificity, which are crucial in avoiding misdiagnosis in real-world healthcare applications. Additionally, there is a lack of empirical studies that evaluate multiple optimizers on a fixed hybrid DL architecture like CNN-LSTM, which combines the strengths of both convolutional and recurrent layers for improved arrhythmia detection.

To address these gaps, this study conducts a comprehensive evaluation of six popular deep learning optimizers, SGD, Momentum, Adagrad, Adadelta, RMSprop, and Adam, on the task of arrhythmia classification using the PTB-XL dataset. The proposed approach employs a CNN-LSTM hybrid model trained under consistent conditions to ensure fair comparison. Emphasis is placed on the Adam optimizer, given its adaptive nature and wide use in deep learning tasks. The study further evaluates performance using five clinically relevant metrics: Accuracy, Sensitivity, Specificity, Precision, and F1-Score.

The goal is not only to determine the most effective optimizer for this particular application but also to provide insights that can guide future research and real-world implementation of DL-based ECG diagnostic systems.

Methodology

Architecture

In this work we investigate several types of DL models known to be effective in biomedical signal processing, and especially in ECG-derived arrhythmia classification. The fundamental parts of the architecture consist of Deep Neural Networks (DNN), Deep Belief Networks (DBN), Recurrent Neural Networks (RNN), and CNN (Mohammad-Parsa Hosseini, Senbao Lu, Kavin Kamaraj, Alexander Slowikowski and Haygreev C. Venkatesh, 2020). Finally, we used a hybrid approach combining CNN and LSTM because a combination of both spatial and temporal feature extraction can be achieved.

Deep Neural Network

A DNN is a feedforward neural networks with many hidden layers sandwiched between input and output layers. Such networks are able to model non-linear dynamics on high-dimensional data. DNN, particularly in the field of ECG analysis, have great capability of capturing the hierarchical structure of waveform features such as P-waves, QRS complexes, and T-waves.

Deep Belief Network

DBNs are generative graphical models with multiple layers of Restricted Boltzmann Machines (RBMs). They are pre-trained by unsupervised learning with the Contrastive Divergence algorithm in order to learn hierarchical representation with input data. A DBN can be fine-tuned by supervision methods, which are well suited for applications like classification and anomaly detection in medical signals.

Convolutional Neural Network

CNNs are specifically for processing data that is structured as a grid, e.g., time series or images. In ECG analysis, 1D CNNs capture both local patterns and spatial relationship of the ECG including, morphology transformation in parts of waveform. Convolutional layers use filters on the ECG sequence in order to capture local information, while pooling layers reduce the dimensionality and improve the translation invariance. CNNs work well as feature extractors, especially with LSTM layers for temporal context modelling.

Recurrent Neural Network

As they are intended for sequential data, RNNs are appropriate for time series data, including ECG signals. Due to their feedback loop, they are provided a memory of the previous inputs, which is crucial for capturing long-range dependencies. However traditional RNNs suffers from vanishing gradients which makes it difficult to capture long-range dependencies.

To address this issue, the use of LSTM units is proposed. LSTMs can learn long-term dependencies with gating mechanisms which are special type of RNN. In this work, due to the extraction process of spatial features of ECG signals, LSTM layers are positioned behind the convolutional layers to learn the tempo-spatial features.

Hybrid CNN-LSTM Architecture

The model is a combined architecture employing both CNN and LSTM. Therefore, the CNN layers are regarded as the spatial ECG waveform feature ex-tractor, capturing the local signal structures. These features are in turn fed to LSTM layers, which capture the temporal structure within the signal. The hybrid network thus integrates the two into a whole and achieves better performance on arrhythmia classification problem.

Activation Functions

The Activation Function (AF) in a neural network set how each node turns the weighted sum of its inputs into an output. Activation functions are sometimes referred to as "flow functions," and when their output range is limited, they may be called "inhibit functions" (Shiv Ram Dubey, Satish Kumar Singh, Bidyut Baran Chaudhuri, 2022). Many activation functions are non-linear, and this non-linearity is a key consideration in layer and network design. Choosing the activation function strongly influences how the network operates and performs, and different parts of a model may use different ones. In DL, activation functions are a central element of neural networks. Tasks such as image categorization, language conversion, object detection, and related applications rely on neural networks—and thus on their activation functions.

In general, the neural network activation function is the most important component of deep learning, which determines the performance of the learning model, the results of the deep learning model, accuracy, and the ability to design or divide the network on a large scale. Neural scales are mostly used in networks. An activation function creates an output for an input or group of inputs, or a node for its output given an input.

Basically, functions decide to turn neurons on or off to get the desired result. It also performs non-linear input transformations to improve the results of complex neural networks. The enable function also helps normalize the output for all inputs between 1 and -1.

AFs need to be computationally efficient, since neural networks may be trained on millions of samples and thus require fast evaluation (Siddharth Sharma, Simone Sharma & Anidhya Athaiya, 2020). They effectively decide whether the information entering a network is relevant. To illustrate how a neuron works and how an AF constrains its output, consider this: a neuron computes a weighted combination of its inputs and then applies an AF to produce the final output.

$$Y = \sum\left(weights \times input + bias\right)\#(1)$$

where Y can be any neuron ranging from -infinity to +infinity. To achieve the required prediction or generalization outcomes and limit the output.

$$Y = AF\left(\sum\left(weights \times input + bias\right)\right)\#(2)$$

To bind the output value, pass the matching neuron to the AF.

The AF is one of the options available when designing a neural network. It can be employed not only in the network's output layer, but also in its hidden layers.

Neural Network Components

Input layer

Receives the feature inputs and passes information from the external world into the network. This stage does not perform computations; its nodes forward signals (functions) onward to the hidden hierarchy.

Hidden layer

Composed of nodes not directly observable from outside the model, forming the network's abstraction. These layers carry out multiple computations on the signals from the input layer and then pass the processed results to the output.

Output layer

This layer introduces the information received by the network to the outside world.

The AF takes the weighted sum of the inputs, adds a bias, and then decides whether the neuron should activate. The AFs objective is to add nonlinearity to the neuron's output signal. A neural network is made up of neurons that function based on weights, biases, and activation functions. Based on inference errors, a neural network fine-tunes the weights and biases of its neurons. This procedure is known as backpropagation. The trigger function enables for backpropagation by feeding gradients and errors to update weights and biases.

Variants of Activation Function

RELU

It denotes a rectified linear device. The most often used activation function. This is mostly implemented in the neural network's hidden layers. The Figure 1 shows the diagrammatic representation of RELU.

- Equation: $A(x) = max(0, x)$. It outputs x when $x > 0$; otherwise, it outputs 0.
- Range: $[0, \infty]$
- Nature: Non-linear, enabling efficient error propagation and the activation of many stacked layers when using ReLu.

Usage: - ReLu uses simpler math operations than tanh and

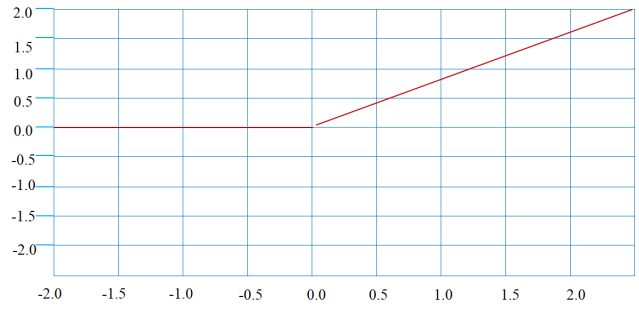

Figure 1: Rectified linear unit

sigmoid, hence it takes less processing. Only a few neurons are active at once, making the network simple, efficient, and straightforward to compute.

Softmax function

It's a sigmoid function that is effective for dealing with classification challenges. The Figure 2 shows the diagrammatic representation of Softmax function.

- Natural: - Nonlinear
- Typically used for managing many classes. The softmax function compresses each class's output between 0 and 1 and divides it by the overall output.

Learning Optimization Algorithm

DL models comprise several components, activation functions, input and output layers, hidden layers, loss functions, and more. A unifying feature across these models is their use of algorithms that generalize from data to make predictions on previously unseen samples. In essence, require a method that maps inputs to outputs, along with an optimization routine. The optimizer adjusts the model's parameters (weights) to minimize the errors made when transforming inputs into outputs. These optimizers affect the sensibility of the deep learning model greatly. When training the deep learning model, we have to tune the weights every time, and we need to minimize the loss function. An optimizer is a process or a mechanism that changes the properties of a NNs, i.e. their weights and learning rates. Therefore, it is reducing total loss along with improving fineness. As the size of the network grows, the weights of the filters become increasingly difficult to determine. This also highlights the necessity of choosing

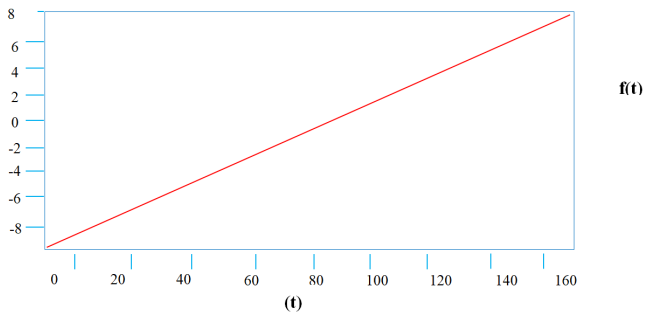

Figure 2: Softmax Function

the appropriate optimization approach for the application. Therefore, understanding these algorithms requires deep study in this domain.

- The epoch is the appearance frequency of the full data on the algorithm.
- A sample refers to a record in the dataset.
- Batch - The number of samples to be used for rationalising model parameters.
- Learning Rate - The parameter that decides at each iteration the step size and moving towards a minimum of a loss function.
- The difference between the estimated result and real value, called cost will be calculated using the cost function / loss function.
- Weights/bias are parameters of a model which are adapted and control the signal between any two neurons.

Implementation

The Figure 3 shows the process applied in this research to classify arrhythmia with "PTB-XL ECG dataset". We first collect ECG data, then conduct necessary signal processing for normalization and preparation of acquired ECGs that is required by deep learning. In the next architecture, we further use a hybrid CNN-LSTM model that considers not only spatial but also temporal features of ECG data. The model is trained and tested with SGD, Momentum, Adam, AdaGrad, and AdaDelta for finding their performance on improving the classification accuracy. Finally, performance analysis is performed to compare the optimizers in several metrics: It shows that Adam has superiority in robustness and reliability for our application.

Dataset

The PTB-XL comprises multilead ECG signals labeled as normal or arrhythmic. The signals were resampled, normalized, and divided into test sets, training, and validation in an 80:20 split.

Network architecture

The deep learning architecture we presented here integrates Convolutional and Recurrent components to perform well in spatial and temporal information extraction from ECG signals suitable for arrhythmia classifications.

Convolutional Layers

The model begins with two 1D convolutional layers. The first applies 32 filters with a kernel size of 7 and a stride of 2, followed by a ReLU activation to add non-linearity. Afterward, a max-pooling layer with pool size 2 reduces the feature dimensions. The second convolutional layer contains 64 filters of kernel size 5 with a stride of 2 followed by ReLU activation and max pooling. This set of convolutional layers can learn localized pattern and morphological structure in ECG signals.

Table 1: Summary of Optimizers Used in This Study

Reference	*Optimizer*	*Description*
Carmina Fjellstrom and Kaj Nystrom, 2022	SGD(Stochastic Gradient Descent)	A simple optimizer that objects to model parameters in an individual training sample. It is simple and memory-efficient, but sometimes slow to converge and susceptible to noisy gradients, making it prone to unstable training, particularly in challenging and noisy biomedical data, such as ECG heartbeat signals.
Aadil Gani Ganie and Samad Dadvandipour, 2023	Momentum	A generalization of SGD that expands the gradient-based end-of-step estimator to include a 'velocity' term that integrates over past gradients. This facilitates faster convergence and removes oscillations. Momentum helps optimization over rugged landscapes, but is not adaptive to changes in the scale of the gradient.
Liu Yang and Deng Cai, 2021	Adagrad(Adaptive Gradient Algorithm)	Scales the learning rate of each parameter by the magnitude of its past gradient. It is effective for rare data but has serious issue of fast decaying of the learning rate, which may cause to stop learning prematurely in very long training runs and degrade its performance on ECG data.
Liu Yang and Deng Cai, 2021	Adadelta	It has a to restrict the accumulation of the squared gradients and a does not require an explicit learning rate. It does better with noisy data but doesn't have the bias correction and robustness of some of the more evolved optimizers out there, such as Adam.
Dongpo Xu, Shengdong Zhang, Huisheng Zhang and Danilo P. Mandic, 2021	RMSprop(Root Mean Square Propagation)	Sustains an average running of the squared gradient to dynamically adjust the learning rate. Especially for sequential and time series data such as ECGs, the RMSprop enables a stable and rapid convergence equivalent to Adam across the experimentation.
S. Selvakumari and M. Durairaj, 2025 Mohamed Reyad, Amany M. Sarhan, and M. Arafa, 2023	Adam(Adaptive Moment Estimation)	Momentum and RMSprop - Calculating first and second moment estimate of gradient. It adjusts the learning rate for each parameter, and introduces bias correction, which leads to better convergence and generalization. Adam was the top performer in all @-metrics in this study.

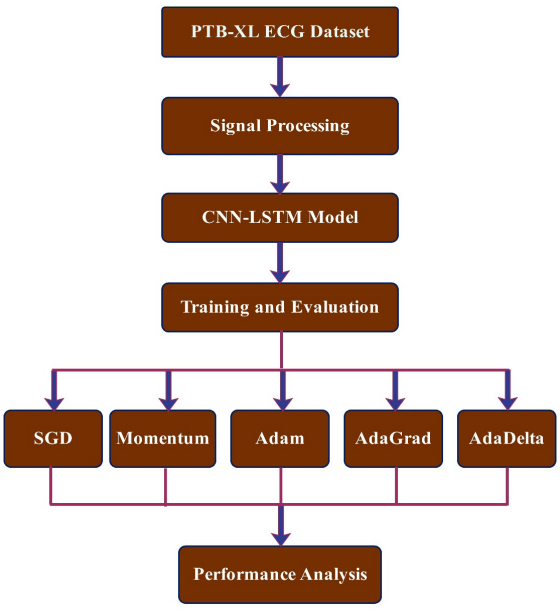

Figure 3: Performance Analysis

LSTM Layer

After feature extraction, an LSTM layer with 100 units is utilized to capture the temporal dependencies existing among ECG sequences. We incorporate a dropout ratio of 0.5 to avoid overfitting and spectrum generalization performance.

Fully Connected Output Layer

The last layer of the model is a dense, fully connected layer which uses softmax as an AF to produce probabilities for two classes, Normal and Arrhythmia.

Activation and Loss Function

The convolutional layers use ReLU to help them train through non-linearity, whereas the softmax ensures a probabilistic readout at the output layer. The network is optimized for binary classification problems with binary cross-entropy loss.

Optimizers and Training Setup

For the optimal optimization algorithm to use, 6 optimizers are used SGD, Momentum, Adagrad Adadelta, RMSProp and Adam and each of them is trained with the model. The initial learning rate for all optimizers is set to 0.001, the batch size is set to 32, and training is performed within 50 epochs.

Result And Discussion

This section delivers a detailed comparison of six commonly used deep learning optimizers for arrhythmia classification on the PTB-XL ECG dataset. The methods assessed are Stochastic Gradient Descent (SGD), Momentum, Adagrad, Adadelta, RMSprop, and Adam. A hybrid CNN–LSTM model is used to capture both spatial and temporal characteristics of ECG signals. This study performed an empirical analysis comparing SGD, RMSProp, and Adam in, who reported that adaptive optimizers converge faster than the other optimizer

Table 2: Performance analysis

Optimizer	*Accuracy (%)*	*Sensitivity (%)*	*Specificity (%)*	*Precision (%)*	*F1-Score (%)*
SGD	91.84	89.35	92.56	90.02	89.68
Momentum	93.62	90.88	94.10	91.71	91.29
Adagrad	92.14	88.45	93.37	89.26	88.85
Adadelta	94.38	91.57	95.42	92.68	92.12
RMSprop	95.67	94.03	96.12	94.39	94.21
Adam	98.26	96.15	97.43	96.78	96.46

(Arshiya Mobeen M and Saistha N, 2022). The comparison of deep learning optimizers on the PTB-XL ECG dataset exemplifies how important it is to select an appropriate optimizer for optimizing models that converge, generalize and have good performance (Jour,Yang Shunxiang, Lian cheng, Zeng Zhigang, Xu Bingrong, Zang Junbin and Zhang Zhidong, 2023). Experiments show that advanced optimizers with the architectural innovation (attention mechanisms and multi-scale networks) can boost classification performance to a large extent, validating the fact that optimizer-model synergy is important for better biomedical signal analysis.

Quantitative Results

The performance results for each optimizer are summarized in the following Table 2.

Comparative Analysis

The results clearly demonstrate that the Adam optimizer significantly outperforms all other tested optimizers across every metric. Adam's strong results stem from its per-parameter adaptive learning rates, computed from running estimates of the gradient's first moment (mean) and second moment. By doing so, it copes well with noisy gradients and changing (non-stationary) objectives, conditions often encountered in real-world ECG data. Figure 4 depicted the diagrammatic representation of performance analysis.

- SGD displayed the weakest performance, primarily due to its reliance on a fixed learning rate, which leads to slow convergence and difficulty escaping local minima in complex optimization landscapes.
- Momentum improved upon basic SGD by incorporating a velocity term that accelerates gradients in the correct direction, yet it still lacked the adaptiveness required to achieve high performance on this biomedical dataset.
- Adagrad, while effective in sparse data contexts, suffered from an excessively decaying learning rate over time, which prevented it from reaching higher accuracy levels.
- Adadelta introduced an adaptive component to counter Adagrad's decay, but its performance was moderate and unstable across different training epochs.
- RMSprop performed strongly, closely trailing Adam in most metrics. Its use of exponentially weighted moving averages helped stabilize training, but it lacks the bias-correction and second-order momentum estimation that make Adam more robust.
- Adam not only converged faster than all other optimizers but also maintained a balance between sensitivity (96.15%) and specificity (97.43%), which is critical in medical diagnostics where both false positives and false negatives must be minimized.

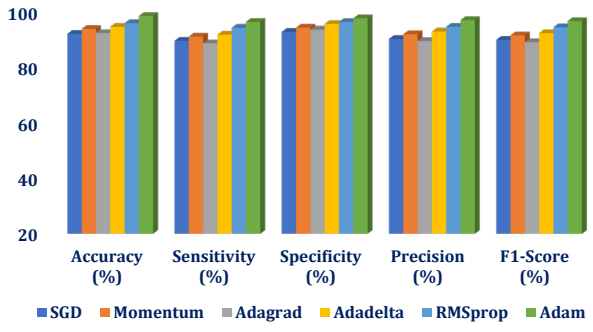

Figure 4: Performance analysis

Interpretation and Implications

The results suggest that optimizer selection plays a vital role in the successful training of deep learning models for ECG-based arrhythmia classification. Given that biomedical signals like ECGs are noisy, non-linear, and time-varying, optimizers must adapt quickly and efficiently to changing gradients. Adam's dynamic update mechanism allows it to generalize well even with limited training data per class, which is often the case in real-world medical datasets.

Summary

- Adam optimizer demonstrated the best overall performance, outperforming all others in accuracy, stability, and speed of convergence.
- RMSprop was a close second but lacked the adaptive momentum benefits that gave Adam its edge.
- SGD, Adagrad, and Adadelta were suboptimal for the non-stationary and multi-scale nature of ECG signals.
- The hybrid CNN-LSTM model, when paired with Adam, achieved state-of-the-art results on the PTB-XL dataset.
- These findings highlight the importance of optimizer selection in deep learning models used for biomedical applications and support the adoption of Adam as a default choice for similar time-series classification tasks.

Conclusion

This study presented a comprehensive evaluation of six widely used deep learning optimizers—SGD, Momentum, Adagrad, Adadelta, RMSprop, and Adam—for arrhythmia classification using the PTB-XL ECG dataset. A hybrid CNN-LSTM model was implemented to capture spatial and temporal patterns within ECG signals.

Quantitative results clearly demonstrate that the choice of optimizer significantly affects model performance. Among the compared methods, the Adam optimizer outperformed all others, achieving 98.26% classification accuracy, 96.15% sensitivity, 97.43% specificity, 96.78% precision, and an F1-score of 96.46%. These results validate Adam's adaptive learning capabilities and robustness in handling high-dimensional biomedical signals.

In conclusion, Adam emerges as the most effective optimizer for ECG-based arrhythmia detection, offering a strong foundation for real-time clinical decision support systems. Future work will explore hybrid optimization strategies, ensemble learning, and explainability for enhanced model transparency and trustworthiness in healthcare environments.

Summary of Key Findings

A hybrid CNN-LSTM model was successfully implemented for classifying arrhythmia using the PTB-XL 12-lead ECG dataset. Six popular deep learning optimization algorithms—SGD, Momentum, Adagrad, Adadelta, RMSprop, and Adam—were systematically evaluated. Among all optimizers, the Adam optimizer consistently yielded the best performance, achieving: Accuracy: 98.26%, Sensitivity: 96.15%, Specificity: 97.43%, Precision: 96.78%, F1-Score: 96.46%. The results clearly show that Adam's adaptive learning rates and efficient convergence are especially well-matched for high-dimensional biomedical time-series such as ECG signals. The study confirms that optimizer selection is critical to the success of deep learning models in healthcare applications.

Future Enhancements

To further advance this research, the following enhancements are proposed:

Ensemble Learning

Combine multiple deep learning architectures (e.g., CNN-LSTM, BiLSTM, Transformer) to boost classification robustness.

Explainable AI (XAI)

Integrate methods like Grad-CAM or SHAP to provide interpretability of model decisions for clinical validation.

Multi-Lead Feature Fusion

Incorporate features from all 12 ECG leads, using attention mechanisms or multi-stream architectures to improve diagnostic granularity.

Real-Time Deployment

Develop lightweight versions of the model for deployment on portable or mobile ECG monitoring devices.

Transfer Learning

Apply pretrained models on other ECG datasets (e.g., MIT-BIH) to validate generalizability.

Clinical Integration

Collaborate with healthcare professionals to test the model in real-world diagnostic workflows, enhancing clinical relevance.

Acknowledgements

None.

References

Aadil Gani Ganie and Samad Dadvandipour (2023). From big data to smart data: a sample gradient descent approach for machine learning. *Journal of Big Data. https://doi.org/10.1186/s40537-023-00839-9*.

Arshiya Mobeen M and Saistha N (2022). Comparative Study on Image Classification Using Different Optimizer. *iJournals: International Journal of Software & Hardware Research in Engineering (IJSHRE)*.

Carmina Fjellstrom and Kaj Nystrom (2022). Deep learning, Stochastic gradient descent and Diffusion maps. *Journal of Computational Mathematics and Data Science. https://doi.org/10.1016/j.jcmds.2022.100054*.

Deekshith Shetty, Harshavardhan C A, M Jayanth Varma, Shrishail Navi, Mohammed Riyaz Ahmed (2020). Diving Deep into Deep Learning: History, Evolution, Types and Applications. *International Journal of Innovative Technology and Exploring Engineering (IJITEE) - ISSN: 2278-3075 (Online), Volume-9 Issue-3, P.No. 2835 – 2846. DOI: 10.35940/ijitee.A4865.019320*.

Dongpo Xu, Shengdong Zhang , Huisheng Zhang and Danilo P. Mandic (2021). Convergence of the RMSProp deep learning method with penalty for nonconvex optimization. *Neural Networks. https://doi.org/10.1016/j.neunet.2021.02.011*

Iqbal H. Sarker (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science. https://doi.org/10.1007/s42979-021-00815-1*.

Jour,Yang Shunxiang, Lian cheng, Zeng Zhigang, Xu Bingrong, Zang Junbin and Zhang Zhidong (2023). A Multi-View Multi-Scale Neural Network for Multi-Label ECG Classification. *IEEE Transactions on Emerging Topics in Computational Intelligence 2023/06/01. Available from: 10.1109/TETCI.2023.3235374*.

Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie & Laith Farhan (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data. https://doi.org/10.1186/s40537-021-00444-8*.

Liu Yang and Deng Cai (2021). AdaDB: An adaptive gradient method with data-dependent bound. *Neurocomputing. https://doi.org/10.1016/j.neucom.2020.07.070*.

M. Durairaj and S. Selvakumari (2025). A Hybrid CNN-LSTM Framework for ECG Classification with Genetic Algorithm-Based Feature Optimization. *Indian Journal of Science and*

Technology, 18, 31. Available from: https://doi.org/10.17485/IJST/v18i31.1160.

Mohamed Reyad, Amany M. Sarhan, and M. Arafa (2023). A modified Adam algorithm for deep neural network optimization. *Neural Computing and Applications. https://doi.org/10.1007/s00521-023-08568-z.*

Mohammad-Parsa Hosseini, Senbao Lu, Kavin Kamaraj, Alexander Slowikowski and Haygreev C. Venkatesh (2020). Deep Learning Architectures. *Springer Nature Switzerland AG. https://doi.org/10.1007/978-3-030-31756-0_1.*

S. Selvakumari and M. Durairaj (2025). A Comparative Study of Optimization Techniques in Deep Learning Using the MNIST Dataset. *Indian Journal of Science and Technology, 18, 10. Available from: https://doi.org/10.17485/IJST/v18i10.121.*

Shiv Ram Dubey, Satish Kumar Singh, Bidyut Baran Chaudhuri (2022). Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing, Volume 503, Pages 92-108. https://doi.org/10.1016/j.neucom.2022.06.111.*

Siddharth Sharma, Simone Sharma & Anidhya Athaiya (2020). Activation Functions in Neural Networks. *International Journal of Engineering Applied Sciences and Technology, Vol. 4, Issue 12, ISSN No. 2455-2143, Pages 310-316.*