IRSTI 28.23.15 https://doi.org/10.26577/jpcsit20253203

Alfarabi Mazhit [1,*], Nazgul Zakariyanova [2]

[1] Kazakh-British Technical University, Almaty, Kazakhstan
[2] Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: a-mazhit@mail.ru

COMPARATIVE ANALYSIS OF CNN MODELS FOR DETECTING CARDIOVASCULAR DISEASES

Abstract. In modern medical practice, cardiovascular diseases (CVDs) remain among the leading causes of mortality and morbidity worldwide. Electrocardiography (ECG) continues to be one of the essential non-invasive methods for diagnosing cardiac rhythm disturbances and other myocardial pathologies. However, traditional ECG analysis relies heavily on specialist interpretation, which can be subjective and prone to human error. With advances in machine learning and deep learning, there is now an opportunity to automate and standardize the diagnostic process. In this study, convolutional neural networks (CNNs)–specifically the ResNet50 and VGG16 architectures–are applied to classify ECG images. Hyperparameter tuning is conducted to optimize batch size and learning rate. In addition, a prototype web service is presented, implemented with React for the frontend and Django REST Framework for the backend, that allows real-time, automated ECG classification. This approach has the potential to ease the workload of cardiologists and enhance the objectivity of diagnosis in clinical environments.

Keywords: machine learning, ECG classification, deep neural networks, ResNet50, VGG16, hyperparameter tuning, Django REST Framework, React, cardiology, computer vision

1. Introduction

Cardiovascular diseases (CVDs) are widely recognized as one of the foremost causes of mortality worldwide, consistently accounting for a substantial proportion of death and disability rates according to recent reports by the World Health Organization [1]. Numerous strategies have been employed in clinical practice to manage, prevent, and diagnose these conditions, reflecting the immense global burden placed on healthcare systems. Despite the availability of a wide array of diagnostic methods, electrocardiography (ECG) remains fundamental due to its non-invasive nature, cost-effectiveness, and ability to provide rapid insights into cardiac health. ECGs capture the electrical activity of the heart over a specific duration, making them instrumental for detecting a variety of arrhythmias, ischemic changes, and other cardiac pathologies. Nevertheless, conventional ECG interpretation relies heavily on specialized expertise, which can be both time-consuming and prone to inter-observer variability [2]. As a result, automated and standardized approaches to ECG analysis have become increasingly desirable to improve diagnostic precision and efficiency.

The landscape of medical diagnostics has evolved significantly with advances in machine learning (ML) and deep learning (DL). Traditional ML algorithms, such as support vector machines or random forests, have demonstrated their utility in tasks like anomaly detection and feature-based classification. However, these methods often depend on handcrafted features and may lack robustness when applied to complex medical images. By contrast, deep learning models–particularly convolutional neural networks (CNNs)–have shown remarkable capabilities in extracting hierarchical features directly from images or signals, obviating the need for extensive manual feature engineering [3]. CNNs have proven efficacious in domains ranging from radiology, where they support the classification of X-ray, CT, and MRI scans, to dermatology, where they assist in identifying malignant skin lesions from dermoscopic images [4]. This success naturally extends to ECG analysis, where CNNs can learn patterns in waveforms that correlate with specific cardiac conditions.

Among the diverse CNN architectures introduced over the past decade, two have consistently garnered attention in the research community: VGG16 and ResNet50. VGG16 is known for its systematic arrangement of convolutional and pooling layers, featuring 16 weight layers in total [5]. Despite its relatively older design, VGG16 remains a popular choice due to its simplicity and effectiveness, making it easier to interpret and modify. In

contrast, ResNet50 addresses the vanishing gradient problem by introducing residual connections that allow gradients to flow more efficiently through deeper networks [6]. Such connections effectively create "shortcuts" within the network, improving both training stability and final performance. This architectural innovation makes ResNet50 appealing for tasks requiring high accuracy, especially when large datasets or complex image structures are involved.

The choice between these architectures can be influenced by multiple factors, including dataset size, computational resources, and the specific characteristics of the images in question. ECG images, which often present subtle morphological differences between pathological and normal waveforms, can benefit from deeper representations if the dataset is sufficiently diverse and well-annotated. Conversely, simpler architectures may suffice when data is limited or less complex. Regardless of the model, hyperparameter tuning is critical to maximizing performance. Hyperparameters such as batch size and learning rate play a pivotal role in network convergence and generalization [7]. A large batch size might expedite computations on modern GPU clusters but risks smoothing out gradient signals, while an excessively high learning rate can cause the model to skip local minima and converge poorly. Systematic experimentation and grid or random search methods typically guide the selection of optimal hyperparameters.

In addition to choosing the most suitable CNN architecture and refining model hyperparameters, it is crucial to design a functional solution that can be seamlessly integrated into clinical workflows. Academic and industrial research often emphasize the accuracy of classification models but give less attention to the process of deploying such models in real-world healthcare settings. To address this gap, a web-based platform has been developed, leveraging React for building an interactive user interface and Django REST Framework for managing server-side logic and API endpoints. Through this system, medical personnel are able to upload ECG images with minimal effort. Once uploaded, each image is processed in near real-time by one of the CNN models–ResNet50 or VGG16–trained on a labeled ECG dataset. The classification results are then returned to the user interface, reducing the time required for diagnostic interpretation and supporting more consistent decision-making.

Such an application can prove invaluable in regions lacking adequate medical infrastructure. Rural or underserved areas may face shortages of qualified cardiologists, leading to delays in diagnosis and treatment. By enabling frontline healthcare workers or technicians to quickly obtain classification results, the proposed approach can facilitate timely clinical interventions and potentially improve patient outcomes. Moreover, the scalability of a web-based solution allows it to be deployed in multiple clinical facilities, with the possibility of centrally updating or retraining the models as new data becomes available.

Future endeavors can incorporate additional refinements, such as data augmentation to enlarge the diversity of training samples, domain adaptation for handling ECG variations across different populations, and multi-class classification to discriminate among multiple arrhythmia subtypes. Advanced frameworks, including Transfer Learning and Explainable Artificial Intelligence (XAI), may further enhance model performance and transparency [8]. Transfer Learning can harness existing pre-trained weights–obtained from large, general-purpose image datasets–to accelerate the training process on relatively modest ECG data collections. Meanwhile, XAI can illuminate the regions or features within an ECG image that contribute most significantly to the model's classification decision, building greater trust among clinicians.

2. Literature review

A substantial body of research has explored the utilization of convolutional neural networks (CNNs) for electrocardiogram (ECG) classification, encompassing both raw time-series signal data and visual representations in the form of ECG images. This duality of approaches stems from the flexibility of CNNs to operate across different data modalities while maintaining high diagnostic performance.

One of the pioneering contributions in this area was made by Acharya et al. [9], who developed a deep CNN model specifically designed to detect myocardial infarction (heart attack) directly from raw ECG signals. Their model architecture included multiple convolutional and pooling layers tailored to capture relevant features from the waveform data, culminating in a fully connected layer that performed the classification. The experimental results demonstrated a remarkably high level of accuracy, sensitivity, and specificity, reinforcing the potential of deep learning to outperform traditional rule-based or manually engineered approaches. Although this work relied on one-dimensional signal input, the underlying architectural concepts–such as hierarchical

feature extraction and layer-wise abstraction–are directly transferable to two-dimensional image-based CNNs. This connection underscores the versatility of CNN architectures across different ECG formats.

In a complementary effort, Kachuee et al. [10] introduced a hybrid deep learning architecture that combined the strengths of CNNs with recurrent neural networks (RNNs) to enhance arrhythmia detection. In their system, CNNs were responsible for spatial feature extraction from short windows of ECG signals, while RNNs, specifically gated recurrent units (GRUs), captured temporal dependencies and sequence-level context. This architecture effectively leveraged the strengths of both model types: the CNN's ability to extract robust spatial features and the RNN's capacity for handling sequential information. Additionally, the authors incorporated residual connections–originally popularized in ResNet architectures–to combat issues such as vanishing gradients and to allow for deeper, more expressive models. This architectural choice set the stage for later adaptations, including the adoption of deeper CNNs like ResNet50 in ECG tasks.

Focusing specifically on image-based ECG analysis, Luz et al. [11] conducted a notable study demonstrating the potential of CNNs to classify arrhythmias using two-dimensional representations of ECGs. These visual formats, often created by plotting ECG signals and converting them into grayscale or RGB images, allow CNNs to identify spatial patterns and morphological features–such as QRS complex shape, ST-segment deviations, or P-wave abnormalities–that may be less evident in raw signal form. Luz et al.'s results offered some of the first concrete evidence that CNNs trained on ECG images could achieve comparable, if not superior, performance to traditional signal-based methods. This opened new avenues for using well-established image classification architectures in cardiology.

Building on this trend, Yildirim [12] introduced an innovative hybrid approach that integrated discrete wavelet transforms (DWTs) with CNNs to perform heartbeat classification. Wavelet transforms were employed to decompose the ECG signal into multiple frequency bands, capturing both time-domain and frequency-domain information. These transformed signals were then rendered as images and fed into a CNN for classification. This methodology allowed for the preservation of both temporal resolution and spectral features, offering a richer input representation. The resulting model achieved strong classification metrics and further confirmed the adaptability of CNNs when applied to hybrid signal-image domains.

A major milestone in deep learning-based ECG interpretation came from the work of Rajpurkar et al. [13], who developed the "Cardiologist-Level" model–an advanced CNN trained on a massive dataset of over 90,000 single-lead ECG recordings. The model was capable of detecting over a dozen types of arrhythmias with diagnostic accuracy that matched or exceeded that of experienced board-certified cardiologists. Although their model operated on 1D signal data, the training methodology, dataset curation, and emphasis on clinical benchmarking set a new standard for ECG classification research. Furthermore, the study's success catalyzed broader interest in adapting similar deep learning frameworks for ECG images, as researchers sought to replicate such high performance in settings where only visual data is available.

These foundational studies–spanning both signal-based and image-based ECG analysis–provide strong justification for investigating and comparing the effectiveness of different CNN architectures, particularly VGG16 and ResNet50, in the context of ECG image classification. Both architectures have shown promise in various computer vision domains, yet their comparative strengths and limitations in the domain of medical imaging, specifically ECGs, remain an active area of exploration. The current study aims to build upon this body of work by implementing and evaluating these two models using a curated dataset of ECG images, with the ultimate goal of identifying which architecture offers superior performance for this clinically significant task.

3. Materials and Methods

3.1 Dataset Overview

The dataset, shown in Figure 1, originates from the Kaggle repository titled "ECG Images Dataset of Cardiac Patients" prepared by user EvilSpirit05 in 2023 [14].

Unlike many ECG collections that offer raw waveform signals, this resource supplies complete ECG *images*, each in a consistent resolution and brightness level, thereby reducing the variability stemming from differences in scanning equipment or lighting conditions. These images generally capture one or more distinct heartbeat cycles, labeled by professional or automated annotation systems. Possible labels range from normal sinus rhythm to various arrhythmic conditions, and each image is stored in a standardized format (for instance, PNG or JPEG) at identical pixel dimensions.

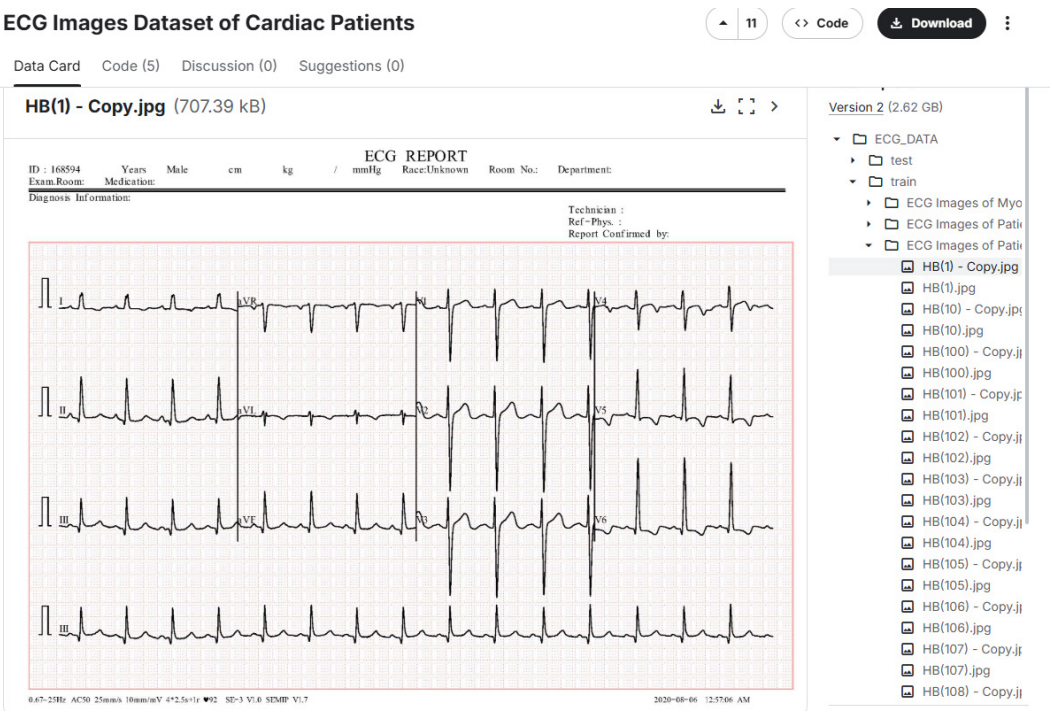

Figure 1 – ECG images dataset

Because of its inherent uniformity, the dataset typically eliminates the need for extensive resizing or brightness normalization. Each image retains a similar visual structure: a standard grayscale depiction of the ECG trace on a background with controlled contrast and clarity. The class balance can vary (some pathologies may be underrepresented), but the consistent presentation across images simplifies model training since convolutional neural networks can more readily generalize from a uniform set of visual patterns. Researchers accessing this Kaggle resource thus begin with a stable baseline, negating common difficulties such as irregular cropping or widely differing resolutions.

To ensure robust evaluation of the proposed models, the process was divided into three clearly defined stages: *training, validation,* and *testing*. These stages were applied consistently to both VGG16 and ResNet50.

- *Training Stage:* 70% of the total ECG image dataset was used to train the models. During this phase, weights were optimized using backpropagation. Data were fed in mini-batches of 16 or 32 images, resized to 224×224 pixels, and normalized to a [0, 1] scale. Data augmentation techniques were optionally applied, though the dataset's consistency often rendered them unnecessary. Models were trained using the Adam or SGD optimizer, with dropout and batch normalization optionally applied to prevent overfitting.

- *Validation Stage*: 15% of the data was reserved for validation, helping monitor model generalization in real time. After each epoch, performance on this set was evaluated using metrics such as validation loss, accuracy, and F1-score. Early stopping was applied if no improvements were observed over a predefined number of epochs (typically 5), and the best-performing model checkpoint was saved.

- *Testing Stage*: The final 15% of the data, not used during training or validation, was used as a hold-out test set. After model training and selection, this unseen subset was used for final performance evaluation. Confusion matrices and classification metrics–including accuracy, precision, recall, and F1-score–were computed to assess true generalization performance.

This structured workflow ensured reproducibility, reduced overfitting risks, and provided objective measurements of model effectiveness under real-world constraints.

3.1.1 Data preprocessing

The validation set played a key role in tuning hyperparameters like learning rate and batch size, as well as preventing overfitting through early stop-

ping. The test set remained untouched until the final performance assessment, ensuring an unbiased evaluation of the trained models.

Although the Kaggle images were already uniform in resolution, an additional check was performed to confirm they matched the 224×224 input size typically expected by VGG16 and ResNet50. In cases where the images exceeded this dimension or had a slightly different aspect ratio, they were resized (or padded/cropped minimally) to 224×224 pixels. This resizing step guaranteed compatibility with the input layers of both CNN architectures and ensured consistent spatial dimensions across the entire dataset.

Because these ECG images already exhibited consistent brightness, grayscale values, and minimal noise, no further normalization–beyond dividing pixel intensities by 255.0–was required in most experiments. This step placed each pixel in the [0,1] range, stabilizing gradient updates during backpropagation. During training, runtime checks confirmed that each batch loaded properly sized 224×224 images, which were then passed to the convolutional layers without additional color-space transformations or extensive augmentations.

3.2 Model Architectures

Deep convolutional neural networks remain central to many image-classification tasks, including medical applications such as ECG analysis. Two prominent architectures, VGG16 and ResNet50, were chosen to classify the standardized ECG images in this project. Each network was originally developed for large-scale natural image recognition but can be adapted efficiently to medical datasets, particularly when those datasets are uniform in resolution and brightness.

3.2.1 VGG16

VGG16, introduced by Simonyan and Zisserman in 2015, comprises thirteen convolutional layers and three fully connected layers, making up sixteen weight layers in total. Its design is centered on repeated blocks that stack 3×3 convolutions and use ReLU activations, with intermittent max pooling to reduce spatial dimensions. Figure 2 below offers a schematic of this architecture, illustrating how each convolution and pooling stage feeds into the next.

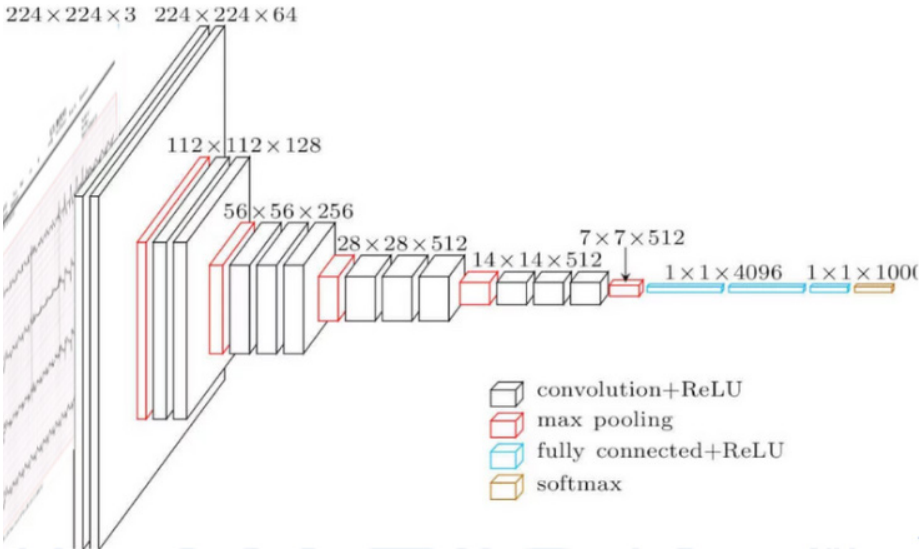

Figure 2 – Schematic representation of VGG16.

Although it was originally created for a 1,000-class ImageNet task, the final fully connected layer in VGG16 can be replaced to match the number of ECG categories in the Kaggle dataset. Dropout layers are often introduced within the fully connected portion to mitigate overfitting, and batch normalization may be incorporated if faster convergence or additional training stability is needed. Because this dataset consists of uniform ECG images, VGG16 benefits from a consistent input dimension of 224×224 pixels and does not require elaborate preprocessing for contrast or brightness adjustments. Transfer learning from ImageNet-pretrained weights commonly accelerates convergence, allowing the network to adapt to distinct visual signals associated with normal or arrhythmic waveforms [19-20].

3.2.2 ResNet50

ResNet50, part of the Residual Network family established by He et al. in 2016, tackles the training difficulties observed in deep networks through the use of skip connections. These connections enable gradients to circumvent some convolutional layers, reducing the vanishing-gradient problem. Figure 3 presents a schematic representation of ResNet50.

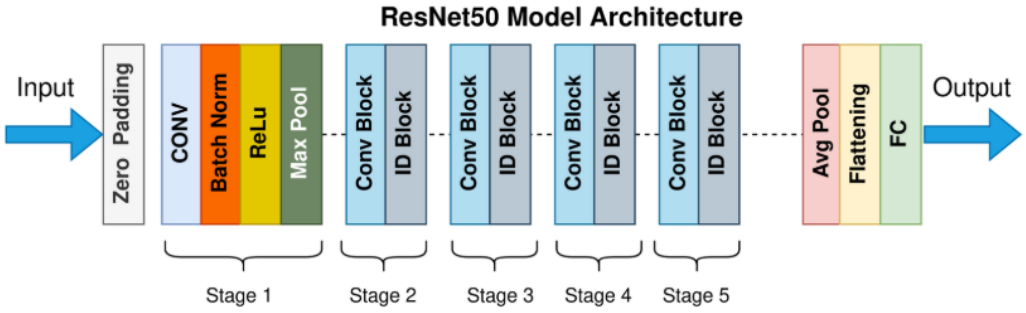

Figure 3 – Schematic representation of VGG16.

This design supports a depth of fifty layers, substantially more than VGG16, and permits the model to capture intricate visual features when provided with adequate data. The final layers of ResNet50, originally geared toward 1,000 ImageNet classes, are fine-tuned or replaced to handle the smaller number of categories in the ECG dataset. Batch normalization is built into the architecture, minimizing covariate shift and supporting stable training even at deeper layers. Transfer learning from ImageNet is also possible, ensuring that learned representations are reused while the model adapts to the morphological nuances of heartbeats captured in uniform ECG imagery.

VGG16's simpler structure can offer ease of interpretation, whereas ResNet50, by being deeper, may learn more nuanced distinctions among waveforms. The remainder of this paper examines how these architectural differences translate into varying classification accuracies, training dynamics, and model complexities.

3.3. Hyperparameter Tuning

Hyperparameter tuning played a pivotal role in optimizing the performance of both VGG16 and ResNet50 when classifying the Kaggle-based ECG images, echoing broader findings on the importance of choosing effective learning rate schedules for deep learning [15]. This step primarily targeted batch size and learning rate, as these two parameters have the most direct impact on model convergence and generalization.

These particular hyperparameters were selected because they exert a disproportionately high influence on the dynamics of training deep neural networks. The learning rate determines the magnitude of weight updates during backpropagation, directly influencing how quickly and effectively the model learns [16-18]. An improper learning rate can either cause the model to diverge (if too high) or lead to extremely slow convergence (if too low). Meanwhile, batch size affects the frequency and stability of gradient updates. Smaller batches provide noisy but potentially more generalizable updates, while larger batches allow for more efficient computation and smoother gradients.

Other hyperparameters, such as number of epochs, activation functions, or optimizer choice, were held constant in this study. This allowed the investigation to focus on learning rate and batch size, which are widely recognized in the deep learning community as the most impactful and sensitive hyperparameters for tuning convergence and performance.

The choice of batch sizes (16 and 32) was grounded in balancing computational efficiency and model performance. A batch size of 16 was selected initially to provide more frequent weight updates, which introduces helpful noise during gradient descent and can help the model escape local minima. However, it also increases training time. A batch size of 32, in contrast, is well-aligned with the memory capacity of modern GPUs and allows for stable convergence without overloading system resources. Our experiments showed that a batch size of 32 consistently yielded higher accuracy, likely due to smoother gradients and more consistent loss minimization over mini-batches.

The learning rates (0.001, 0.0001, 0.00001) were chosen to represent a descending range from fast to slow convergence. A rate of 0.001 is standard in many CNN setups and was expected to give rapid initial convergence. However, deeper networks like ResNet50 showed instability at this rate. The mid-level rate of 0.0001 provided the best trade-off: sufficiently fast learning while maintaining stability and avoiding overshooting minima. The smallest rate, 0.00001, ensured extremely gradual updates–useful for fine-tuning or avoiding divergence–but at the cost of longer training times.

This tuning process followed a *grid search approach*, a systematic method for exhaustively searching through a predefined subset of hyperparameters. Specifically, the combinations of two batch sizes (16 and 32) and three learning rates (0.001, 0.0001, 0.00001) were evaluated, forming a total of six configurations. Each configuration was trained independently on the same training and validation splits to ensure consistency and fairness in comparison. For each run, metrics such as accuracy, precision, and F1-score were monitored to assess performance.

Grid search was chosen over random search or Bayesian optimization due to its simplicity and complete coverage of the small parameter space. Since the computational cost was manageable–thanks to the relatively small dataset size and the use of transfer learning–grid search offered a practical balance between thoroughness and efficiency. Ultimately, it enabled the precise identification of optimal configurations without introducing additional tuning complexity. For each pair, performance metrics were logged on both validation and test sets. As summarized in the Results section, VGG16 with batch size 32 and learning rate 0.0001 or 0.00001 performed best, reaching 98% accuracy. These results indicate that even on relatively small, clean datasets, proper tuning is vital to achieving optimal performance.

3.4. Training Procedure and Implementation Details

Following hyperparameter tuning, the chosen configurations for learning rate and batch size were applied consistently to VGG16 and ResNet50 across multiple training runs. Training was conducted on a Python 3.9 environment, primarily using TensorFlow (Keras) or PyTorch. GPU acceleration, often provided by NVIDIA RTX-series devices, allowed mini-batch gradient updates at a reasonable speed even for deeper architectures like ResNet50. Each epoch processed all images within the training set, with the optimizer–either Adam or stochastic gradient descent with momentum–updating weights via backpropagation. Dropout and batch normalization were introduced in certain experiments to combat overfitting or stabilize activation distributions.

An early stopping strategy again leveraged validation metrics to determine when training should halt. Once improvements in validation accuracy or F1-score plateaued for several consecutive epochs, the process ended to conserve resources and avoid overfitting. A final model checkpoint, representing the highest validation performance, was reserved for the test evaluation phase. Because the ECG images all shared a 224×224 resolution and consistent brightness, runtime preprocessing remained minimal beyond ensuring that each batch loaded correctly.

A brief web application prototype was developed to illustrate how these trained models might function in a clinical or research environment. The backend, built with Django REST Framework, accepted uploaded ECG images and performed classification with either VGG16 or ResNet50, depending on which model was selected as the top performer. A React-based frontend allowed users to upload image files through a concise interface, and the server responded with class predictions in real time. The screenshot below (Figure 4) shows an example of the page where users can upload their ECG images for classification.

In a full production setting, security measures such as HTTPS, authentication, and audit logging would be implemented to satisfy data protection requirements [21-22]. Containerization solutions could also be adopted to facilitate scalability across multiple clinical sites or research laboratories. Although this prototype was not configured for large-scale deployments, it demonstrated the feasibility of integrating a CNN-based ECG classification system into a practical application accessible to end users. By automating the image upload and inference processes, the tool potentially reduces the need for specialized technical expertise while offering near-instant diagnostic support.

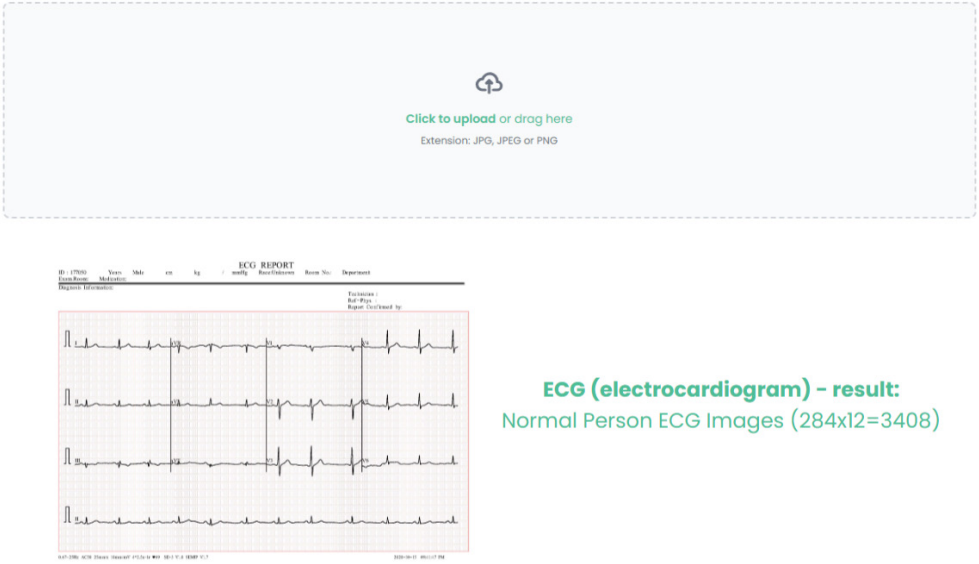

Figure 4 – Sample upload interface for the ECG classification web application.

3.5. Technical Implementation Details

The training and evaluation of the models were conducted in a high-performance computing environment equipped with an NVIDIA Tesla T4 GPU (15 GB VRAM), running CUDA 12.4 and driver version 550.54.15. The experiments were implemented in Python 3.11.12 using TensorFlow 2.18.0 as the primary deep learning framework. More details shown on Figure 5.

All models were trained using GPU acceleration, with the TensorFlow backend configured to automatically allocate GPU memory as needed. During training, typical memory usage ranged from 8.3 GB to 11.5 GB, depending on the model size and batch configuration.

Both VGG16 and ResNet50 architectures were initialized using ImageNet pre-trained weights and fine-tuned on the ECG image dataset. Model training was performed using a batch size of 32, a learning rate of 0.0001, and 30 epochs per run. The Adam optimizer was employed with categorical cross-entropy as the loss function.

Each epoch took approximately 60–80 seconds, depending on model complexity. Each epoch training time shown on Figure 6 for VGG16 and Figure 7 for ResNet50.

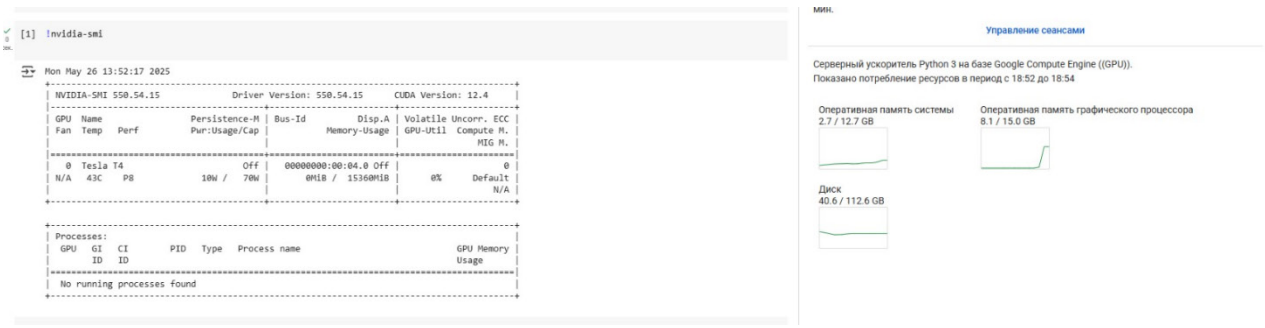

Figure 5 – Technical parameters of computer.

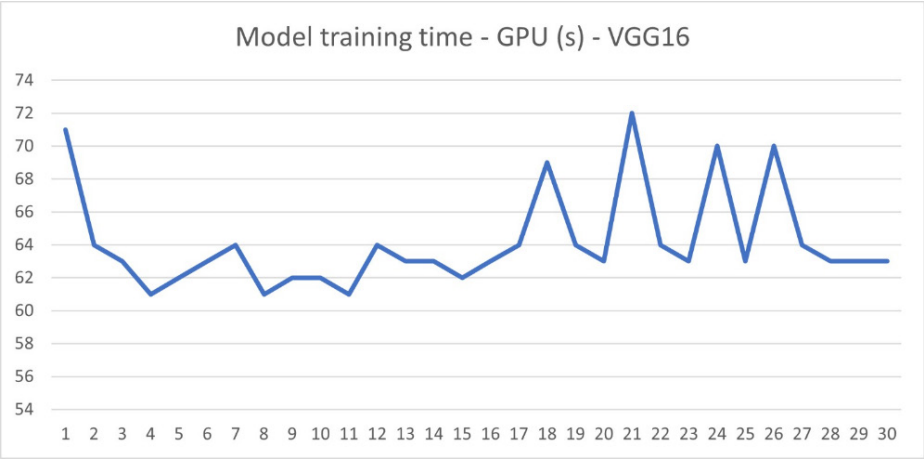

Figure 6 – VGG16 model training time.

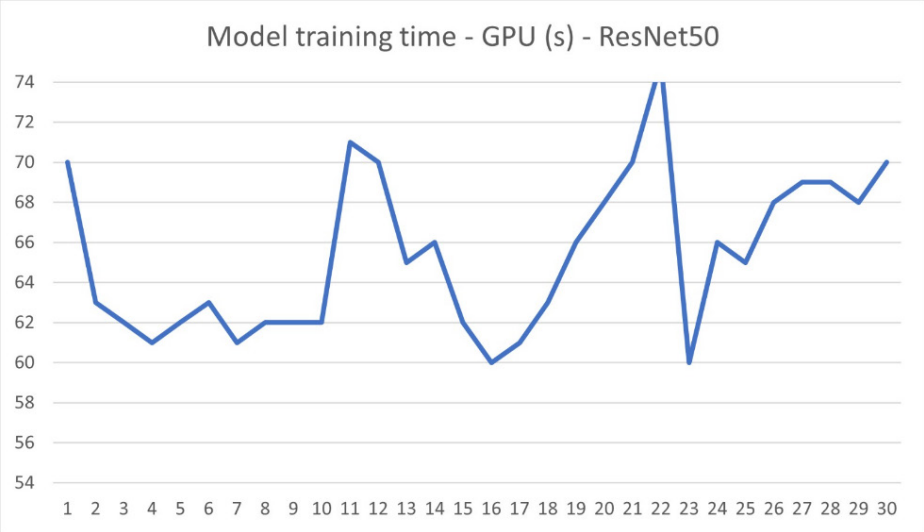

Figure 7 – ResNet50 model training time.

In addition, early stopping was implemented to prevent overfitting, and training logs (loss, accuracy, and validation metrics) were recorded for all runs. Post-training evaluation was conducted on a separate test set using confusion matrices and F1-score analysis.

4. Results

Initial experiments compared VGG16 and ResNet50 under identical training conditions: epoch count set to 20, a batch size of 16, and a learning rate of 0.0001. Under these parameters, VGG16 reached a classification accuracy of 92%, while ResNet50 attained 81%. Figure 8 presents the confusion matrix for VGG16 in this initial run, illustrating that misclassifications were generally concentrated in a small subset of classes. Figure 9 shows the corresponding confusion matrix for ResNet50, where off-diagonal entries are more pronounced, indicating that certain pathological and normal ECG categories were not as distinctly separated.

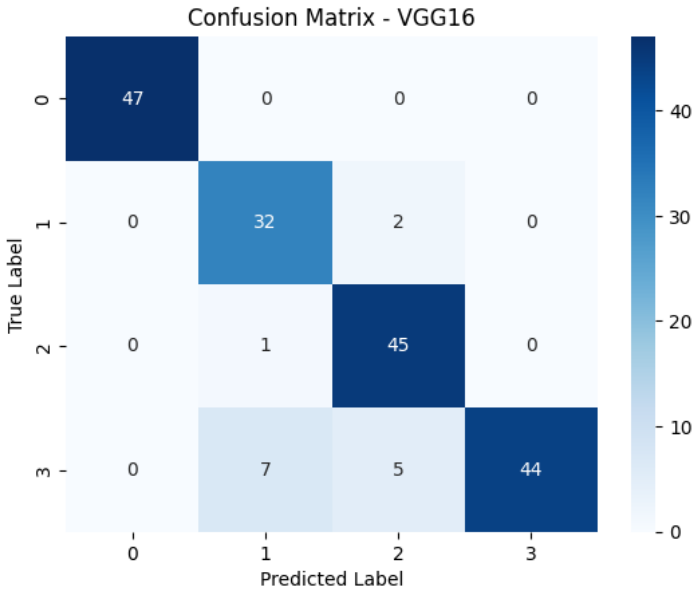

Figure 8 – Confusion matrix of VGG16 model.

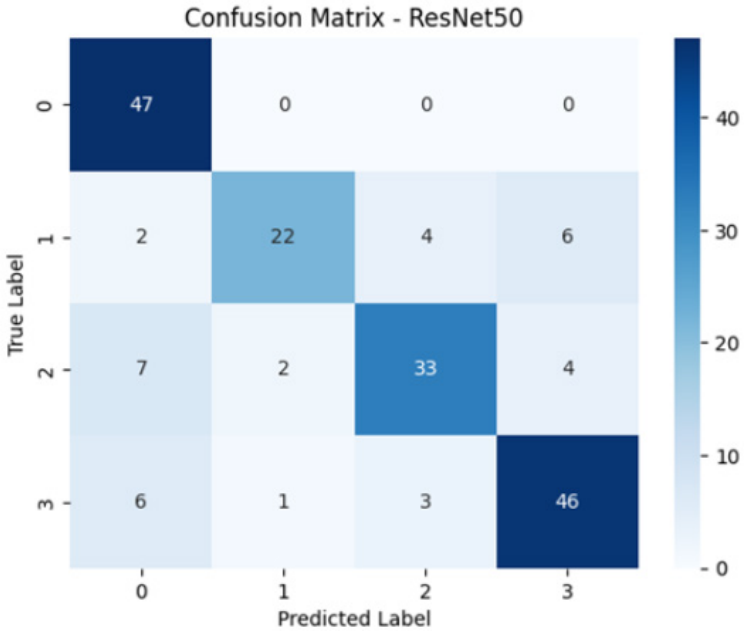

Figure 9 – Confusion matrix of ResNet50 model.

Observing that VGG16 outperformed ResNet50 in these initial trials, subsequent attention turned exclusively to VGG16 for a deeper exploration of hyperparameters. The new tests all remained at 20 epochs but varied batch size (16 or 32) and learning rate (0.001, 0.0001, 0.00001). Higher batch sizes introduced fewer but more stable gradient updates each epoch, while lower learning rates tended to refine weights more gradually. Table 1 below summarizes the final accuracy values across these parameter combinations:

Table 1 – Evaluation metrics of VGG16 under different hyperparameter settings.

Batch size	Learning rate	Accuracy	Recall	Precision	F1-score
16	0.001	0.97	0.9675	0.97	0.97
	0.0001	0.92	0.9275	0.9175	0.915
	0.00001	0.93	0.9225	0.925	0.925
32	0.001	0.96	0.9475	0.96	0.9525
	0.0001	0.98	0.9825	0.98	0.9825
	0.00001	0.98	0.9825	0.98	0.9825

Accuracy notably improved under several of these setups. When the batch size was 32 and the learning rate set to either 0.0001 or 0.00001, the model achieved 98%, marking the highest classification success observed in this study. Figure 10 provides a composite display of confusion matrices for each hyperparameter variation tested with VGG16. The diagrams illustrate how higher accuracies corresponded to fewer off-diagonal errors, particularly in ECG classes that presented subtle morphological distinctions. All experiments consistently ran for 20 epochs, without early stopping or reduction in the number of iterations, ensuring a uniform basis for comparison across different parameter settings.

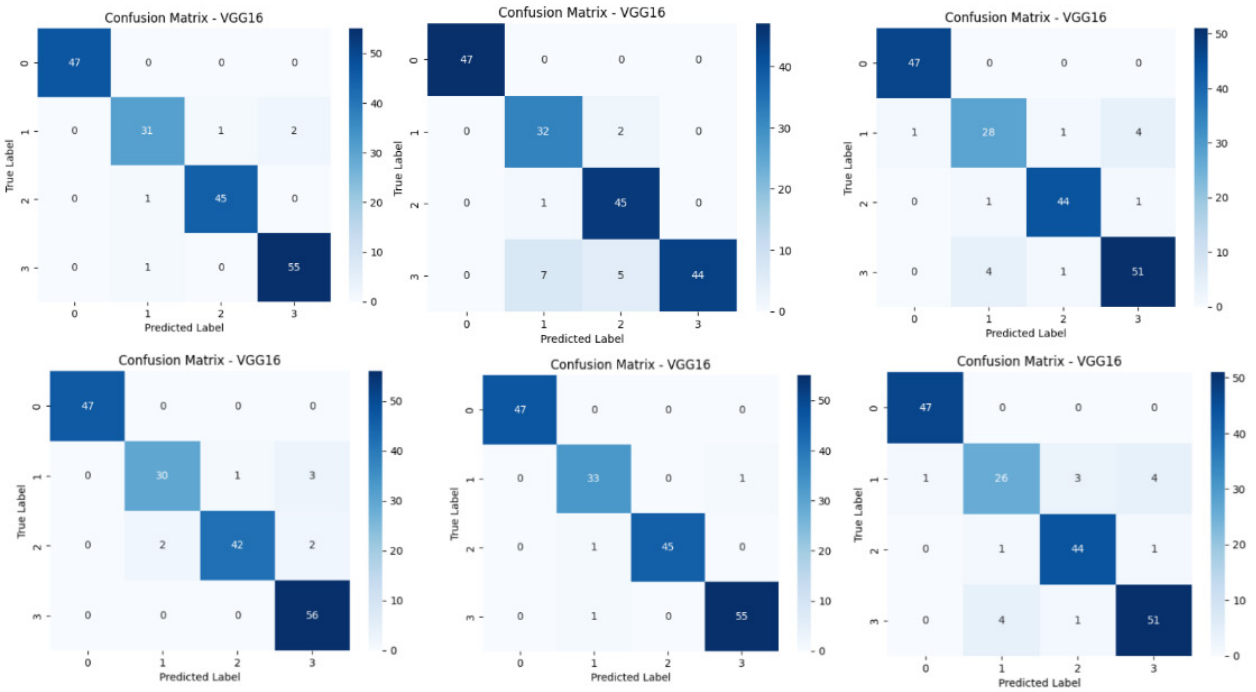

Figure 10 – Confusion matrices of models with tuned hyperparameters.

5. Discussion

The findings of this study demonstrate a significant difference in performance between two prominent convolutional neural network (CNN) architectures, VGG16 and ResNet50, when applied to the classification of standardized ECG images. Under identical training conditions, VGG16 consistently outperformed ResNet50, achieving a peak accuracy of 98% under optimal hyperparameter configura-

tions. This superior performance can be attributed to VGG16's simpler architecture, which may offer better generalization capabilities on relatively small or uniform datasets, as is the case with the ECG image dataset used in this research.

ResNet50, while theoretically more powerful due to its deeper architecture and residual connections, may be more prone to overfitting or underfitting when applied to datasets lacking diversity or scale. The confusion matrices illustrated in Figure 9 show that ResNet50 produced more misclassifications compared to VGG16, particularly among arrhythmic classes with subtle morphological differences. The reduced performance highlights the necessity for either more extensive data augmentation, regularization, or dataset scaling when deploying deeper models like ResNet50.

Hyperparameter tuning played a pivotal role in optimizing the performance of the models. The results suggest that a batch size of 32 and learning rates of 0.0001 or 0.00001 provide the best outcomes for VGG16, confirming that stability in gradient updates and moderate learning steps lead to superior convergence. Figure 10 supports this observation, showing clearer separation of classes and fewer off-diagonal entries in the confusion matrices for well-tuned configurations.

5.1. Training and Validation Curves

ResNet50: The training and validation curves for ResNet50 are illustrated in Figure 11 (accuracy and loss). As shown, the model demonstrates a gradual increase in training accuracy over 30 epochs, reaching above 80%. However, the validation accuracy is highly variable, with noticeable oscillations between epochs. This instability suggests sensitivity to specific mini-batches and possible overfitting.

In terms of loss, both training and validation loss exhibit a steady downward trend, starting from approximately 1.4 and converging toward 0.55. Despite this improvement, the final loss values remain higher than those of VGG16, indicating less effective optimization.

VGG16: Figure 12 (accuracy and loss) present the training dynamics for VGG16. The accuracy curve shows rapid convergence within the first 10 epochs, with both training and validation accuracy stabilizing above 95%. The close tracking of validation accuracy to training accuracy indicates excellent generalization.

The corresponding loss graph further supports this conclusion. Both curves descend smoothly, converging near 0.2 by the end of training. Minimal divergence between the training and validation losses demonstrates that the model is not overfitting and is learning robust representations.

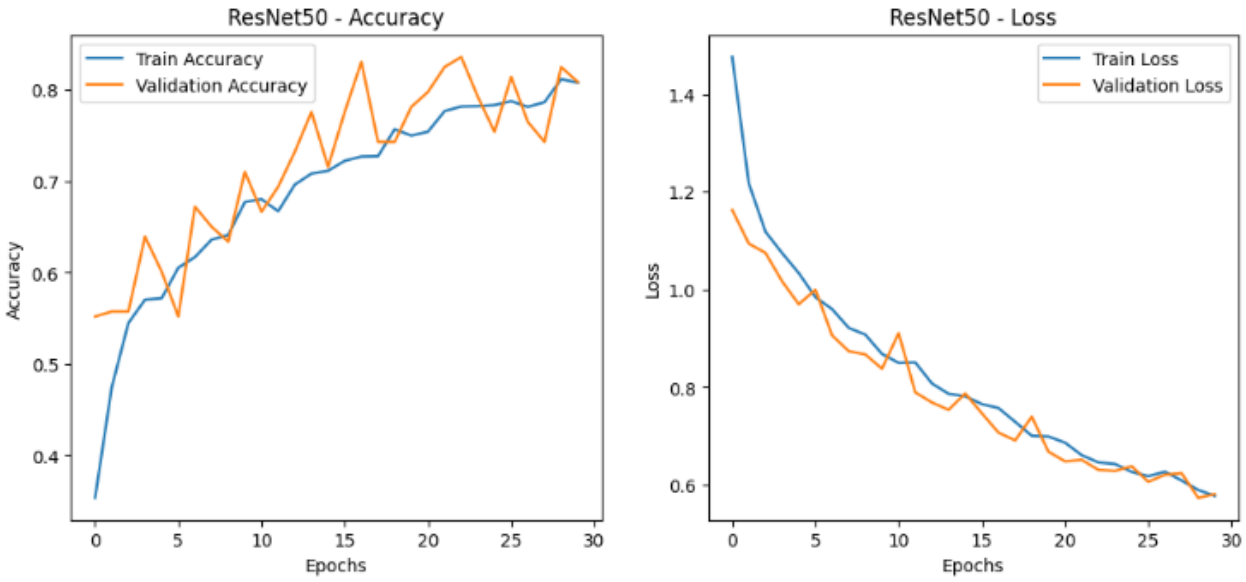

Figure 11 – Accuracy and loss curve of ResNet50.

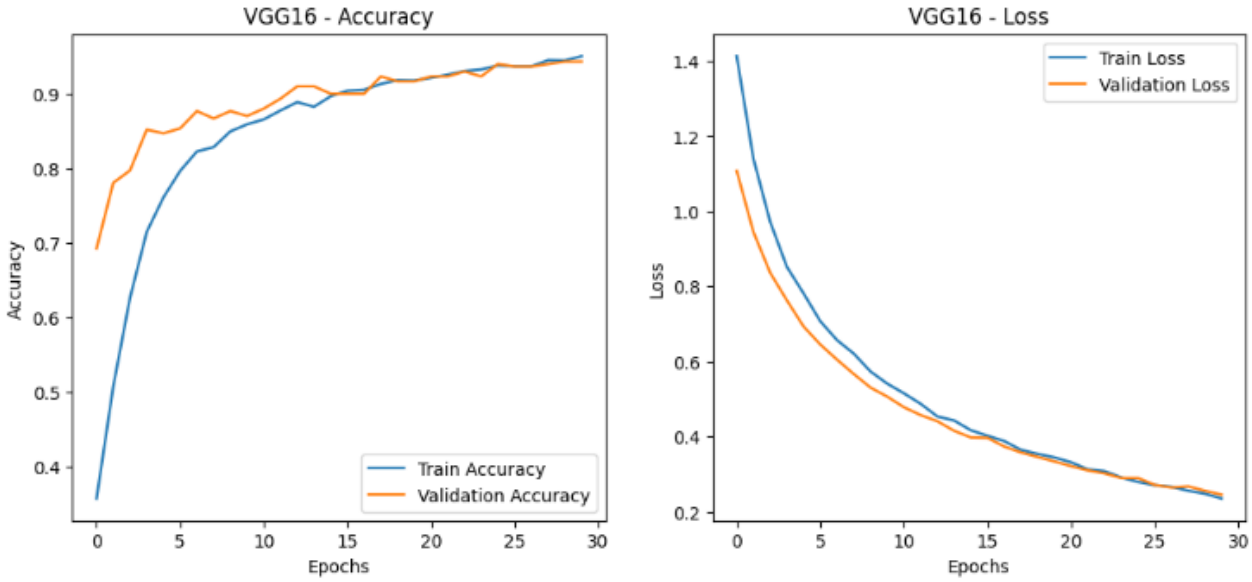

Figure 12 – Accuracy and loss curve of VGG16.

5.2. Limitations and Interpretability

This study has several limitations that must be addressed in future work. First, the dataset is derived from a single publicly available source and may not capture the full spectrum of ECG variation found in clinical practice. This limits generalizability across different patient populations or recording devices. Second, only two CNN architectures were explored. Future research should include a broader range of models, including lightweight architectures for edge deployment and hybrid CNN-RNN frameworks for temporal data.

Additionally, this study did not incorporate interpretability tools. The inclusion of explainable AI (XAI) techniques, such as Grad-CAM, would help clinicians understand which parts of the ECG contributed most to the model's decisions, thereby enhancing trust and facilitating adoption.

One of the key limitations of the current study is the lack of assessment of model robustness to real-world distortions frequently encountered in ECG acquisition. Factors such as poor electrode contact, patient motion, or electrical interference can introduce noise and artifacts that alter ECG morphology. As the dataset used in this study consists of clean, high-resolution ECG images, the performance of the trained models under noisy or artifact-laden conditions remains untested. Future work should incorporate synthetic noise injection, real-world distorted samples, or domain adaptation techniques to evaluate and improve the resilience of CNN models to such perturbations.

5.3. Clinical Applicability

The developed web service prototype demonstrates the feasibility of deploying deep learning models for real-time ECG classification in clinical or remote settings. Potential use cases include:

- Rural clinics lacking cardiologists, where healthcare workers can obtain preliminary diagnoses.

- Emergency departments needing quick triage of incoming ECGs.

- Telemedicine platforms for continuous remote monitoring.

- Educational settings for training medical students and technicians.

In real-world deployment, enhancements such as HTTPS encryption, role-based access, user authentication, and compliance with data protection regulations (e.g., GDPR, HIPAA) will be essential. Integration with hospital information systems and mobile devices can further extend accessibility.

5.4. Statistical Analysis of Multiple Runs

To evaluate the robustness and stability of the proposed VGG16 model, the training and testing pipeline was executed over 5 independent runs using identical hyperparameter settings (batch size = 32, learning rate = 0.0001, epochs = 30). This approach accounts for stochastic variability introduced by random weight initialization, data shuffling, and mini-batch selection.

For each run, performance was measured on the hold-out test set using four primary metrics: accuracy, precision, recall, and F1-score. The mean and standard deviation for each metric are summarized below:

- Accuracy: 0.9758 ± 0.0017
- Precision: 0.9772 ± 0.0021
- Recall: 0.9738 ± 0.0020
- F1-score: 0.9754 ± 0.0021

The narrow standard deviations across all metrics indicate a high degree of consistency in model performance across different runs. This consistency strengthens the reliability of the VGG16 architecture in classifying ECG images under the current dataset and training configuration.

Figure 13 presents a bar plot of the average scores for each metric, accompanied by error bars representing ±1 standard deviation. The consistently high scores and low variance further support the claim that the model generalizes well across the dataset and is not highly sensitive to random initialization or data splits.

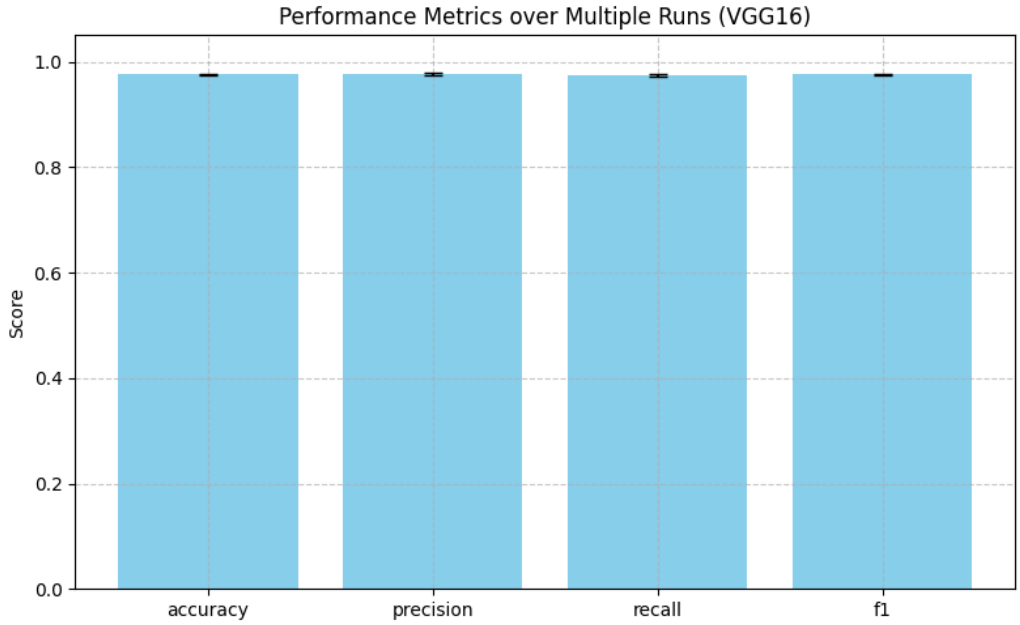

Figure 13 – Performance metrics

5.5. Architectural Improvements and Future Extensions

To further enhance the classification performance and generalization capabilities of the models, several architectural improvements and advanced strategies may be considered in future research:

- Attention mechanisms such as Squeeze-and-Excitation (SE) blocks or Convolutional Block Attention Modules (CBAM) can be integrated into baseline architectures like VGG16 or ResNet50. These modules help the network focus on diagnostically relevant spatial features within ECG images, potentially improving sensitivity to subtle morphological differences.

- Model ensembling could also be explored, where predictions from multiple models (e.g., VGG16, ResNet50, and even lightweight architectures) are combined using soft voting or stacking. Alternatively, extracted features from both CNNs could be aggregated and passed into a meta-classifier such as XGBoost for final prediction. This approach may increase robustness and accuracy.

- Incorporating lightweight or next-generation architectures, such as MobileNetV2 or EfficientNet, may reduce computational load while maintaining high accuracy. These models are especially relevant for edge deployment scenarios, such as portable ECG devices or real-time mobile diagnostics.

Exploring these directions can help optimize both predictive power and practical deployment, thereby increasing the clinical relevance of ECG image classification models.

6. Conclusions

The primary aim of this research was to identify the most effective deep learning approach for classifying standardized ECG images. The dataset originated from a Kaggle repository, featuring scans in uniform brightness and resolution. The use of VGG16 and ResNet50 architectures allowed for a comparative analysis under consistent training conditions. From the initial tests, VGG16 demonstrated higher classification performance relative to ResNet50, prompting further hyperparameter tuning that confirmed VGG16's suitability for ECG classification in this setting.

The process involved a stable training routine of 20 epochs, with systematic variations in batch size and learning rate. The results indicated that hyperparameter adjustments–particularly batch size of 32 and learning rate of 0.0001 or 0.00001–could substantially elevate VGG16's accuracy, reaching up to 98%. Confusion matrices revealed that even subtle morphological distinctions in ECG waveforms were correctly recognized when the model was appropriately tuned.

Implementing this workflow involved minimal data preprocessing efforts, given the dataset's inherent uniformity, yet highlighted the critical influence of hyperparameters on convergence and generalization. A web-based prototype was subsequently developed, integrating the trained model into a Django REST and React framework, thereby demonstrating the feasibility of near-real-time inference.

Overall, these findings illustrate how a well-chosen architecture, combined with measured experimentation on learning rate and batch size, can unlock high levels of accuracy in medical image classification. Future work could investigate more extensive data augmentation, integrate interpretability modules, or broaden the dataset scope to include additional arrhythmias or clinical scenarios. The encouraging performance results also suggest the viability of deploying such models in clinical workflows, particularly in settings that benefit from consistent ECG capture protocols and reliable classification.

Author Contributions

Conceptualization, A. M. and N. Z.; Methodology, A. M.; Software, A. M. and N. Z.; Validation, A. M.; Formal Analysis, A. M.; Investigation, A. M. and N. Z.; Resources, A. M.; Data Curation, N. Z.; Writing – Original Draft Preparation, A. M. and N. Z.; Writing – Review & Editing, A. M. and N. Z.; Visualization, A. M.; Supervision, A. M. and N. Z.; Project Administration, A. M. and N. Z.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. World Health Organization. (2023). *Cardiovascular diseases (CVDs)*. Retrieved from https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)

2. Schots, B. B., Pizarro, C. S., Arends, B. K., Oerlemans, M. I., Ahmetagić, D., van der Harst, P., & van Es, R. (2025). Deep learning for electrocardiogram interpretation: Bench to bedside. *European Journal of Clinical Investigation*, *55*, e70002.

3. Mienye, I. D., & Swart, T. G. (2024). A comprehensive review of deep learning: Architectures, recent advances, and applications. *Information*, *15*(12), 755.

4. Thakur, G. K., Thakur, A., Kulkarni, S., Khan, N., & Khan, S. (2024). Deep learning approaches for medical image analysis and diagnosis. *Cureus*, *16*(5).

5. Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

6. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778).

7. Li, S., Zhao, P., Zhang, H., Sun, X., Wu, H., Jiao, D., ... & Wang, D. (2024). Surge phenomenon in optimal learning rate and batch size scaling. *arXiv preprint arXiv:2405.14578*.

8. Gunning, D. (2017). Explainable Artificial Intelligence (XAI). *Defense Advanced Research Projects Agency (DARPA), Program Information Release*.

9. Acharya, U. R., Fujita, H., Sudarshan, V. K., Oh, S. L., Adam, M., Koh, J. E., ... & San Tan, R. (2016). Automated detection and localization of myocardial infarction using electrocardiogram: a comparative study of different leads. *Knowledge-Based Systems*, *99*, 146-156.

10. Kachuee, M., Fazeli, S., & Sarrafzadeh, M. (2018, June). Ecg heartbeat classification: A deep transferable representation. In *2018 IEEE international conference on healthcare informatics (ICHI)* (pp. 443-444). IEEE.

11. Luz, E. J. D. S., Schwartz, W. R., Cámara-Chávez, G., & Menotti, D. (2016). ECG-based heartbeat classification for arrhythmia detection: A survey. *Computer methods and programs in biomedicine*, *127*, 144-164.

12. Yildirim, Ö. (2018). A novel wavelet sequence based on deep bidirectional LSTM network model for ECG signal classification. *Computers in biology and medicine*, *96*, 189-202.

13. Rajpurkar, P., Hannun, A. Y., Haghpanahi, M., Bourn, C., & Ng, A. Y. (2017). Cardiologist-level arrhythmia detection with convolutional neural networks. *arXiv preprint arXiv:1707.01836*.

14. https://www.kaggle.com/datasets/evilspirit05/ecg-analysis/data

15. Smith, L. N. (2017, March). Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)* (pp. 464-472). IEEE.

16. 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 (TOSEM)*, *31*(3), 1-40.

17. Nematzadeh, S., Kiani, F., Torkamanian-Afshar, M., & Aydin, N. (2022). Tuning hyperparameters of machine learning algorithms and deep neural networks using metaheuristics: A bioinformatics study on biomedical and biological cases. *Computational biology and chemistry*, *97*, 107619.

18. Bartz, E., Bartz-Beielstein, T., Zaefferer, M., & Mersmann, O. (2023). *Hyperparameter tuning for machine and deep learning with R: A practical guide* (p. 323). Springer Nature.

19. Fang, A., Kornblith, S., & Schmidt, L. (2023). Does progress on ImageNet transfer to real-world datasets?. *Advances in Neural Information Processing Systems*, *36*, 25050-25080.

20. Kim, H. E., Cosa-Linan, A., Santhanam, N., Jannesari, M., Maros, M. E., & Ganslandt, T. (2022). Transfer learning for medical image classification: a literature review. *BMC medical imaging*, *22*(1), 69.

21. Pant, P., Rajawat, A. S., Goyal, S. B., Bedi, P., Verma, C., Raboaca, M. S., & Enescu, F. M. (2022). Authentication and authorization in modern web apps for data security using Nodejs and role of dark web. *Procedia Computer Science*, *215*, 781-790.

22. de Almeida, M. G., & Canedo, E. D. (2022). Authentication and authorization in microservices architecture: A systematic literature review. *Applied Sciences*, *12*(6), 3023.

Information about authors

Alfarabi Mazhit, Second-year master's student in Software Engineering at Kazakh-British Technical University (Almaty, Kazakhstan, a-mazhit@mail.ru), ORCID: 0009-0003-7308-2131

Nazgul Zakariyanova, Senior Lecturer at Al-Farabi Kazakh National University (Almaty, Kazakhstan, znazb11@gmail.com), ORCID: 0009-0000-0842-5394

Submission received: 22 April, 2025.
Revised: 27 May, 2025.
Accepted: 27 May, 2025.