Optimizing Deep Learning Architecture for ST Elevation Detection in Myocardial Infarction Diagnosis

Walaa H. Elashmawi[a,b], Sohayla I. Hamed[c], Ahmed Ali[d,e,*]

[a] *Faculty of Computers & Informatics, Suez Canal University, Ismailia, Egypt*
[b] *Faculty of Computer Science, Misr International University, Cairo, Egypt*
[c] *Computer Engineering, Ain Shams University, Cairo, Egypt*
[d] *College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Alkharj Saudi Arabia*
[e] *Higher Future Institute for Specialized Technological Studies, Cairo, Egypt*

Abstract

Myocardial infarction (MI) is the ischemic death of myocardial tissue, often causing acute coronary syndrome (ACS). ST-elevation myocardial infarction (STEMI), diagnosed by ECG elevation, is the most fatal heart attack and increases heart arrhythmia and heart failure risk. Deep learning (DL) techniques are used in medical diagnoses to categorize electrocardiogram (ECG) delineations into feature maps to detect disease-causing abnormalities. The training of features depends on the diagnosis, and severe episodes of ischemic heart disease and STEMI increase the risk of physician error. This chapter presents a diagnostic DL model with an architecture modulated using Particle Swarm Optimization (PSO), a nature-inspired optimization, to improve the identification of unhealthy classes in the data. The process done to automate the selection of the optimal architecture is called Neural Architecture Search (NAS). Various DL architectures, including ConvNetQuake, channel-based, and transfer-based ResNet50, were evaluated, with the best-performing model selected for final testing. Because memory profiling is essential to ensure optimization without the risk of out-of-memory (OOM) errors, the resource consumption of each model is reported. The results demonstrated that channel-split PSO ResNet50 achieved the maximum results with a precision of 88%. Additionally, dynamic memory profiling recorded an average 72% processing power usage. The results provide in-

*Corresponding author email: a.abdalrahman@psau.edu.sa

sights into enhancing cardiovascular health monitoring and diagnosis thus emphasizing the link between network architecture and computational resource consumption.

Keywords: ECG, Deep Learning, Convolution Neural Network (CNN), Thread Consumption, Graphical Processing Unit (GPU), Signal Processing, Particle Swarm Optimization (PSO), Neural Architecture Search (NAS), S-T Elevation Myocardial Infarction (STEMI), Ischemia

---

1. Introduction

The heart goes through many stages of electrical activation as it pumps blood to the entire body. Historically, electrocardiograms were created as galvanometers to measure the heart's electrical activation at the body's extremities. Later innovations applied multiple electrodes, or leads, placed throughout the body. The acquired data is plotted as waves, called electrocardiograms (ECGs), showing the difference in voltage potential by time. ECGs are widely used for efficient and accurate monitoring of cardiac health. They are used by various healthcare professionals and caretakers, including cardiologists, physicians, emergency-room paramedics, and nurses.

The accuracy of physician inspection by ECG is approximately 54% and increases to 67% with extra training [1]. Computerized ECG interpretation proved an accuracy of 88.0% for as long as 20 years, proving it is a long-standing method of cardiac health assessment that will continue in the future [2]. Professionals use manual methods to diagnose heart problems, even life-threatening ones such as Myocardial Infarction (or colloquially known as "heart attacks"), from ECG signals. The main problem with manual diagnosis of heart attacks, like other time-series data, is the difficulty of detecting and categorizing different waveforms and morphology in the signal. Another less severe difficulty is the understanding of certain "hidden" patterns. For a human, a proper diagnosis is extensively time-consuming and prone to errors. Consider life-threatening conditions that account for approximately one-third of all deaths around the globe. Therefore, obtaining the best categorization and diagnosis from ECGs is paramount.

ECG features are divided into two main types, waves, and intervals [3]. Waves are segments of the ECG that represent the heart muscle circulatory cycle. Intervals represent locations between each of the waves. Five main waves exist (P, Q, R, S, T) and are shown in Figure 2. P wave reflects the

spread of electrical activation (depolarization) through the heart's right and left atria as shown in Figure 1. The T wave represents the heart muscle relaxation, known as ventricular repolarization. PR interval lies between the beginning of the P wave and the beginning of the Q wave (i.e., initial depolarization or contraction of the ventricle, the heart is about to "pump"). The QRS complex represents the total depolarization of the ventricles ("the pump is working"). The QT interval starts at the beginning of the Q wave and ends at the T wave. It represents the entire period of contraction and relaxation of the ventricles (i.e., the pump starts at the bottom chambers, the ventricles, where the pressure is highest, and then fluid flows up to the top chambers). Finally, the ST segment represents the interval between the end of the S wave (ventricular depolarization) and the beginning of the T wave (ventricular repolarization). It represents the period when the ventricles are depolarized ("contracted, will relax").

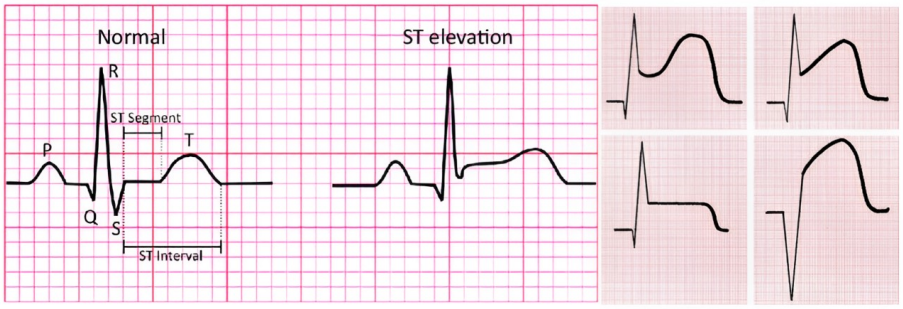


Figure 1: Heart muscle CT scan showing coronary vessels and coronary circulation

Due to its possibly missable patterns of risk factors ([4]), diagnosing myocardial infarction (MI), usually known as a heart attack, often relies on the use of an electrocardiogram (ECG or EKG). These miss-able signs are usually ones of acute coronary syndrome (ACS), which, in approximately 80% of patients, will lead to MI. Physicians usually look for abnormal patterns in the ECG waves or intervals detected by electrodes in multiple locations, called leads. The values of the leads are the members in the courtroom of body ionization. Basic acute patterns are required for confirmatory diagnosis of MI, regardless of the type of atypical activity in the ECG itself. The most common cases constituting MI data of cardiac tissue occlusion (blockage) and necrosis (death) are:

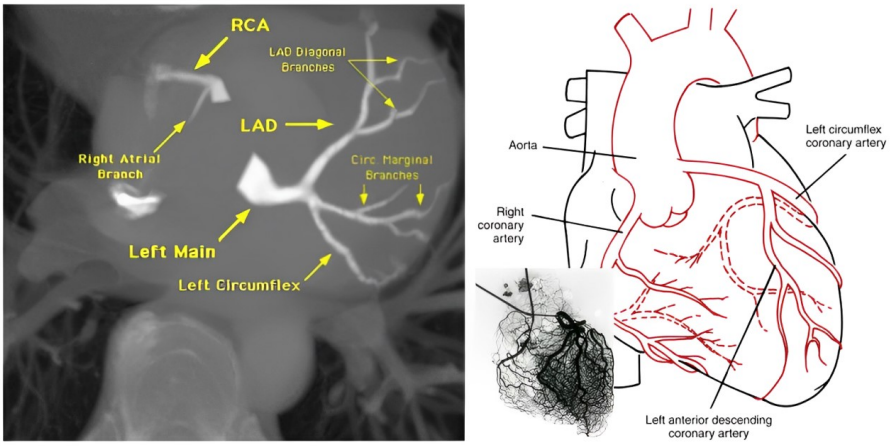


Figure 2: Normal and Anterior ST-segment Elevation pattern of ECG signal.Variations of ST elevation.

- **Anterior STEMI:** Anterior, meaning nearer to the front, ST-segment elevation causes the majority of deaths if not treated urgently. This type of STEMI is caused by the blockage of the left anterior descending artery (LAD). This type of STEMI is not straightforward to identify due to the multitude of ECG patterns and leads required to be investigated for successful diagnosis.

- **Inferior STEMI:** Inferior, meaning lower in position, ST-segment elevation accounts for 40-50% of all MI cases. In most cases, it involves blockage of the right coronary artery (RCA), while in rarer cases, it may involve the circumflex artery (CX). The former blockage potential ECG pattern was demonstrated in Figure 2.

- **Posterior STEMI:** Posterior, meaning nearer to the hind, ST elevation involves the blocking of the posterior descending artery (PDA) or posterior interventricular artery (PIA). This vessel branches from the RCA, meaning that inferior STEMI can cause this type of blockage and further result in posterior STEMI.

The previous types of STEMI are all caused by blockages to certain coronary (of the heart) blood vessels. If undetected, each blockage will cause a cessation of oxygen and blood flow to the myocardium or middle heart muscle. If this repeats, it can cause irreversible muscle damage and sudden death.

To minimize life-threatening consequences, it is crucial to gather clean patient ECGs to identify case symptoms and features. Obtaining those features from patient data requires a multi-step process for extracting, transforming and loading high-quality ECG recordings [5]:

1. **Baseline ECG:** A baseline ECG is taken as an initial evaluation of the patient suspected to have MI or has had a heart attack. This step aims to detect pre-existing anomalies that could influence subsequent findings' validity.
2. **Serial ECGs:** Serial ECGs involve taking multiple ECGs at different time points, usually at regular intervals, to monitor changes in patient ECG data. This step plays a vital role in facilitating continuous monitoring and disease progression tracking, particularly in situations with a high risk of disease exacerbation.
3. **ST-Segment Changes:** This is the key feature observed on an ECG to diagnose MI. There are several patterns to ST changes, such as elevations and depressions (i.e., the diagnosis of STEMI usually focuses on elevations).

Additionally, some features are used to confirm the diagnosis of MI, such as T-wave changes and Q-waves in case of significant damage to the heart muscle. Figure 3 summarize an example of serial monitoring of STEMI.

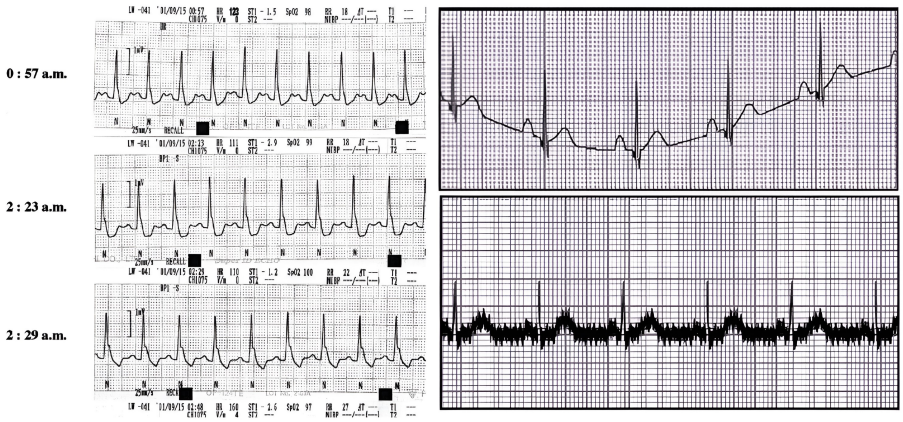


Figure 3: Serial ECGs of a patient throughout the day.Baseline wander and power line interference of an ECG.

Achieving an accurate diagnosis is contingent upon the acquisition of high-quality ECG recordings, free from artifacts and electrical noise. Any

bio-signal acquisition falls under many interferences from the external environment that affect the quality and accuracy of that signal. ECGs are no exception. According to [6], there are many types of interference:

1. ***Muscle Artifacts:*** Muscle artifacts occur as high-frequency noise on the ECG, often due to the movement of muscles near the electrodes. They can make actual cardiac signals indiscernible.
2. ***Baseline Drift:*** Baseline drift is a slow, gradual shift of the baseline of the ECG recording over time, as shown in Figure 3. It is usually caused by poor electrode contact and can mislead the identification of abnormalities.
3. ***Power Line Interference:*** Power line interference occurs as harmonic spikes in the ECG, as shown in Figure 3, often due to electrical noise from nearby power sources.

To minimize the influence of artifacts and ensure accurate signal interpretation, appropriate filtering techniques such as band-pass or band-stop filters are recommended during ECG acquisition. However, even with meticulous physician efforts in diagnosing ST-elevation myocardial infarction (STEMI), misdiagnosis remains a potential clinical challenge. In light of these considerations, the primary objective of this paper is as follow:

- Explore different architectures for the diagnosis of STEMIs.
- Propose a cardiologist-level optimized deep learning model to detect unhealthy patients from stochastic batches.

This paper is organized as follows: Section 2 delves into the latest advancements in MI diagnosis models, exploring how researchers are leveraging technology to improve diagnostic accuracy and efficiency. Also, this section highlights the difference between major architectures. Section 3 refers to the neural network designs that are used to process the ECGs and provide the required diagnosis. Section 4 outlines the datasets utilized with the proposed architecture. Section 5 focuses on this paper's methodology and major optimization technique. Section 6 compares the results attained among the different architectures and their optimized versions. Section 7 summarizes some limitations faced while working on the models discussed in this paper. Finally, Section 8 wraps up all the conclusive details of this study and proposes other ways to improve it.

2. Background

Accurate diagnosis of MI is paramount for improving patient outcomes. This section explores the evolving landscape of MI diagnosis models, highlighting how researchers harness technology's power to achieve faster and more precise detection. We will delve into recent advancements in artificial intelligence (AI), specifically machine learning (ML) and deep learning (DL) approaches used to analyze patient data including, ECG signals, vectorcardiogram (VCG) signals and biomarkers.

2.1. Categorization Based Models

Categorization based models make the most use of categorizing different modes of patient data into classes, thus presenting mainly a healthy and a unhealthy class. Class-distinguishing features are drawn using statistical methods, mathematical transforms, ML techniques or a combination of either. While they are not highlighted in the works cited in this subsection, other intermediate classes can be detected as well.

Building upon the work of [7], several feature extraction techniques such as: Local Binary Pattern (LBP), Higher Order Spectra (HOS), and Discrete Wavelet Transform (DWT), were leveraged into a method that provides an adaptable, more personalized, real-time healthcare solution. In addition, optimization heuristics were capitalized alongside the mainly statistical techniques mentioned.

To address potential limitations in existing diagnostic machine learning methods, the authors of [8] proposed a novel approach that leverages dimensionality reduction using Linear Discriminant Analysis (LDA) in conjunction with hybrid feature selection algorithms. Their study aimed to enhance the decision-making capabilities of ML algorithms, specifically Support Vector Machines (SVM) and Decision Trees (DT). The effectiveness of their approach was further evaluated using Artificial Neural Networks (ANN). The results demonstrated significant improvements across various performance metrics, including F1 score, specificity, and sensitivity, in all test cases.

In comparison to other literature works that usually used morphological images or bio-signals, [9] proposed the diagnosis of Acute Myocardial Infarction (AMI) by identifying risk factor biomarkers from ECG metadata. The results proposed a five feature model that included cardiac troponin I, HDL cholesterol, HbA1c, anion gap, and albumin, which achieved comparable results to other models with more features.

[10] first subject ECGs to Flexible Analytic Wavelet Transform (FAWT) to output ECG sub-bands. Each reconstructed sub-band FAWT coefficient is input into a sample entropy framework which calculates that entropy of that particular sub-band. Finally, a classification is yielded after features are subjected to conventional ML algorithms.

[11] proposes a method that uses both temporal (intervals between beats) and morphological (beat patterns) to categorize ECGs. Because this analysis requires complex, multi-dimensional relationships that are modeled by support vectors, multiple SVMs are used to generate feature descriptors. Each feature descriptor has a corresponding machine, where the feature descriptors were divided among morphological and temporal features. Higher accuracy and performance of heart beat categorization are advantages of this method.

According to [12] and [13], classification and regression-based architectures are trialed on complimentary diagnostic wave and heart morphological features, ECGs and VCGs, along with feed-forward networks. In addition, features like chest pain or other clinical features may be incorporated into tests. At first, features are extracted via windows or retraced via transforms. Then, a neural network with numerous dense blocks, such as VGG-16, is used to refine the traced features. This feed-forward technique serves the classification problem. After classification, machine learning techniques such as logistic regression (LR) and decision tree (DT) are used to calculate the probability of a new case being of a certain class, which serves the regression problem. The main advantage of their proposed architecture is its applicability in clinical settings despite its potentially difficult code.

2.2. Deep Learning Based Models

Deep Learning (DL) based models are those that exclusively use neural networks to draw feature maps for each class. The loading, transforming and pre-processing of data can utilize no neural networks, but the classification and optimization processes primarily use neural networks. Unlike last subsection, this subsection outlines studies that have relied on ANNs rather than statistical categorization of class-distinguishing features.

[14] compared the performance of processed ECGs using CNN-Dense with different optimizers such as Adadelta, Adam and Nadam, where each optimizer performance is ranked by the number of data used to compute the gradient descent. The system processes one-dimensional input ECG signals and passes them through multiple layers for diagnosis. The proposed CNN architecture includes layers for input, convolution, max-pooling, dropout,

dense, and Softmax.During testing, different neural network architectures like EchoState, ECG-Net, CNN-Adadelta, Block-based neural networks, FC-ANN, VGGNet, GoogleLeNet, AlexNet, ANN, and RNN are presented.

The authors in [15] employed deep transferable representations to accurately categorize heartbeats for monitoring cardiovascular health. They proposed a method using deep convolutional neural networks to classify arrhythmias according to the Association for the Advancement of Medical Instrumentation (AAMI) EC57 standard and transfer this knowledge to the classification of MI. The methodology involves pre-processing ECG signals, training a classifier, and utilizing learned representations for MI prediction. Their study demonstrated competitive accuracy results compared to other existing methods and showed effective separation of data points for both classifying arrhythmia and MI (thus confirming the medical claim of the correlation between both cardiac conditions).

Applying deep learning techniques along with entropy, specifically artificial neural networks with one-dimensional convolutional layers for signal processing tasks, is employed by [16]. To mitigate the vanishing gradient problem and enhance model capacity, the convolutional layers within the proposed neural networks incorporate residual connections, as described in the cited work. The networks were trained using the Adam optimizer with a learning rate adjustment during training to prevent overfitting. Entropy-based features such as Shannon entropy, Approximate entropy, and Sample entropy improved the accuracy of heart disease classification. The main investigation in their study is the integration of SincNet layers for low-level feature extraction in future work. [17] proposed an architecture that classifies heart attacks without machine learning steps. Unlike [16], focal loss was used to handle data imbalances and constitute focus in difficult classes.

[18] focuses on short segments of ECG signals collected through bio-metric recognition. The study includes an analysis of identification and verification performance using bio-metric recognition processes, with scenarios for one-to-many matching (identification) and one-to-one matching (verification). The main aim of the method in [18] is to improve recognition performance over long periods and under different signal changes. Various deep learning models were tested for biometric recognition, showing different levels of accuracy based on the segmentation method and session length of the signal.

While a different application of ConvNetQuake architecture is presented by [19]. This architecture was originally designed to identify earthquakes. Certain ECG leads are input into a multi-channel neural network comprising

convolutional layers, flattening layers, and a fully connected layer.

Deep learning models, despite their impressive capabilities, can often face challenges in optimization due to complex architectures and high dimensionality. This is where natural-inspired meta-heuristic algorithms come in. These algorithms mimic natural processes, such as swarm intelligence (like ants or bees) or evolutionary selection, to guide the training process of deep learning models. By drawing inspiration from these natural phenomena, meta-heuristics can effectively search for optimal solutions within the vast parameter space of deep learning models. The authors in [20] propose a neural network whose architecture and hyperparameters can be optimized using Marine Predators Algorithm. After reliable ECG features are extracted, a random solution is initialized, and then it is trained with a neural network to update the prey position. The diagnosis is successful if the network converges; if not, a solution is re-established, and then another block of the CNN is trained. In addition, a search for the best learning rate parameter value is implemented before a neural network can be improved [21]. Some improvements can be enabled by the python compute engine CUDA which works by parallelizing an algorithm or a neural network using multiple GPUs.

The existing literature has mostly focused on diagnosing myocardial infarction through ECG categorization or image-based deep-learning methods. Even though many of the architectures in literature perform well in terms of accuracy, precision, and efficiency, they are mostly too complicated to be featured in portable systems, such as traditional web applications. In addition, the models featured in this section involve complex designs that heavily rely on feature engineering and reduction, such as in [12].

In the current research, almost no swarm optimization techniques were tested with traditional, benchmarked neural networks.

In contrast, the proposed model introduces a less-reliant feature-engineering version of optimized traditional neural networks that are lightweight enough to be embedded into the backend of portable applications. The model seeks to optimize traditional deep neural networks with stochastic sampled, feature-reduced ECGs. We aim to design an optimized model that can embedded along with models of other data modalities in a web app, for example, rather than a singular non-optimized model in a particular application (web or desktop or medical server). Overall, the results for this architecture are promising enough, especially with unhealthy patients, to make way for later or heavier implementations with several rounds of feature engineering.

2.3. Optimizing the Working of Deep Learning Model

The optimization of deep learning architectures is necessary for several reasons including reducing latency, improving performance without increasing computational costs, improving scalability, memory efficiency and emphasizing generalization of classification or diagnosis models. Recent research done to develop optimization techniques for deep learning models also involve the automation of neural network architecture selection or Neural Network Architecture Search (NAS). This subsection gives a few examples of applications with optimization techniques done to automate neural network architectures.

[22] presents an open-source system called OpenNAS which works by integrating two major swarm intelligence (SI) techniques, Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) to generate a neural network for image classification. Major milestones presented include the achievement of improved model accuracy through the evaluation of different model architectures, which would otherwise be burdensome to a novice user especially when choosing the correct hyper-parameters. OpenNAS takes in the dataset as images and generates a neural network using an AutoKeras or an SI approach. Stacking ensembles can also be used to combine meta-heuristics of models generated in this approach. By the end, the performance is evaluated via fine-tuned, pre-trained models to check for the best model setup.

[23] proposes a low-latency search strategy based on PSO to apply the searched network as a backbone for object detection tasks. During the search process, a multi-objective fitness function is used to balance between network performance and resource consumption. To evaluate particle performance, weight sharing and dynamic early stopping are proposed. Finally, the fine-tuned globally optimal particle is used to build an object detection model that balances between accuracy and efficiency without considerable trade-offs.

[24] builds functional brain network (FBN) classification based on neural networks without labor-intensive, unreliable hand-crafted architectures. This method includes three phases: the individual expression phase, the individual evaluation phase and the individual update phase. In the particle swarm, each individual is at first the neural architecture. But as the methodology progresses, brain network topological feature vectors are constructed and by the end a fitness measure is applied to determine the historical optimum of each individual and the global optimum of the population, in order to update all individuals until an end condition is met. Experiments on benchmark

datasets show that CNNs searched achieve better classification results than other hand-crafted CNNs.
In the current research, not much experiments were made on medical or patient datasets, only benchmark general datasets. In addition, the devices used for current research do not consider cases of severe environment limitations. This is crucial to consider especially in an economy that, at best, prioritizes health over IT while at the same time applying state-of-the-art technology. The results are arguably reliable to provide insight into the automation of complex neural networks and later on provide hindsight into their advancement with regards to backward compatibility, especially under the tightest of circumstances.

3. Transfer Learning: AlexNet, ResNet, ConvNetQuake

Transfer learning is a technique that applies a trained model intended for a task as a basis for another related task. This approach employs the knowledge obtained from the source task to improve the learning or performance of the target task. In this work, the main aim is to classify and detect patterns of MI in ECGs, thus leading to a diagnosis, so image and wave-based neural networks will be used to differentiate between the required classes (healthy or unhealthy). Transfer learning neural networks applied here are AlexNet, which was primarily used for the ImageNet dataset comprising of several type of animals, ResNet, which was used to shortcut AlexNet and ConvnetQuake, which was used to detect changes in seismic waves. The following section explains how each of the neural networks was built and how each was applied to MI diagnosis.

3.1. AlexNet

The authors [25] trained a neural network to classify 1.2 million images from the ImageNet LSVRC-2010 contest into 1000 categories, achieving top-1 and top-5 error rates of 37.5% and 17%, respectively, surpassing previous models. AlexNet consists of approximately 600,000 neurons and 60 million parameters, optimized with non-saturating neurons and multi-CPU/GPU implementation.

3.1.1. Key Architecture of AlexNet

- **Layers:** AlexNet has 8 layers, including 5 convolutional layers, max-pooling layers, and 3 fully connected layers, ending with a softmax

classifier. Initial experiments featured 15 classes but were later refined to two (healthy/unhealthy) due to learning rate limitations and saturation risks. Feature graphs were validated post-convolution, and label generation was tested after saving the architecture. The first layer had large receptive fields with a large stride, decreasing in later layers. Max pooling used a s-pixel pool size ($s = 2$) over a $z \times z$ neighborhood ($z = 3$). Overlapping pools reduced error rates and overfitting.

- **Activation Function Non-linearity and Saturation:** Saturation occurs when network outputs stagnate, leading to vanishing gradients. AlexNet used the Rectified Linear Unit (ReLU), which accelerated training and mitigated this problem. ReLU's non-linearity saturates slower than functions like sigmoid or tanh, improving training efficiency.

- **Normalization:** Unlike sigmoid or tanh-based models, ReLU in AlexNet does not require input normalization. As long as some training samples produce positive ReLU inputs, learning progresses. Local response normalization (LRN) was used to normalize neuron activations. Batch normalization was not employed but could stabilize and accelerate training.

- **Reducing Overfitting and Dropout:** To prevent overfitting, AlexNet employed dropout in fully connected layers, randomly ignoring neurons during training. Shuffling ECG image batches further regularized the model. Simple random shuffling proved more stable and effective, while data augmentations like rotation, color shuffling, and horizontal flipping did not improve performance.

3.1.2. Similar Applications in Literature

[26] used ECG signal processing techniques to detect arrhythmia classes, applying Continuous Wavelet Transform (CWT) for feature extraction. ECG scalograms of varying lengths (1s, 2s, and 3s) were input into AlexNet, with 3-second signals yielding the best results. The proposed model in section 5 used a 10,000-millisecond window for Hermite transform extraction, aligning with this study.

[27] merged three datasets to compile over 5,000 multi-lead ECG recordings. Z-score normalization ensured consistent scaling, and CWT extracted features from 15-second scalograms, input into a modified AlexNet. This study showed AlexNet's adaptability to continuous wave data, outperforming models like DWT Random Forest and SVM.

The proposed model, described in sections 5 and 6, balances model complexity and performance. AlexNet converges quickly but has lower accuracy across orthogonal leads (standard, chest, Frank).

[28] explored feature extraction using Fractional Discrete Cosine Transform (FDCT), Radon Wavelet Transform (RWT), and Fractional Wavelet Transform (FWT). Red Fox Optimization was used for feature selection, and an improved AlexNet (i-AlexNet) outperformed BiLSTM and FDDN. In contrast, the proposed model applies Particle Swarm Optimization (PSO) to optimize model complexity, ensuring stable classification without oscillations.

3.2. ResNet

The authors in [29] addressed several challenges faced by AlexNet and other previous neural networks, mainly that dense networks are hard to train and that vanishing gradients may continue to appear. The main aim is to design a residual learning framework to ease the training of networks that are deeper than older ones, thus improving image recognition and computer vision tasks. The standard experiments were carried out on the ImageNet dataset where the residual network can contain up to 152 layers, all while having a lower complexity. Standard experiments have shown a 3.57% error rate on the test dataset(much lower than the 17% error rate of AlexNet).

3.2.1. Key Architecture of ResNet

Prior networks to ResNet, notably AlexNet, faced a crucial, irreversible issue of decreasing metrics and worsening performance as a very deep AlexNet for example quickly converges and saturates after many reruns and predictions. A rerun AlexNet degrades over time and as it becomes denser. This degradation has adverse effects on most aspects of a good application aspects of a network, such as usability, reliability and longevity of that network. Degradation is not caused by overfitting, so adding more layers, randomizing parameters, or even shuffling batches of input only increases training error and does not solve the issue. To combat this, it became necessary to design a very deep neural network that allows for possible varieties in parameter dimensions and feature sizes per layer without risking degradation. In this subsection, we outline precisely how the architecture of ResNet accommodated this work's proposed model. In section 6, ResNet50 was used to train and test this paper's proposed model among different data channels.

- **Layers:** ResNet consists of convolutional layers followed by parameter-increasing shortcut residual blocks, where each residual block has at

least two convolutional layers. Standard experiments used between 34 to 152 layers, but a maximum of 50 pre-trained layers was more convenient for this paper. Max pooling is applied before the final fully connected (dense) layer to determine the number of feature maps.

- **Residual Blocks:** In plain networks, convolutional layers must have the same number of filters for the same output and feature map size, and the number of filters should be doubled when the feature map size is halved. Converting a residual network to a plain network requires down-sampling through convolutional layers with a lower stride. Residual networks use identity shortcuts when input and output dimensions match, allowing deep computer vision tasks without extra computations and parameters. When dimensions increase, residual functions can be projected by smaller dimension functions or shortcut with identity mapping and padded with zero entries [29]. Element-wise addition between input, bias, and residual function remains negligible, enabling ResNet to bypass connections and mitigate issues from high parameter dimensions. Figure 4 illustrates residual blocks in ResNet-50.

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 7, 7, 2048)	23,587,712
flatten_1 (Flatten)	(None, 100352)	0
dense_3 (Dense)	(None, 1024)	102,761,472
dense_4 (Dense)	(None, 512)	524,800
dense_5 (Dense)	(None, 2)	1,026

Total params: 126,875,010 (483.99 MB)
Trainable params: 103,287,298 (394.01 MB)
Non-trainable params: 23,587,712 (89.98 MB)

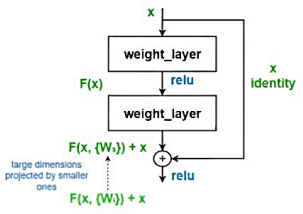


Figure 4: A summary of ResNet-50 parameters and the working of a residual block

- **Deeper Networks and Gradient Flow:** ResNet projects large dimensions into smaller ones, making it scalable with versions like ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152. ResNet-50 and beyond use a bottleneck architecture to reduce computational complexity, with each bottleneck block containing a 3×3 convolution layer between two 1×1 convolution layers. Vanishing and exploding gradients are minimized as ResNet maintains gradient flow during forward and backward passes (backpropagation). The results in section 6 include a batch normalization layer after each convolutional layer, stabilizing training by normalizing inputs before passing through each layer.

Normalized feature maps improve performance after each convolutional layer.

3.2.2. Similar Applications in Literature

The authors in [30] used more advanced denoising techniques such as wavelet transforms, down-sampling to 200Hz, and QRS detection via the Pan-Tompkins algorithm. A 13-layer ML-ResNet network was employed with a single lead feature branch made up of 3 residual blocks for MI detection and localization. To attain better spatial information representation, the single lead network learns representative features at different levels. After the initial single-lead feature experiment, multi-lead feature fusion was performed to provide a more comprehensive diagnosis in comparison to the initial experiment. Some improvements proposed in [30] include mass data utilization, inter-patient variability models, and feature expansion.

The experiments in section 6 considered some of the improvements in [30]. For instance, ResNet experiments featured a variety of leads to ensure inter-patient variability and feature expansion. Different experiments with different leads provide insights into STEMI cases(anterior, inferior, posterior). In addition, rather than a simpler ResNet-13, ResNet-50 with a bottleneck architecture was used to make use of better feature detection and classification.

In [31], an exclusive hybrid model that combines a modified ResNet for feature extraction and a novel-trained Vision Transformer(ViT) model is presented. The main innovation in this paper is the focus on the improvements of the ViT architecture. A multi-branch ViT network was used with a channel attention mechanism. This slim model enhances the richness of feature and information learning with a reduced complexity. The main disadvantage of this model is that it may overlook crucial local details.

3.3. ConvNetQuake

The main advantage of ConvNetQuake[32] is that the orders of magnitude are much faster than other feature-engineering reliant methods. The application of ConvNetQuake is not limited to earthquake detection. As explained in [19], ConvNetQuake can be used as a deep learning technique to diagnose abnormalities in ECG data (continuous wave data similar to seismic earthquake data) to yield good results without the heavy reliance on feature-engineering and feature-reduction.

3.3.1. Key Architecture of ConvNetQuake

Unlike previous transfer learning neural networks, ConvNetQuake does not provide new innovations to the technologies used in neural network design. However, in terms of MI diagnosis, experiments were carried out to provide a cardiologist level training [19].The core limitation of [19] is that no comprehensive tests were carried out. In this subsection, we outline the major architecture of ConvNetQuake to accommodate the diagnosis of MI.

- **Preparation:** The primary distinction between ConvNetQuake and other neural networks is in the necessity for meticulous preparation of input tensors for the network. Our experiments used random batches of min-max normalized ECGs regardless of their corresponding classes. This was done so that most of the classification task is left to the neural network (and later to the swarm optimization as in section 5 and 6).

- **Layers:** ConvNetQuake used a series of blocks where each block is composed of a series of convolutional layers followed by flattening layers, a sigmoid layer and another series of batch normalization layers. During backpropagation, a ReLU activation is used to call each pair of convolution and batch normalization layers . At the end, the feature maps are reshaped, flattened and a sigmoid classification layer is used. Note that ConvNetQuake does not prioritize the sensibility of the classification or would've otherwise used a softmax classification layer. This difference is ignorable because there are only 2 classes, either a healthy or an unhealthy patient.

3.3.2. Similar Applications in Literature

ConvNetQauke is a relatively novel model which was mainly used to leverage a better, more cardiologically accurate diagnosis [19], as in section 2. To review ConvNetQuake, [33] challenged conventional accuracy and sensitivity in MI diagnosis. Specifically, ConvNetQuake is among many medically applied models with a low interpretability, "black box" nature even when using area under curve (AUC) rather than accuracy. There is a lack of exploratory studies focusing on the interpretability of training and testing complex time series ECG data, which risks failed understanding of this medically critical area.

Finally, transfer learning techniques enable efficient use and reuse of pretrained models and accelerate model development, which makes them powerful and applicable ML and DL workflows.

4. PTB ECG Diagnositic Database

Delving deeper into the realm of data resources for MI diagnosis models, we now explore the Physikalisch-Technische Bundesanstalt (PTB) Diagnostic ECG Database [34] [35] [36]. This comprehensive dataset, meticulously curated by PhysioNet, provides a cornerstone for researchers developing and evaluating automated MI detection algorithms.

A collection of electrocardiogram (ECG) data from 290 subjects is included in the database. The age range of the subjects spans from 17 to 87 years, with a mean of 57.2 years. Data for each subject is divided into 1 to 5 separate recordings. While the majority of subjects are male (209), with a mean age of 55.5 years, there are also 81 females (mean age 61.6 years). It's important to note that age data is missing for one female and 14 male subjects.

Records are provided as 15-lead ECGs, consisting of 12 conventional leads (i, ii, iii, avr, avl, avf, v1, v2, v3, v4, v5, v6) along with the 3 Frank lead ECGs (vx, vy, vz). Table 1 provides a basic description of the ECG leads sampled, with each signal recorded at sampling rates of up to 10 kHz.

Most ECG recordings within the database are accompanied by a corresponding header file (.hea) containing detailed clinical summaries for each subject. These summaries typically include demographic information (age, gender), diagnosis, and, when applicable, data on medical history, medications, interventions, coronary artery pathology, ventriculography, echocardiography, and hemodynamics. It's important to note that clinical summaries are absent for 22 subjects. The diagnostic classifications for the remaining 268 subjects are presented in Table 2. The most focused class throughout the proposed work is the Myocardial Infarction class.

5. Proposed Methodology

This study presents an automated electrocardiogram (ECG) analysis methodology to improve diagnostic accuracy non-invasively. The core of the proposed method lies in a two-pronged approach. First, it emphasizes the critical role of feature selection. By meticulously extracting the most informative and reliable morphological features from the raw ECG signal, the method aims to capture the most relevant aspects of the electrical activity within the heart. These features likely hold the key to accurately differentiating between healthy and abnormal cardiac function. Secondly, the methodology

Table 1: *Description of ECG Leads*

Lead	**Type**	**Description**	**Usage**
i	Standard Limb Lead	Potential difference between the right arm (RA) electrode (+) and the left arm (LA) electrode (-)	Lateral MI
ii	Standard Limb Lead	Potential difference between the right arm (RA) electrode (+) and the left leg (LL) electrode (-)	Inferior MI
iii	Standard Limb Lead	Potential difference between the left arm (LA) electrode (+) and the left leg (LL) electrode (-)	Inferior MI
avr	Augmented Voltage Right	Activity with the right arm (RA) as reference and views the heart from the perspective of the left arm and left leg	Inferior MI
avl	Augmented Voltage Left	Activity with the left arm (LA) as reference and views the heart from the perspective of the right arm and left leg	Lateral MI
avf	Augmented Voltage Foot	Activity with the left leg (LL) as reference and views the heart from the perspective of the right arm and left arm	Inferior MI
v1, v2, v3, v4, v5, v6	Chest Lead	Transverse plane view of the heart's electrical activity, labeled V1 (right sternal border) and V6 (left midaxillary line)	Anterior MI Posterior MI
vx, vy, vz	Frank VCG Lead	Three orthogonal components of the vectorcardiogram representing the x (transverse), y (frontal), and z (sagittal) axes	Location of myocardial damage

optimizes existing deep learning components within the employed neural network architecture. This optimization process seeks to enhance the network's ability to learn complex relationships between the extracted features and the corresponding patient health status. Ultimately, the combined effect of these strategies is to achieve a more robust and reliable classification of patients as either healthy or unhealthy. A detailed illustration of the overall methodology is discussed in the following subsections and presented in Figure 5,

Table 2: *PTB Diagnostic Dataset classes*

Diagnostic Class	**Number of subjects**
Myocardial Infarction	148
Cardiomyopathy/Heart failure	18
Bundle branch block	15
Dysrhythmia	14
Myocardial hypertrophy	7
Valvular heart disease	6
Myocarditis	4
Miscellaneous	4
Healthy controls	52

allowing for a deeper understanding of the proposed approach.

5.1. Input Phase

The main aim of this phase is to prepare the input that allows for the optimal extraction of the best features. This enables the cleanest and best quality multi-channel ECG data.

- **Signal Extraction from Data Files:** This methodology starts with data files in the form of signal(.dat) and header files(.hea and .xyz). The data are annotated by dividing patient information into dictionaries where each dictionary contains basic information (age, sex, date), diagnosis (class for admission), hemodynamics (cardiac information), and therapy (medication). The architecture to be described requires Python waveform generation libraries to run ECG records and annotations. ECG records of healthy and unhealthy patients were indexed, each containing the path to the corresponding signal.

- **Signal Data Transformation:** After data has been properly presented by unique patient ID, the signals are readily extracted. Despite using a clean dataset as referred to in section 4, data anomally checks need to be carried out first before processing. ECGs were checked using a power-line interference filter, in case interference occurred while medical equipment were used to collect patient data thus removing potential electric noise. This filtration utilized an FIR (Finite Impulse Response) filter. The FIR filter acts as a low-pass filter specifically

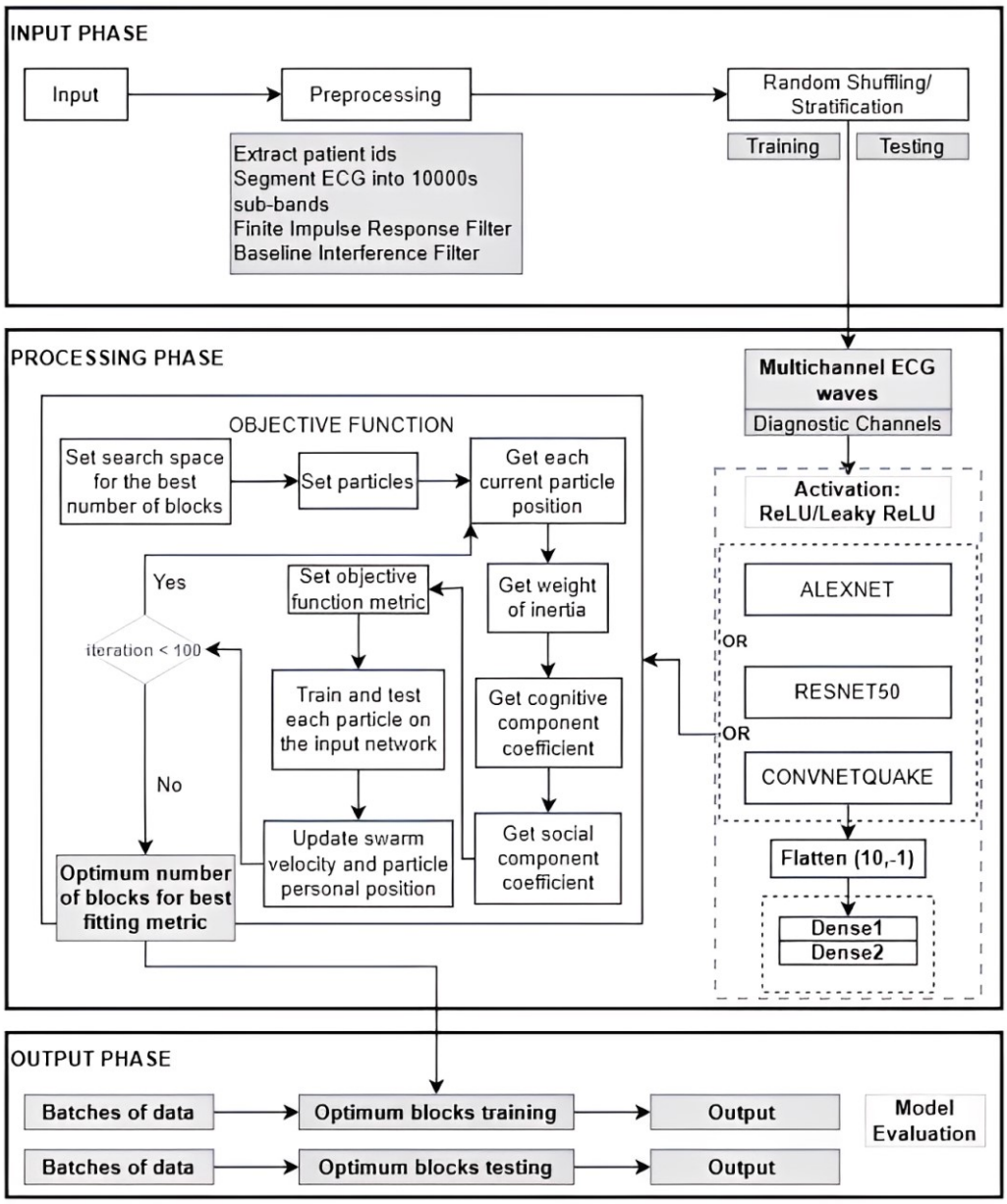


Figure 5: Overview of an optimized ischemia diagnosis model

designed to target and retain frequency components within the typical range of ECG signals. The filter utilizes a cut-off frequency of 50Hz, ensuring the elimination of higher-frequency noise that may interfere

with subsequent analysis. Typically, the most beneficial ECG frequencies for diagnosing ischemia are 50Hz and below. Figure 7 shows the basic view of the FIR filter. The proposed architecture in this work preliminarily requires the efficient clustering of ECGs in order to identify the correct classes of different features. For instance, STEMIs require the localization of ST elevation areas. Efficient clustering of ECGs requires efficient feature and morphology representation. The next step is to highlight the most important features in the ECGs. For the purpose of this work, ST elevation is the main feature that is to be recreated or exaggerated at its corresponding time stamp. In [20], each prominent ECG feature was recreated by the third, fourth and fifth Hermite expansion coefficients.
According to [37] and [38], Hermite polynomial expansion can cluster ECG beats and features efficiently. In this work, the coefficients used were the third until the seventh (since we need the minimum required expansions to reconstruct an ECG in the worst case because of the tradeoff between accuracy given by polynomial order and computation time). Hermite polynomials were applied with the third through seventh orders as given in equations 1 to 5 to directly resemble an ECG, as shown in figure 6.

$$H_3(x) = x^3 - 3x \tag{1}$$

$$H_4(x) = x^4 - 6x^2 + 3 \tag{2}$$

$$H_5(x) = x^5 - 10x^3 + 15x \tag{3}$$

$$H_6(x) = x^6 - 15x^4 + 45x^2 - 15 \tag{4}$$

$$H_7(x) = x^7 - 21x^5 + 105x^3 - 105x \tag{5}$$

- **Data Stratification and Batching:** By now, the data will be practically ready for stratification into training and validation sets. The clean, extracted signals are stratified and shuffled randomly into the standard 80% training and the 20% validation sets. Then, each extracted array of ECG signal data is randomly sampled in batches with a window size of 10,000 seconds. Each batch is made up of random samples of each ECG channel per category. Because ECG amplitude is not required for the diagnosis in this work, Min-max normalization is

used to ensure ECGs remain regular throughout the stochastic batching process. ECG recordings categorized as healthy and unhealthy are grouped together. This grouping process results in batches comprised of multichannel signals. These prepared batches are then directed towards the subsequent processing phase, as illustrated in Figure 7.

- **Remarks on Handling Data:** In the next section, which discusses the processing phase, three neural networks were tested, AlexNet, ResNet and ConvNetQuake. The ECGs extracted, cleaned and sampled were kept both as arrays and as images so as to allow for more adaptibility when training neural networks. This also achieves the correct dimension for the ECGs being analyzed, where the images work with networks adaptable for 2D data and arrays work with 1D networks, while preserving the original channels used for STEMI diagnosis (an anterior, inferior and a Frank sagittal axis, refer to table 1) in section 4.

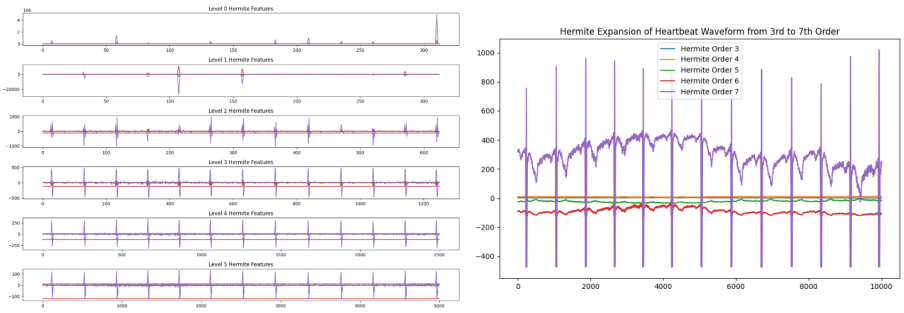


Figure 6: Example of Hermite polynomial expansion on an unhealthy patient

5.2. Processing Phase

The processing phase is divided into two main processes. The first process is passing the extracted multi-channel signals through standard CNNs, which are AlexNet, ResNet50, or ConvNetQuake [19] network with an element-wise ReLU activation, as shown in Figure 5.

After that, the output is reshaped into a flat array, then passed through two linear layers and finally a decision layer. The forward pass through this architecture is to output the best-trained number of pairs of dense blocks according to a set metric. Then, this number of pairs is used to train and test one of the standard architectures used.

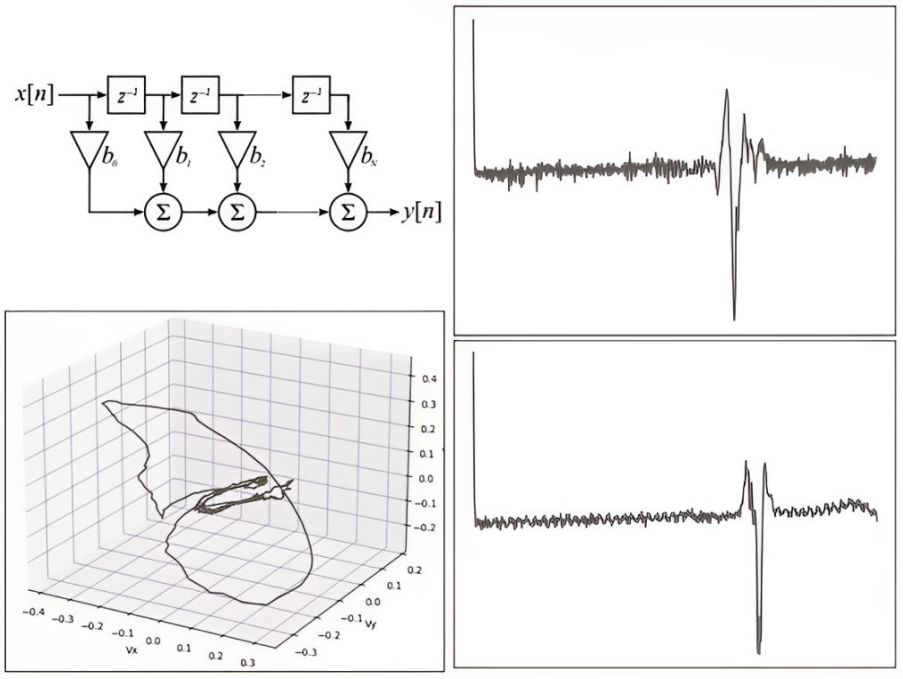


Figure 7: Myocardial Infarction patient against a Healthy patient

The second process utilizes one of the most efficient stochastic population-based techniques: particle swarm optimization (PSO)[39]. presents a compelling optimization technique distinguished by its simplicity, efficiency, and ability to handle problems without requiring gradient information [40]. This versatility makes PSO applicable to a wide range of optimization scenarios [41]. The core principle of PSO involves a swarm of particles representing candidate solutions that are randomly distributed within the search space. Each particle's fitness is evaluated by the objective function, akin to finding food at its location. Notably, each particle possesses knowledge of its current fitness, its personal best solution encountered so far (local optimum), the best solution discovered by the entire swarm (global optimum), and its own velocity. This information guides the particles' movement through the search space, allowing them to explore promising regions iteratively and potentially converge toward the optimal solution.

The successful implementation of PSO hinges on a select set of parameters that significantly influence the algorithm's behavior and effectiveness in optimizing a particular problem. These parameters encompass:

- Particle swarm size (num_p): This defines the number of candidate solutions explored simultaneously.
- Problem dimensionality (d): The number of variables involved in the optimization problem.
- Particle velocity (V): The rate at which particles move through the search space.
- Inertia weight (ω): This factor regulates the influence of a particle's past trajectory, balancing the exploration of new regions with the exploitation of promising areas identified earlier.
- Particle velocity limits ($minV$, $maxV$): These establish constraints on the maximum velocity of particles, preventing them from escaping the search space.
- Cognitive learning rate (c_1): This parameter determines the weight given to a particle's personal best solution, influencing its tendency to revisit promising areas it has encountered.
- Social learning rate (c_2): This parameter determines the weight given to the swarm's globally best solution, guiding particles towards areas identified as potentially optimal by the entire swarm.
- Random factors (r_1,r_2): These stochastic elements introduce a degree of variation in particle movement, promoting diversification and mitigating the risk of premature convergence on sub-optimal solutions.

The goal is to locate the single configuration that best satisfies the criteria established by the chosen problem representation (i.e., the global best position of a particle).
Within the PSO, each particle is characterized by vectors of d-dimensional, which represent its position and velocity, respectively, according to Equations 6 and 7:

$$X_i = (x_{i1}, ..., x_{id}) \tag{6}$$

$$V_i = (v_{i1}, ..., v_{id}) \tag{7}$$

where X_i and V_i are the position and velocity vector of a particle i in the search space, *space*.

The PSO algorithm starts by randomly initializing particles' positions and velocities. Subsequently, each particle's fitness is evaluated according to the objective function. Identify the particle with the best fitness value among all particles in the swarm and assign it the current global best solution (*gbest*) while the local best one is assigned to the personal best (*pbest*). This work utilizes an objective function centered on identifying the optimal number of blocks within the employed Convolutional Neural Networks (CNNs). The aim is to achieve the best performance as measured by non-optimized metrics, specifically ConvNetQuake and ResNet50.

As particles navigate the search space in their quest for the optimal solution, their velocity and position undergo continuous updates according to Equations 8 and 9, respectively. This iterative process guides their exploration and convergence towards the most promising regions.

$$V_i^{k+1} = \omega V_i^k + c_1 r_1 (pbest_i^k - X_i^k) + c_2 r_2 (gbest_i^k - X_i^k) \tag{8}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{9}$$

where K is the current iteration number for a particle i in a set of epochs. c_1 and c_2 are cognitive and social learning rates that affect the intensity of the random forces acting in the direction of *pbest* (previous best) and *gbest* (global previous best). r_1 and r_2 are random values ranging from 0 to 1 and ω denotes the inertia weight.

During the update process, it's customary to impose limitations on both the positions and velocities of particles within the swarm [42][43]. These bounds serve a crucial purpose, as illustrated in Algorithms 1 and 2, corresponding to the particle position and velocity clamping, respectively. By enforcing these constraints, we ensure that particles remain within the defined search space, preventing them from venturing into irrelevant areas and hindering the optimization process.

Algorithm 1 Particle position clamping

```
if x_i < minX then
    x_i ← minX
else if x_i > maxX then
    x_i ← maxX
end if
```

Algorithm 2 Particle velocity clamping

1: **if** $v_i < minV$ **then**
2: $\quad v_i \leftarrow minV$
3: **else if** $v_i > maxV$ **then**
4: $\quad v_i \leftarrow maxV$
5: **end if**

Specifically, the objective function plays a central role in PSO, assigning a score to each particle based on how well its corresponding model fits the data. This score serves as a measure of the particle's "goodness". A balance between exploration and exploitation guides particle movement. The cognitive coefficient (c_1) represents the influence of a particle's personal best solution (*pbest*), while the social coefficient (c_2) reflects the influence of the swarm's globally best solution (*gbest*) discovered so far. These coefficients are further modulated by random factors (r_1 and r_2) to introduce an element of stochasticity. As illustrated in Figure 8, each particle's configuration, represented by the number of blocks in a CNN architecture, is trained on the data. The resulting performance metric, such as ROC AUC (Area Under the Curve) or accuracy, is then used to update both the particle's personal best and potentially the global best solution (*gbest*) if a better performing configuration is found. This iterative process of exploration and exploitation continues until the optimal number of blocks is identified, effectively leading the swarm toward the configuration that yields the best performance value.

In PSO, the inertia weight (ω) plays a crucial role in balancing exploration and exploitation during the search process. To facilitate a comprehensive update of the entire swarm, a common strategy involves linearly decreasing the weight of inertia over time. This approach allows particles to maintain some of their current velocity and direction in the early stages, promoting exploration of a wider search space. As the optimization progresses and the swarm converges towards promising regions, the decreasing inertia weight gradually reduces the influence of past movements, encouraging particles to focus on exploiting these areas for potentially better solutions. According to existing empirical studies, as the number of iterations increases, the typical decrease range is 0.9 - 0.4 [44]. Therefore, the inertia weight can be viewed as the fluidity of the medium in which a particle travels [45]. A higher value (i.e., 0.9) indicates extensive exploration in a low-viscosity medium, so gradually decreasing the inertia to a lower value (i.e., 0.4) indicates more exploitation

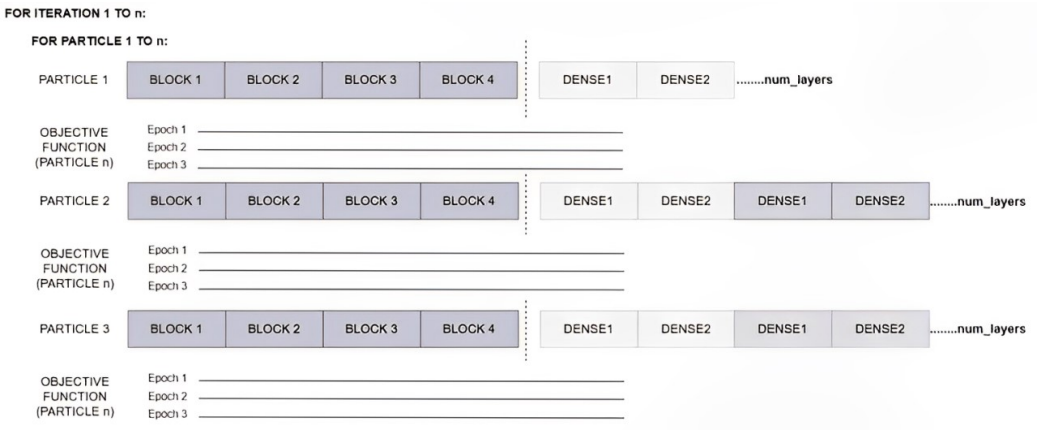


Figure 8: An illustrative example of utilizing particle swarms to optimize the number of dense layers in a CNN.

in a higher-viscosity medium.

In case of more exploitation, PSO naturally lacks global search ability at the end of each iteration. The swarm-based optimization technique can get stuck at a local saddle point or minima without a good global search ability. Consequently, adaptive strategies such as adjusting the weight of inertia (coefficient to the velocity) are suggested to provide results closer to the optima such as the Linear Decreasing Inertia Weight (LDIW) strategy according to Equation 10 :

$$\omega_k = (\omega_{start} - \omega_{end})(\frac{k_{max} - k}{k_{max}}) + \omega_{end} \tag{10}$$

where ω_{start} and ω_{end} are the initial and final values of inertia weight, k is the current iteration number, k_{max} is the maximum iteration number, and $\omega_k \in [0, 1]$ is the inertia weight value in the k-th iteration.

The PSO algorithm terminates after reaching a pre-defined number of iterations (k_{max}). In this instance, the stopping criterion is set to a specific number of iterations performed by a swarm of 100 particles. The successful completion of this stage yields the optimal number of dense block pairs within the CNN architecture, which is expected to achieve the best possible metric value or performance characteristic. Algorithm 3 provides a detailed illustration of the core steps involved in this optimized deep learning model.

During the training phase, the PSO function runs with every training epoch (bench-marked at 5000 epochs), with default exploration and exploita-

Algorithm 3 Optimized DL Model using PSO Algorithm

Require: Search space (*space*), Maximum number of iterations (k_{max}), number of particles (num_p), number of pairs of dense blocks (num_layers), personal best score (*score*), personal best position (*pbest*)

Ensure: *gbest* position (found at the last iteration and where the ROC AUC is calculated to attain the best number of blocks)

Initialize particle position and velocity randomly.
Initialize global best position, best position (Initialize to zero for start)
Initialize global best score, best score (Initialize to a large value, e.g., ∞)

Initialize inertia weight, ω
Set the cognitive coefficient, c_1 and the social coefficient, c_2
for $k \leftarrow 1$ to k_{max} **do**
  for $i \leftarrow 1$ to num_p **do**
    Set objective function $f(x_i)$ to particle position
    Clamp the position to get *pbest* of x_i
    Clamp the velocity to get the velocity v_i
    for j in range of num_layers **do**
      Fit the model to the current particle x_i
      Calculate the ROC AUC
      Calculate the score
      if score $<$ best_score(x_i) **then**
        $gbest(x_i) \leftarrow x_i$
        $best\_score(x_i) \leftarrow score$
      end if
    end for
  end for
  Update the particle velocity and position
  Generate random factors r_1 and r_2
  Update particle velocity using Equation 8
  Update particle position using Equation 9
end for
return the best particle position so far *gbest*

tion of 100 particles. Each particle poses as a neural network with a potential number of dense layers, while each epoch is measured by accuracy, loss and area under curve (AUC). Updates occur until a final number of dense layers is yield. After the output phase, re-training the neural network with the selected number of dense layers is suggested to check the results. As for the validation phase, it is done alongside the training phase. Finally, the testing phase is done first on the test split of the data and second on known cases and compared in order to draw a confusion matrix.

5.3. Output Phase

Building upon the previous phase, where an optimal number of blocks was identified for the deep learning model, this final stage focuses on rigorously evaluating its performance. The chosen number of blocks is tested on batches of ECG signal data. These tests can be conducted on individual ECG channels or the complete one-dimensional ECG signal. Various metrics are employed to comprehensively assess the model's effectiveness, including accuracy, confusion matrix, and receiver operating characteristic (ROC) curve.

6. Experimental Results and Performance Analysis

This section presents a comparative analysis of various benchmark models and architectures trained on both single-channel and multi-channel electrocardiogram (ECG) data. All experiments were conducted using a 10-core CPU and a single GPU. While training outcomes were comparable for single-channel and multi-channel models, subsequent testing revealed a clear superiority of multi-channel models across various evaluation metrics, including accuracy, precision, recall, and F1-score. These metrics are derived from the calculation of true positives (i.e., Correctly predicted positive cases), true negatives (i.e., Correctly predicted negative cases.), false positives (i.e., Incorrectly predicted as positive "Type I error"), and false negatives (i.e., Incorrectly predicted as negative "Type II error").

- Accuracy: Measures the proportion of correct predictions out of all predictions ($TotalPrediction$). It can be computed according to Equation 11.

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalPrediction} \tag{11}$$

- Precision: Measures how many of the positive predictions were actually correct. It can be computed according to Equation 12.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{12}$$

- Sensitivity (Recall): Measures how many of the actual positive cases were correctly identified. It can be computed according to Equation 13.

$$Sensitivity = \frac{TruePositive}{TruePositive + FalseNegative} \tag{13}$$

- F1-Score: Is the weighted average of Precision and Recall. It takes both false positives and false negatives into account. It can be computed according to Equation 14.

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{14}$$

Furthermore, the confusion matrix can provide a class-by-class breakdown of the classifier's effectiveness. This breakdown includes the counts of true positives, false positives, true negatives, and erroneously listed false negatives as represented in Table 3.

Table 3: *Confusion matrix view*

	Positive	Negative
Positive	True Positive	False Negative
Negative	False Positive	True Negative

A set of experiments is conducted to achieve promising results regarding the average accuracy of comparative architectures. Based on further experiments, including channel-base and transform-based architectures, and finally, the optimized deep learning model using particle swarm optimization, the best one is selected. These experiments are illustrated below.

6.1. Experiment on Average Accuracy

Initial experiments utilized multi-channel input data, specifically employing three primary channels. These channels were preprocessed into image format to serve as input to a preliminary non-optimized implementation of

Table 4: *Average accuracy of non-optimized model architectures*

Model	**AlexNet**	**ResNet50**	**ResNet101**	**ConvNetQuake**	**ConvNetQuake**
Leads	ii, v6, vz	ii, v6, vz	v1, v2, v3, v4, v5, v6	ii, v6, vz	v2, v6, vz
Notes	Standard ImageNet	Feature matrices are in batches	Hermite transform of averaged leads	Physician-level deep learning	Test with anterior lead, v2
Avg. Accuracy (%)	63.8	98.67	90	88	85

the proposed architecture. Table 4 provides a summary of the mean accuracy achieved across the various channels evaluated.

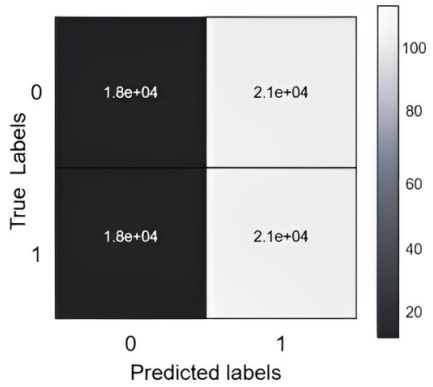
Among the non-optimized three-channel architectures evaluated, ResNet50 demonstrated superior performance, achieving an average accuracy of 98.67%. ConvNetQuake followed as a close second with an average accuracy of 88%, while AlexNet exhibited the lowest performance. To conduct a more in-depth analysis, ResNet50 and ConvNetQuake, given their promising initial results, were selected for further experimentation and comprehensive metric evaluation.

6.2. Further Experiments on the Selected Non-Optimized Architectures

A ResNet50 model was retrained for single-channel ECG classification using a Hermite transform (orders 3, 4, 5) applied to data preprocessed with a low-pass filter (as depicted in Figure 7). This model achieved a 90% accuracy rate during training, as indicated in Table 4. However, subsequent testing on various ECG channels revealed unsatisfactory performance.

The first experiment performed on ConvNetQuake involved Hermite-transformed 3-channel input, where the 3-channel waves were filtered, averaged, and transformed to be input into the network. The second experiment on ConvNetQuake used stochastic random batches of untransformed, normalized waves from the 3-channels to be input into the network. While both experiments were performed over long periods of time, the second experiment

yielded a better confusion matrix, as shown in figure 9. The confusion matrix shows that the experiment best works in cases of unhealthy patients, with a precision of 80% and an f1-score of 31%.



Classification Report:

	precision	recall	f1-score
0	0.15	0.74	0.25
1	0.80	0.19	0.31
accuracy			0.28
macro avg	0.47	0.47	0.28
weighted avg	0.69	0.28	0.30

Figure 9: Channel-based ConvNetQuake confusion matrix

Similarly, on ResNet50, the channel-based experiment yielded a better confusion matrix. The confusion matrices of both experiments are shown in figure 10. From the confusion matrices, it can be seen that the channel-based ResNet50 performed better due to a higher predicted diagnosis in unhealthy patients. However, the main disadvantage is the high value of false negatives (a person is unhealthy but is diagnosed as healthy). For this reason, even if ResNet50 performed better in terms of unhealthy patient accuracy and precision, shown in table 6, an optimization is required.

Certain diagnosis cases were undetected by all models, such as patients with the possible existence of Anterior, Posterior, or both. In Figure 11, two chest leads and a Frank lead through the ResNet-50 showed that the chest leads reported almost equal results, which in the context of deep learning shows a good and consistent model, but medically shows a probable confusion between anterior and posterior STEMI cases. This is predicted to be due to the interchange in S-T segment appearance, where an Anterior case shows an elevation and a Posterior case shows a depression. If a patient was diagnosed with an Anterior case, all models failed to show that there are opposing patterns in chest ECG channels. In figure 12, the testing phase occurred on local patient test cases unlike simply the test split shown in the previous diagram.

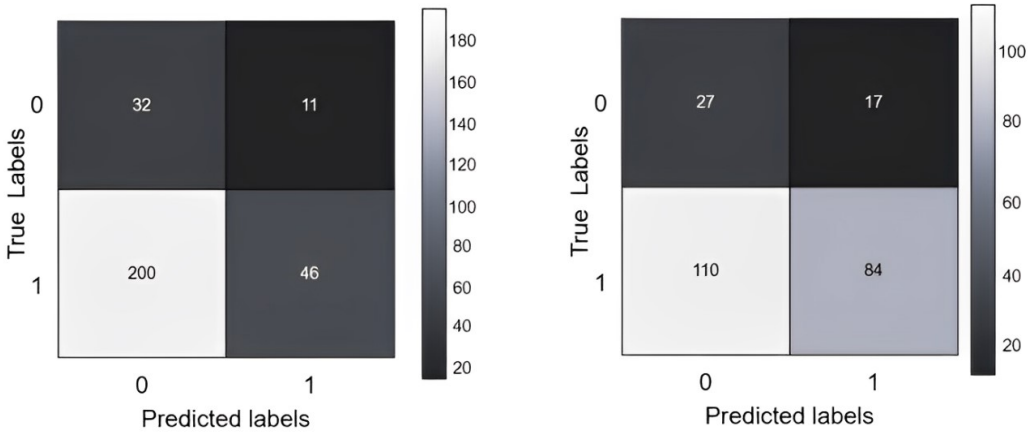


Figure 10: Transform-based ResNet50 and Channel-based ResNet50

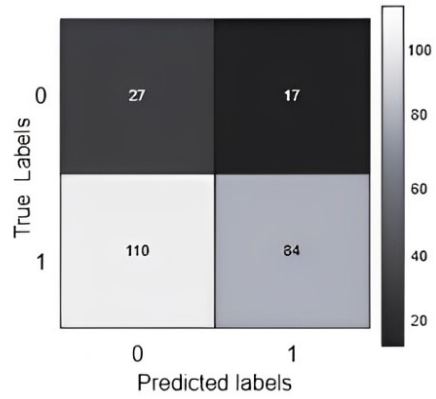
6.3. Experiments on Optimized Architecture using PSO

An optimization process was undertaken to determine the optimal number of blocks for each architecture, to maximize the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve according to the formula in Equation 15. Note that the global optimum aimed for is to find the best number of dense blocks (to output a good diagnosis with a reasonable computation burden) to attain the optimum ROC-AUC.

$$\text{ROC AUC} = \int_0^1 \text{TPR}(\text{FPR}^{-1}(t))\, dt \tag{15}$$

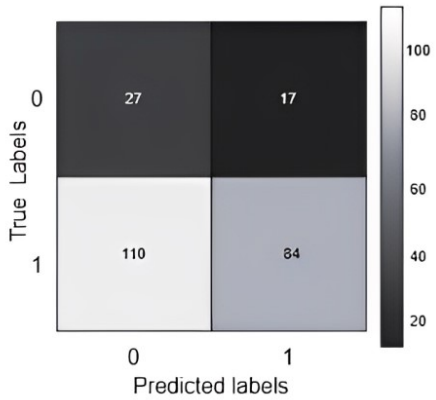
where TPR denotes the True Positive Rate (Sensitivity), FPR^{-1} represents the inverse of the False Positive Rate (1 - Specificity), t ranges from 0 to 1, representing the threshold used for classifying positive and negative instances.

To optimize the best-performing models (ConvNetQuake and ResNet50), the parameter settings are shown in table 5. ConvNetQuake required a lower learning rate to yield good results with a high number of iterations, since no appropriate results were yielded otherwise. Despite the lack of importance given to the calculation of loss in this paper, ConvNetQuake used Binary Cross Entropy since it only performed well when given only 2 classes, unlike ResNet50 which is much more versatile. In this paper, all experiments only feature binary class diagnosis.



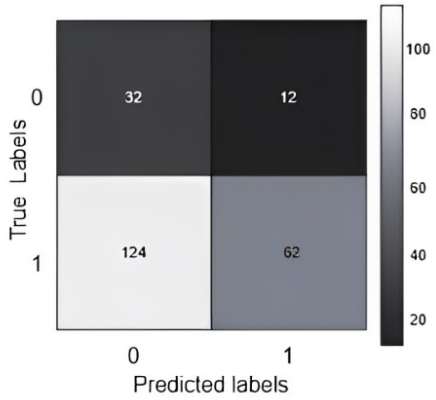
Classification Report:

	precision	recall	f1-score
0	0.20	0.61	0.30
1	0.86	0.49	0.62
accuracy			0.51
macro avg	0.53	0.55	0.46
weighted avg	0.74	0.51	0.57



Classification Report:

	precision	recall	f1-score
0	0.19	0.61	0.29
1	0.85	0.47	0.60
accuracy			0.49
macro avg	0.52	0.54	0.45
weighted avg	0.74	0.49	0.55



Classification Report:

	precision	recall	f1-score
0	0.21	0.73	0.32
1	0.88	0.42	0.57
accuracy			0.48
macro avg	0.55	0.58	0.45
weighted avg	0.77	0.48	0.53

Figure 11: Classification reports of a ResNet-50 experiment with three leads: v2, v6, vz

Table 6 shows the main model evaluation metrics.According to the results shown in table 6 and in table 8, ConvNetQuake is quite a stable model, espe-

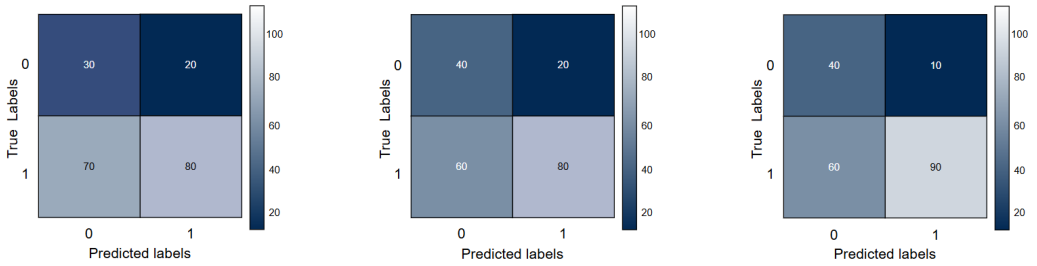


Figure 12: Testing optimized ResNet50 on local test cases with three leads: v2, v6, vz

Table 5: *Parameter settings for the two best performing models*

Parameter	ResNet50	ResNet50-PSO	ConvNetQuake	ConvNetQuake-PSO
Learning Rate	0.001	0.001	0.0001	0.0001
Batch Size	16	16	16	16
Loss	Sparse Categorical Cross Entropy	Sparse Categorical Cross Entropy	Binary Cross Entropy	Binary Cross Entropy
Optimizer	Adam	Adam	Adam	Adam
No of iterations	20	20	5,000	5,000
No of particles	-	3	-	3

cially with unhealthy patients. The result also shows the number of layers, or how dense, should each good-performing 3-channel model be to attain a good operational characteristic, as shown in table 8 and figure 13. In table 8, it is shown that the maximum number of pairs of dense blocks required by ResNet50 to achieve a good ROC, is 8 pairs of blocks, where ConvNetQuake showed stability with a maximum of 1 pair of dense blocks. This proves a major disadvantage in terms of processing time, as the former takes more time and requires more resources than the latter. In figure 13, upon optimization, it is seen that there is an expected increase in the probability of true positives even if the ROC value itself is lower than the non-optimized experiment.

Transform-based ResNet50, despite having agreeable results, shows a better diagnosis in terms of false negative results only (predicted to be healthy but is actually unhealthy). As previously stated, channel-based ResNet50 showed the best confusion matrices regarding both true positive and false negative, as shown in Figure 10.

In conclusion, PSO with an objective function that takes a number of layers yielded a better ROC curve, using half the epochs for 1 iteration, despite the possibility of having a lower ROC value than the non-optimized

Table 6: *Average metrics between ResNet50 and ConvNetQuake*

Model Architecture	**Avg Precision**	**Avg Recall**	**Avg F1 Score**	**Avg ROC**
ConvNetQuake-PSO	0.840	0.190	0.310	0.530
ConvNetQuake	0.800	0.190	0.310	0.500
Transform-based ResNet50	0.800	0.190	0.310	0.530
Channel-based ResNet50	0.863	0.460	0.597	0.540
Channel-based ResNet50-PSO	0.880	0.460	0.600	0.600

Table 7: *Number of blocks required for good characteristic curve in optimized models*

Model	ResNet50			ConvNetQuake		
Leads	ii	v6	vz	ii	v6	vz
Number of blocks required after PSO	8	1	2	1	1	1

version of a model. Given the stated results, ResNet50 with PSO is the most ideal model in terms of not only accuracy but also ROC (even if ConvNetQuake with PSO yielded better split-wise results). The main trade-offs for using ResNet50 with PSO are processing power and practical cost, which can be solved using better GPUs (Graphical Processing Unit) and more feature-reduction techniques. On the other hand, the practical cost of applying channel-based ResNet50 is much higher due to the high proportion of false negative patients(unhealthy patients diagnosed as healthy). Thus ConvNetQuake is evaluated as a better real-life model since the proportion of true positives and low-risk false positives is considerably higher.

6.4. Resource Management and Experiment Conditions

The results obtained in this study are significantly influenced by hardware constraints, particularly the use of a single GPU and a multi-core CPU for training complex deep learning architectures. Since neural networks benefit from high-performance computing environments, our experimental results faced by severe limitations in regards to environmental setup: random access memory (RAM), virtual memory, processing and driver speed. In this section, we outline the major setbacks of our methodology from a computation perspective.

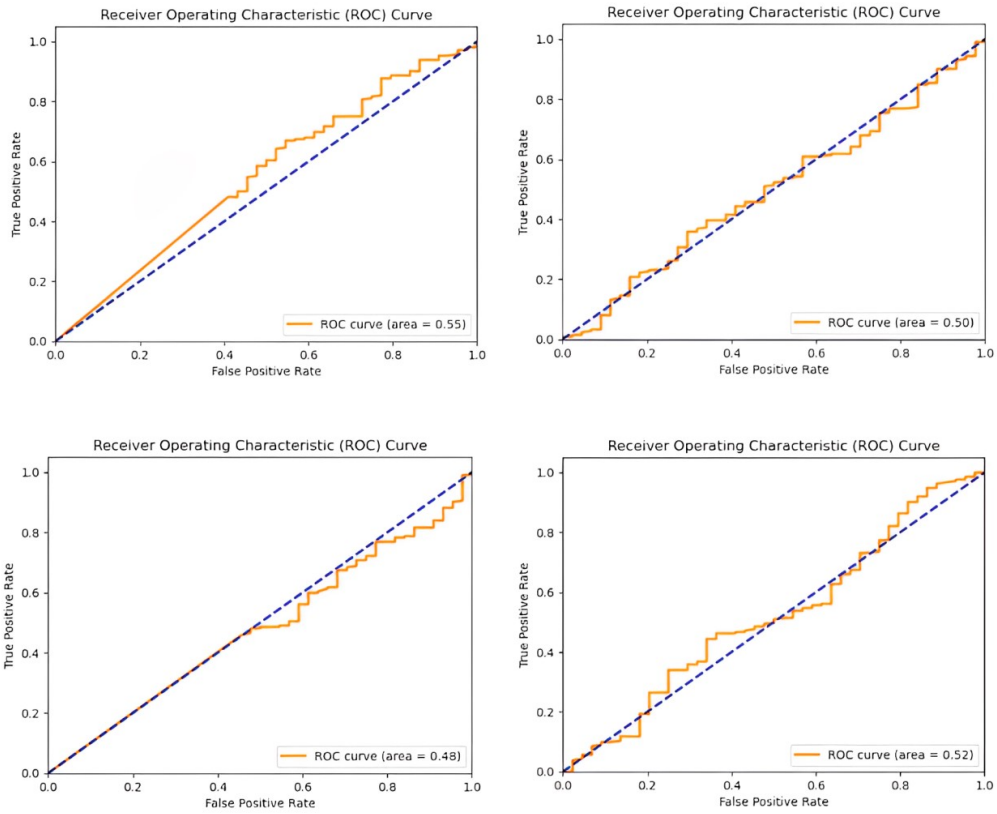


Figure 13: Left to right: Non-optimized channel-based ResNet50 ROC versus channel-based ResNet50 with PSO ROC of final particle in the final iteration; Top to bottom: Channels ii and v6 for ResNet50

- GPU Memory Constraints and Processing Limitations: Our experiments were conducted on a multi-core CPU (set to 4 processing cores in hardware and 10 cores in PyTorch thread processing) and a single GPU, which significantly affected the handling of intermediate ECG processing as well as tensor generation. Large neural networks with residual blocks like ResNet50 and ConvNetQuake require at least 24GB, with a batch size of 64, of video random access memory (VRAM). The VRAM used in our experiments was 6GB with a batch size of 32 for initial experiments. However, due to such limited VRAM availability and the consistent occurrence of OOM errors, the batch size had to be reduced to 10, even while applying PSO. These limitations affected the batch

size required for sufficient, accurate training, testing and even later integration, without which model generalization could've been improved.

- CPU Bottlenecks and Thread Scheduling Overhead: Despite enabling multi-threading optimizations, the saving of intermediate tensors between neural network layers and other similar memory-intensive operations were slowed down (increasing latency). Our experiments on a 3GB RAM device in a non-distributed environment showed a significant slow down in hyperparameter tuning and model evaluation in comparison to an otherwise high-performance distributed GPU setup.

- RAM Limitations and Model Execution Overhead: Complex deep learning networks require persistent tensor allocation, intermediate bias and weight updates. Batch normalization layers were kept in-tact, which increase runtime memory consumption (with an average of approximately 500MB). Virtual memory was overloaded, further degrading the expected performance for even an optimized model.

- GPU Drivers and Software Compatibility Issues: In comparison to recent CUDA(Compute Unified Device Architecture), cuDNN(CUDA Deep Neural Network) or DRAM(Dynamic random access memory) versions [21], the driver used introduced compatibility issues with those performance-enhancing features. CUDA graphs and other functionality used in complex neural networks were unavailable in our experiments. This resulted in longer execution time compared to more recent GPU setups.

The issues explained in this section support the claim that our methodology showed high precision, but suffered from frequent OOM errors, expectedly longer training times and lower expected scalability. There is a tradeoff between accuracy and computational feasibility, which suggests that real-time, industry deployment of these models would obligate hardware upgrades and GPU acceleration.

7. Limitations of the Proposed Methodology

The main limitations encountered in this work can be summarized as limitations with throughput and design. Those limitations directly affect the applicability of the chosen architecture on software components. Throughput

Table 8: *State-of-the-art (SOA) DL models on cardiovascular disease diagnosis*

Methodology	Optimization	Environment	Avg Acc	Avg Sens	Top F1
Our Work (PSO-ConvNet)	Adam, Particle Swarm	CPU: 4 GB RAM, Intel core i5-7200U Processor: 2.5 GHz GPU: Intel HD Graphics 620, Radeon R5 M430	90.0%	19.00%	0.310
Our Work (PSO-ResNet)			98.67%	46.00%	0.597
M. Fradi et al.[14] 1D-CNN	Adadelta, Adam	CPU: 8 GB RAM, Intel core i7-8565U Processor: 4.6 GHz GPU: NVIDIA GeForce MX250 and NVIDIA Driver v451.67 CUDA v11.0	99.61%	98.85%	0.990
M. Kachuee et al.[15] Deep Residual CNN	Adam	Processor: GeForce GTX 1080Ti	95.90%	95.10%	-
S. Śmigiel et al.[16] SincNet with entropy	Adam	CPU: 192 GB RAM, dual-Intel Xeon Silver 4210R GPU: Nvidia Tesla A100	89.20%	89.30%	0.891
M. Hammad et al.[17]	Adam, without focal loss	-	89.72%	81.11%	0.830
D. A. AlDuwaile et al.[18] Biometric-EffectiNet	-	CPU: 240 GB DDR4 RAM, Intel Core i5-8600K Processor: 3.60 GHz 6-core GPU: One GTX 1080 Ti GAMING OC 11GB	94.18%	-	-
S. Lankford et al.[22] OpenNAS-PSO	Tree-based Pipeline Tool Particle Swarm Ant Colony	Google Cloud Platform AutoML GPU: NVIDIA GeForce GTX 1080 Ti	94.30%	-	-
Junzhong Ji et al.[24] PSO-FBN	Particle Swarm	Codeocean distributed capsules	73.35%	78.40%	-
E. H. Houssein et al.[46] MPA-CNN	Marine Predators	-	99.32%	98.61%	0.987

limitations concern the model running and output. In this work, all models are run on a multi-core CPU and a single-core GPU, leading to slow operation, quick memory clutter, and resource exhaustion per run. Parallelization of some models seemed to resolve the issue, but it led to problems with testing and software compatibility. Model design limitations deal with how complex the code or model is expected to be. This limitation affects software compatibility, model testability, and applicability. Since feature engineering is the least focused area of this work, each model design seeks to test the robustness of conventional classifiers without heavy reliance on feature engineering and precise segmentation. In this study, the most ideal model reports the best metrics. However, some trade-offs are expected when considering the ideal software model.

8. Conclusion and Future Directions

By the end of this work, it is concluded that optimized benchmark models should be dense enough to allow for the correct diagnosis of channel-split ECGs. This paper explored many model architectures and implementations, mainly under the assumption that feature engineering and categorization

principles, such as discrete wavelet or Fourier transform, are not applied. The main outlined technique in this paper starts with random batches of window-sized ECGs of selected channels that are extracted, trained, tested, and optimized. Finally, all models should be ready to be applied as separate software or web app components. Due to existing limitations related to throughput and design where all models are run on a multi-core CPU and a single-core GPU, leading to slow operation, quick memory clutter, and resource exhaustion per run. Parallelization of some models seemed to resolve the issue, but it led to problems with testing and software compatibility. To mitigate the constraints in the previous section, it is recommended to use cloud-based GPU clusters for distributed training. This means that resources will not be tied to the local, non-distributed environment. Model quantization and thread parallelization techniques can be employed to reduce memory footprint or debt. On the long term, to accommodate high-dimensional processing, the need to upgrade to newer GPUs with larger VRAMs is inevitable. At the end of this study, clinical uses of this methodology may be possible in the future, but given the analysis in this study it is not feasible. For better clinical application, clinical diagnosis conventions should be used. For instance, an analysis comprising of merely statistically evaluating the ST elevation regions only, then during diagnosis identifying the ST region and classifying it. Of course, even if the methodology is fully correct, clinical procedures for MI likely extend far beyond just checking for an ST elevation.

Data availability

Publicly available datasets were analyzed in this study. This data can be found here:

https://physionet.org/content/ptbdb/1.0.0/

Disclosure statement

The authors declare no conflict of interest.

Funding

This study is supported via funding from Prince sattam bin Abdulaziz University project number (PSAU/2024/R/1446).

References

[1] D. Cook, S. Oh, M. Pusic, Accuracy of physicians' electrocardiogram interpretations: A systematic review and meta-analysis, JAMA internal medicine 180 (11) (2020) 1461–1471, publisher Copyright: © 2020 American Medical Association. All rights reserved. doi:10.1001/jamainternmed.2020.3989.

[2] N. M. Estes, Computerized interpretation of ecgs, Circulation: Arrhythmia and Electrophysiology 6 (1) (2013) 2–4, doi: url10.1161/CIRCEP.111.000097.

[3] L. Costanzo, Physiology, Board review series, Wolters Kluwer Health, 2014.
URL https://books.google.com.eg/books?id=A1g5zQEACAAJ

[4] H. M. R. K. Imran A Khan, C. K. Panda, Atypical presentations of myocardial infarction: A systematic review of case reports, Cureus 15 (2023). doi:10.7759/cureus.35492.

[5] R. Jothiramalingam, A. Jude, J. D, Review of computational techniques for the analysis of abnormal patterns of ecg signal provoked by cardiac disease, Computer Modeling in Engineering & Sciences 128 (2021) 1–25. doi:10.32604/cmes.2021.016485.

[6] R. Kher, Signal processing techniques for removing noise from ecg signals, Journal of Biomedical Engineering and Research 3 (1) (2019).

[7] M. Hassaballah, Y. M. Wazery, I. E. Ibrahim, Ecg heartbeat classification using machine learning and metaheuristic optimization for smart healthcare systems, Bioengineering 10 (429) (2023). doi:10.3390/bioengineering10040429.

[8] B. Kolukisa, H. Hacilar, G. Goy, Evaluation of classification algorithms, linear discriminant analysis and a new hybrid feature selection methodology for the diagnosis of coronary artery disease, IEEE International Conference on Big Data 1 (8622609) (2018). doi:10.1109/BigData.2018.8622609.

[9] A. S. M. Faizal, W. Y. Hon, T. M. Thevarajah, A biomarker discovery of acute myocardial infarction using feature selection and machine learning

[medical & biological engineering & computing (2023) 61:2527–2541], Medical & Biological Engineering & Computing (2023) 61 (8622609) (2023). doi:10.1007/s11517-023-02841-y.

[10] M. Kumar, R. B. Pachori, U. R. Acharya, Automated diagnosis of myocardial infarction ecg signals using sample entropy in flexible analytic wavelet transform framework[evaluation of systems' irregularity and complexity: Sample entropy, its derivatives, and their applications across scales and disciplines, edited by dr. anne humeau-heurtier], Journal of Entropy 19 (488) (2017). doi:10.3390/e19090488.

[11] V. Mondejar-Guerra, J. Novo, J. Rouco, Heartbeat classification fusing temporal and morphological information of ecgs via ensemble of classifiers, Biomedical Signal Processing and Control 10 (1016) (2018). doi:10.1016/j.bspc.2018.08.007.

[12] A. M. Nastaran Jafari Hafshejani, R. Hajian, Identification of myocardial infarction using morphological features of electrocardiogram and vectorcardiogram, IET Signal Processing 1 (2021). doi:10.1049/sil2.12072.

[13] C. L. Tsien, H. S. F. Fraser, W. J. Long, R. L. Kennedy, Using classification tree and logistic regression methods to diagnose myocardial infarction, Studies in health technology and informatics 52 Pt 1 (1998) 493–7.
URL https://api.semanticscholar.org/CorpusID:7104966

[14] M. Fradi, L. Khriji, M. Machhout, Automatic heart disease class detection using convolutional neural network architecture-based various optimizers-networks, IET Smart Cities 3 (12003) (2021). doi:10.1049/smc2.12003.

[15] M. Kachuee, S. Fazeli, M. Sarrafzadeh, Ecg heartbeat classification: A deep transferable representation [version 2; under subject computers and society, last revised 12 jul 2018], arXiv:1805.00794v2 [cs.CY] 1 (1805.00794) (2018). doi:10.1109/ICHI.2018.00092.

[16] S. Smigiel, K. Pałczynski, D. Ledzinski, Ecg signal classification using deep learning techniques based on the ptb-xl dataset, Journal of Entropy 23 (1121) (2021). doi:10.3390/e23091121.

[17] M. Hammad, M. H. Alkinani, B. B. Gupta, Myocardial infarction detection based on deep neural network on imbalanced data[version 1; accepted: 1 december 2020, published online: 6 january 2021], Springer Nature: Multimedia Systems (2022) 28 (10.1007) (2021). doi:10.1007/s00530-020-00728-8.

[18] D. A. AlDuwaile, M. S. Islam, Using convolutional neural network and a single heartbeat for ecg biometric recognition, Journal of Entropy 23 (733) (2021). doi:10.3390/e23060733.

[19] A. Gupta, E. Huerta, Z. Zhao, I. Moussa, Deep learning for cardiologist-level myocardial infarction detection in electrocardiograms, in: T. Jarm, A. Cvetkoska, S. Mahnič-Kalamiza, D. Miklavcic (Eds.), 8th European Medical and Biological Engineering Conference, Springer International Publishing, Cham, 2021, pp. 341–355.

[20] E. H. Houssein, M. Hassaballah, I. E. Ibrahim, D. S. AbdElminaam, Y. M. Wazery, An automatic arrhythmia classification model based on improved marine predators algorithm and convolutions neural networks, Expert Systems with Applications 187 (2022) 115936. doi:https://doi.org/10.1016/j.eswa.2021.115936.

[21] W.-C. Yeh, Z. Liu, S.-Y. Tan, S.-K. Huang, Implementation of parallel simplified swarm optimization in cuda, ArXiv abs/2110.01470 (2021).
URL https://api.semanticscholar.org/CorpusID:238259284

[22] S. Lankford, D. Grimes, Neural architecture search using particle swarm and ant colony optimization (2024). arXiv:2403.03781.
URL https://arxiv.org/abs/2403.03781

[23] T. Gong, Y. Ma, Pso-based lightweight neural architecture search for object detection, Swarm and Evolutionary Computation 90 (2024) 101684. doi:https://doi.org/10.1016/j.swevo.2024.101684.
URL https://www.sciencedirect.com/science/article/pii/S2210650224002220

[24] J. Ji, X. Wang, Convolutional architecture search based on particle swarm algorithm for functional brain network classification, Applied Soft Computing 149 (2023) 111049. doi:https://doi.org/10.1016/j.asoc.2023.111049.

URL https://www.sciencedirect.com/science/article/pii/S1568494623010670

[25] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Communications of the ACM 60 (2012) 84 – 90.
URL https://api.semanticscholar.org/CorpusID:195908774

[26] A. S. B. Mahel, N. Harold, H. Solieman, Arrhythmia classification using alexnet model based on orthogonal leads and different time segments, 2022 Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus) (2022) 1312–1315.
URL https://api.semanticscholar.org/CorpusID:248270067

[27] S. T. Aarthy, J. L. M. Iqbal, Modified parametric-based alexnet structure to classify ecg signals for cardiovascular diseases, Measurement: Sensors (2023).
URL https://api.semanticscholar.org/CorpusID:259071514

[28] M. S. Kolhar, R. N. Kazi, H. Mohapatra, A. M. A. Rajeh, Ai-driven real-time classification of ecg signals for cardiac monitoring using i-alexnet architecture, Diagnostics 14 (2024).
URL https://api.semanticscholar.org/CorpusID:270746671

[29] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015) 770–778.
URL https://api.semanticscholar.org/CorpusID:206594692

[30] C. Han, L. Shi, Ml-resnet: A novel network to detect and locate myocardial infarction using 12 leads ecg, Computer methods and programs in biomedicine 185 (2020) 105138.
URL https://api.semanticscholar.org/CorpusID:204967424

[31] J. A. Wahid, M. Xu, M. Ayoub, S. Husssain, L. Li, L. Shi, A hybrid resnet-vit approach to bridge the global and local features for myocardial infarction detection, Scientific Reports 14 (2024).
URL https://api.semanticscholar.org/CorpusID:267809213

[32] T. Perol, M. Gharbi, M. Denolle, Convolutional neural network for earthquake detection and location, Science Advances 4 (2)

(2018) e1700578. arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.1700578, doi:10.1126/sciadv.1700578.
URL https://www.science.org/doi/abs/10.1126/sciadv.1700578

[33] P. Xiong, S. Lee, G. Chan, Deep learning for detecting and locating myocardial infarction by electrocardiogram: A literature review, Frontiers in Cardiovascular Medicine 9 (2022).
URL https://api.semanticscholar.org/CorpusID:247783067

[34] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, H. E. Stanley, Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals, Circulation 101 (23) (2000) e215–e220.
URL http://circ.ahajournals.org/content/101/23/e215.full

[35] R. Bousseljot, D. Kreiseler, A. Schnabel, Nutzung der ekg-signaldatenbank cardiodat der ptb über das internet, Biomedizinische Technik / Biomedical Engineering (2009) 317–318doi:10.1515/bmte.1995.40.s1.317.

[36] Bousseljot, Ptb diagnostic ecg database, PhysioNet, retrieved October 23, 2024 (September 2004).
URL https://physionet.org/content/ptbdb/1.0.0/

[37] M. Lagerholm, C. Peterson, Clustering ecg complexes using hermite functions and self-organizing maps, IEEE transactions on bio-medical engineering 47 (2000) 838–848. doi:https://doi.org/10.1109/10.846677.

[38] A. Hashim, R. Bakhteri, Arrhythmia detection based on hermite polynomial expansion and multilayer perceptron on system-on-chip implementation, ARPN Journal of Engineering and Applied Sciences 10 (2015). doi:https://www.researchgate.net/publication/285482588.

[39] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: MHS'95. Proceedings of the sixth international symposium on micro machine and human science, Ieee, 1995, pp. 39–43.

[40] M. Arasomwan, A. Adewumi, An adaptive velocity particle swarm optimization for high-dimensional function optimization, in: 2013 IEEE

Congress on Evolutionary Computation, 2013, pp. 2352–2359. doi:10.1109/CEC.2013.6557850.

[41] W. H. El-Ashmawi, A. F. Ali, M. A. Tawhid, An improved particle swarm optimization with a new swap operator for team formation problem, Journal of Industrial Engineering International 15 (1) (2019) 53–71. doi:10.1007/s40092-018-0282-6.
URL https://hdl.handle.net/10419/267616

[42] M. A. Arasomwan, A. O. Adewumi, On the performance of linear decreasing inertia weight particle swarm optimization for global optimization, The Scientific World Journal 2013 (1) (2013) 860289. doi:https://doi.org/10.1155/2013/860289.

[43] H. Liu, Z. Wen, W. Cai, Fastpso: Towards efficient swarm intelligence algorithm on gpus, in: Proceedings of the 50th International Conference on Parallel Processing, ICPP '21, Association for Computing Machinery, New York, NY, USA, 2021. doi:10.1145/3472456.3472474.

[44] Y. Shi, R. Eberhart, Empirical study of particle swarm optimization, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Vol. 3, 1999, pp. 1945–1950 Vol. 3. doi:10.1109/CEC.1999.785511.

[45] R. Poli, J. Kennedy, T. Blackwell, Particle swarm optimization: An overview, Swarm Intelligence 1 (10 2007). doi:10.1007/s11721-007-0002-0.

[46] E. H. Houssein, D. S. Abdelminaam, I. E. Ibrahim, M. Hassaballah, Y. M. Wazery, A hybrid heartbeats classification approach based on marine predators algorithm and convolution neural networks, IEEE Access 9 (2021) 86194–86206.
URL https://api.semanticscholar.org/CorpusID:235476138