Optimizing ECG Heartbeat Classification with Improved Genetic Algorithm and Stacking Ensembles

Thien B. Nguyen-Tat[1,2*], Duc Thang Nong[1, 2], Phuc Hao Nguyen[1, 2], Trong Tam Nguyen[1, 2]

[1]University of Information Technology, Ho Chi Minh City, Vietnam.
[2]Vietnam National University, Ho Chi Minh City, Vietnam.

*Corresponding author(s). E-mail(s): thienntb@uit.edu.vn;
Contributing authors: 21522593@gm.uit.edu.vn;
21522047@gm.uit.edu.vn; 21521405@gm.uit.edu.vn;

Abstract

The analysis of electrocardiogram (ECG) data presents substantial challenges as a result of the complexity of the data and the presence of noise, despite its widespread use in the diagnosis of heart conditions. In this investigation, we suggest a novel methodology that integrates a Stacking ensemble learning framework with an enhanced Genetic Algorithm (GA) for hyperparameter optimization to improve the automatic classification of ECG signals. Although traditional GAs are effective, they frequently fail to achieve global optimum solutions due to limited population diversity, particularly when they heavily rely on mutation. In order to overcome this constraint, we implement improvements to the GA's selection, crossover, mutation, and termination processes, which guarantee optimal outcomes and efficient resource utilization. Experimental evaluations indicate that the proposed Stacking ensemble method, which employs Logistic Regression as the meta-model, achieves an F1 score of 93.16% and an accuracy of 98.83%. These results emphasize the potential of combining enhanced GAs with Stacking ensemble learning to achieve robust and accurate ECG signal classification, thereby providing a promising solution for automated cardiac diagnostics.

Keywords: ECG signal, Machine Learning, Ensemble Stacking, Genetic Algorithm

1 Introduction

Cardiovascular diseases (CVDs), encompass a group of disorders affecting the heart and blood vessels. These include coronary artery disease, arrhythmias, heart failure, and conditions like hypertension. They often develop due to a combination of risk factors such as unhealthy diets, physical inactivity, smoking, and genetic predisposition. CVDs claiming approximately 10,000 lives daily in the European Region. Many of these deaths are caused by acute conditions such as heart attacks and strokes. Globally, ischemic heart disease stands as the leading cause of death, responsible for 13% of all fatalities. According to the World Health Organization (WHO) [1], this condition has experienced the most significant rise in mortality rates since 2000, with deaths increasing by 2.7 million to reach a total of 9.1 million in 2021. Another critical condition is arrhythmia., which refer to irregular or abnormal heart rhythms, where the heart may beat too fast, too slow, or irregularly. Typically, the heart's rhythm is controlled by electrical impulses originating from the sinoatrial node, but when there is a disturbance in this electrical conduction process, the heart rhythm becomes altered.

Critical tool in Diagnosing arrhythmia the electrocardiogram (ECG), which provides a non-invasive method to assess heart function by recording the electrical activity generated during each heartbeat. The ECG measures electrical impulses as they propagate through the heart, resulting in characteristic waveforms, including the P wave, QRS complex, and T wave. These patterns reflect the depolarization and repolarization of the heart's chambers, offering insights into cardiac rhythm, conduction abnormalities, ischemia, and structural heart conditions [2]. By identifying irregularities such as arrhythmias or ischemic changes early, ECGs play a vital role in diagnosis, monitoring, and guiding timely interventions to prevent severe outcomes.

Although the ECG is useful for detecting various cardiac abnormalities, accurate diagnosis of the ECG requires doctors to have considerable professional knowledge. Note that the proper diagnosis of cardiovascular diseases is of paramount importance since these are the cause of death for about one-third of all deaths around the globe. For instance, mil- lions of people experience irregular heartbeats, which can be lethal in some cases. Therefore, accurate and low-cost diagnosis of arrhythmic heartbeats is highly desirable.

To tackle the challenges in ECG signal analysis, the integration of machine learning (ML) techniques has emerged as a promising solution. Modern hospitals are equipped with ECG machines capable of generating large volumes of diagnostic data, paving the way for machine learning applications. As healthcare evolves, the demand for accurate analysis and interpretation of ECG signals continues to grow. Research on ECG signal classification spans from traditional machine learning to advanced deep learning, achieving noteworthy outcomes. For instance, Manish Sharma et al. in 2019 [3], introduced a new approach by using wavelet-based ECG features and the result shows that the support vector machine (SVM) has the best result with F1-score of 0.994, which is more accurate than the existing algorithms. In 2021, Maria Papadogiorgaki et al. [4] performed an experiment with many extracted features approaches from ECG, and the KNN shows the effective result. In the same year, Saira Aziz et al. [5] proposed a fusion algorithm based on FrFT and TERMA to detect R, P, and T peaks, and the result of peak detection has improved the classification task. Pham et al. [6], propose a new approach in ECG classification with 2-5 seconds of ECG signal instead of n individual heartbeats, the result demonstrating good clinical applicability. Despite the positive results of traditional machine learning, there are still a lot of disadvantages, such as requiring expertise and experience and being easily affected by noise and interference. All these disadvantages have created a start for a deep learning approach in ECG classification.

Deep learning (DL) has proven its ability in ECG classification with many positive results, particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs). In 2017, Rajpurkar et al. [7] demonstrated the potential of CNNs in achieving near-human accuracy for arrhythmia detection on a large-scale dataset. Acharya et al. [8], CNN model has reached an accuracy of 94.03% and 93.47% with and without noise removal respectively. M. Kachuee et al. [9] with accuracy of 93.4% and 95.9% on arrhythmia classification and MI classification. RNN-based models, including long short-term memory (LSTM) networks, have been employed to capture the temporal dependencies in ECG signals, as shown by Yildirim et al. [10]. Although the results are almost absolute, researchers still find many approaches in ECG classification tasks. In 2024, Weiran Ji et al. [11] utilized a method based on GRU and Convolutional Neural Network and achieved an overall classification accuracy of 99.47% on the European ST-T Database, with a precision of 98.76% and a recall of 97.92% on the MIT-BIH Arrhythmia Database, especially the classification accuracy for classes N and Q reached 100%. Fan Jiang et al. [12] proposed DCETEN: a lightweight ECG automatic classification network based on Transformer, which achieved 9.84% accuracy and a 99.67% F1 score on the MIT-BIH Arrhythmia Database. End of 2024, Faramarz Zabihi et al. [13] with a hybrid machine learning and deep learning approach achieve a 99.26% accuracy rate and outperform existing methods.

To enhance accuracy in ECG classification, ensemble Stacking has emerged as an effective approach by leveraging the strengths of multiple machine learning models. By combining diverse classifiers, ensemble learning mitigates the limitations of individual models, resulting in a more robust and reliable prediction system. In 2015, Elsayyad et al. [14] utilized an ensemble with C5.0 decision tree to classify normal and abnormal rhythms has reached high accuracy result. Rabinezhadsadatmahaleh et al. [15] in 2020, proposed a stacked ensemble of deep and conventional machine learning classifiers for human biometric identification and reached an accuracy of 99.02 and F-score of 99.01. In 2023, P Kunwar et al. [16] presented the stacked ensemble model to classify the stroke patients using ECGs only with accuracy of 99.7%. In 2024, Sadia Din et al. [17] with the proposed model outdoes the existing models, achieving an accuracy of 95.56% and an F-score of 99.34% . Ensemble learning has shown particular promise in leveraging the complementary strengths of individual classifiers. However, the application of ensemble learning to ECG signals remains underexplored. Optimization techniques such as Genetic algorithms have been increasingly applied to ECG classification to enhance feature selection and hyperparameters tuning. GA is particularly effective in searching large solution spaces and handling noisy or redundant features. For example, Khezripour et al. [18] proposed methods with Genetic algorithm feature extraction for classifying ECG signals in different classes of rhythm to diagnose heart rhythm diseases and achieve good results, ANN classifier with 88.92% for testset and ANFIS with 88.83%. In 2024, Na Zhao et al. [19], developed a model that combines a GA and the stacked ensemble method, capable of differentiating between normal and abnormal ECGs and classifying four types of rhythm for children living in high-altitude areas.

As mentioned earlier, despite extensive research on applying machine learning (ML) and deep learning (DL) to classify ECG signals, several significant research gaps remain: Traditional ML methods, such as SVM, KNN, and feature extraction algorithms, often require deep domain expertise and are highly susceptible to signal noise. Although many studies leveraging CNN and RNN have achieved high accuracy, the application and comparative effectiveness of various DL models remain under-explored. Furthermore, while ensemble learning methods, particularly Stacking, have demonstrated exceptional effectiveness in other fields, their application to ECG classification has not been thoroughly investigated. Along with that, a potential limitation of Stacking is the risk of overfitting, which should also be given strong attention. In addition, although considered a powerful tool for hyperparameters optimization, the Genetic algorithm (GA) still faces many challenges related to processing speed and resource consumption during training. The primary reason lies in the lack of diversity in genetic generations. Despite a certain mutation rate, most subsequent generations remain overly dependent on the initial generation, leading to excessive computational time and cost to achieve expected hyperparameters performance. In fact, there is a scarcity of in-depth studies addressing these limitations, making hyperparameters optimization for machine learning models using GA a significant challenge.

To address these research gaps, this work focuses on applying and improving the Genetic algorithm in combination with stacking ensemble learning for ECG prediction models. The primary goal is to introduce a novel approach that enhances classification accuracy while optimizing computational time and resources. The contributions of this study are as follows:

- Experimenting with and evaluating the performance of various model architectures on the MIT-BIH Arrhythmia dataset.
- Improving key steps in the Genetic algorithm, including generation construction, mutation, crossover, and implementing stopping criteria suitable for the new improvements. Results demonstrate that these enhancements help identify optimal hyperparameters that achieve expected performance while significantly reducing computational time and resources compared to traditional Genetic algorithms.
- Applying the Stacking ensemble method combined with Cross-validation to fully leverage the performance of base models optimized by the Genetic algorithm and minimize the risk of overfitting, achieving the highest performance.
- Comparing with existing ECG heartbeat classification technologies on the MIT-BIH Arrhythmia dataset, our approach achieves competitive performance even without employing techniques to address data imbalance.

Category	Annotations
N (0.0)	• Normal • Left/right bundle branch block • Atrial escape • Nodal escape
S (1.0)	• Atrial premature • Aberrant atrial premature • Nodal premature • Supra-ventricular premature
V (2.0)	• Premature ventricular contraction • Ventricular escape
F (3.0)	Fusion of ventricular and normal
Q (4.0)	• Paced • Fusion of paced and normal • Unclassifiable

Table 1 Classes annotated in the dataset and their meanings

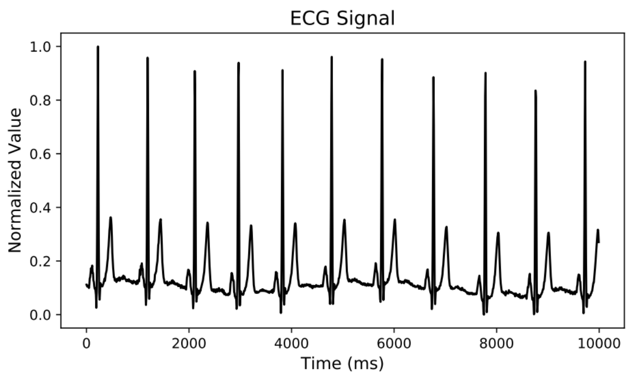


Fig. 1 ECG signals before extraction

2 Material and methods

2.1 Dataset overview

We utilize the ECG heartbeat categorization dataset of ECG Heartbeat Classification: A Deep Transferable Representation paper Kachuee, M., Fazeli, S., Sarrafzadeh, M. (2018, June) [20]. The dataset is a famous collection of heartbeat signals used in heartbeat classification, the MIT-BIH Arrhythmia Dataset with source is The MIT-BIH Arrhythmia Database. The dataset includes 109,446 samples categorized into 5 classes, which is large enough for training a deep neural network. The MIT-BIH Arrhythmia Database, which contains 48 half-hour excerpts of two channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979. Twenty-three recordings were chosen at random from a set of 4000 24-hour ambulatory ECG recordings collected from a mixed population of

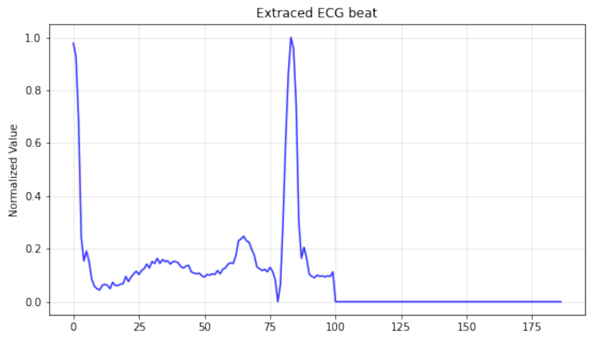


Fig. 2 ECG signals after extraction

inpatients (about 60%) and outpatients (about 40%) at Boston's Beth Israel Hospital; the remaining 25 recordings were selected from the same set to include less common but clinically significant arrhythmias that would not be well represented in a small random sample. We used annotations in this dataset to create five different beat categories by the Association for the Advancement of Medical Instrumentation (AAMI) EC57 standard. See Table 1 for a summary of mappings between beat annotations in each category.

2.2 Data preprocessing

The signals correspond to the shapes of the heartbeat on the electrocardiogram (ECG) for the normal case and the cases affected by different arrhythmias and myocardial infarctions. These signals are preprocessed and segmented, with each segment corresponding to a heartbeat. In order to transform the dataset from the MIT-BIH Arrhythmia Database into more trainable and suitable data, the transformations proposed in the paper ECG Heartbeat Classification: Deep Transferable Representation [20] were applied. The proposed beat extraction method is both straightforward and effective in obtaining R-R intervals from signals with varying morphologies. The process of extracting heartbeats involves the following steps:

1. Divide the continuous ECG signal into 10-second windows and select one window for analysis.
2. Normalize the amplitude values of the signal to a range between 0 and 1.
3. Identify all local maxima using the zero crossings of the first derivative.
4. Determine the potential R-peak candidates by applying a threshold of 0.9 to the normalized local maxima values.
5. Calculate the median of the R-R intervals to establish the nominal heartbeat period (T) for the selected window.
6. For each identified R-peak, extract a segment of the signal with a length equal to 1.2T.
7. Pad each extracted segment with zeros to ensure a consistent, predefined length.

As can see be seen in Figure 1 and Figure 2 the ECG signals before and after extraction. The extracted signal is more visible and electrical signals changes can be easily observed.

In this study, we employed the Genetic algorithm combined with the Stacking ensemble learning method to optimize the performance of classification models. The initial dataset consisted of 109,446 samples; however, due to the large size of the data and to ensure processing efficiency, we utilized only 80% of the total samples as the primary dataset. This portion of the data was divided into three subsets in a 7:1:2 ratio, corresponding to the training (train), validation (validation), and testing (test) sets. The training set was used to train the base models and optimize hyperparameters using the Genetic algorithm. The validation set was employed to evaluate the performance of the models during base model selection and optimization using the Genetic algorithm. Finally, the testing set was used for overall evaluation, comparing the hyperparameters-optimized models with the meta-model in the Stacking ensemble learning method.

2.3 Dataset Analysis

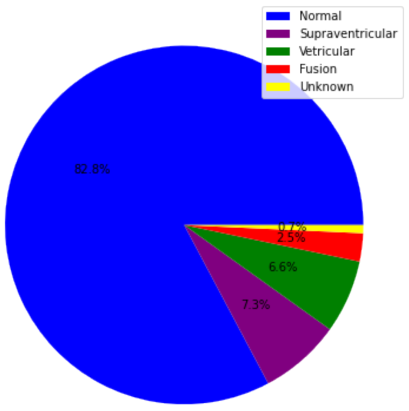


Fig. 3 Ratio of the dataset's classes

As we are doing a classification task in the medical domain, it is important to consider the ratio of the classes in the datasets. We can see in Figure 3 that the class N, which stands for normal cases has a very high ratio in the dataset(82,8%). However, other classes like class F, only play a small part in the whole dataset with 0.7% and the class with the second highest rate only plays 7,3%. This means that the dataset we are using is not balanced and might affect the results when training the model to be biased into one class. Due to the imbalanced dataset issue, we use the F1-score as a main evaluation for the model.

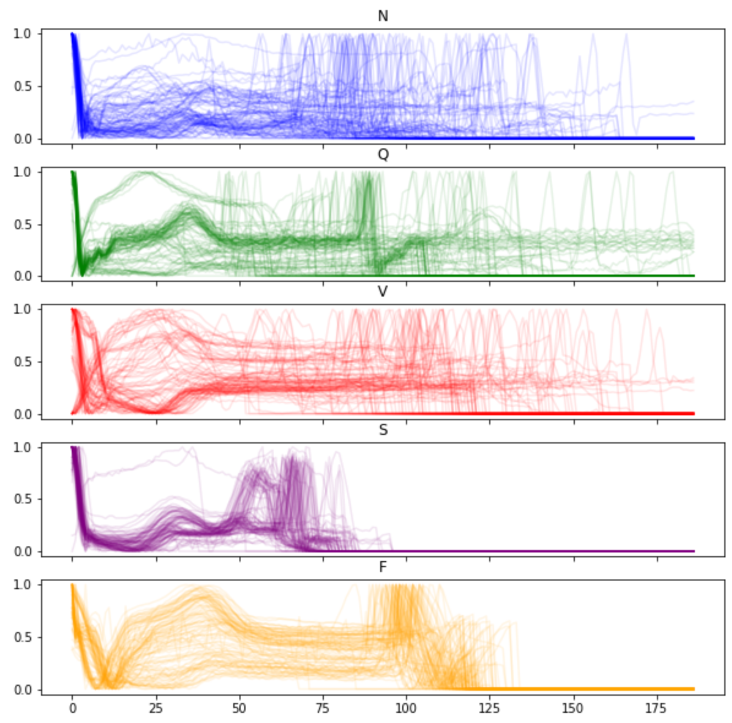


Fig. 4 100 samples for each class in dataset

We visualize 100 samples of each class after extraction. As presented in Figure 4 we could see the difference between each class at the same window of time.

2.4 Baseline Models Selection

Choosing an appropriate base classifier is one of the important factors in building an effective stacked ensemble model. We will train and evaluate a set of candidate models on the dataset to select the base models. The models we choose are SVM [21], KNN [22], LR (Logistic Regression) [23], Naive Bayes (NB) [24], Decision Tree (DT) [25], Random Forest (RF) [26], Linear Discriminant Analysis (LDA) [27], Categorical Boosting (CatBoost) [28], LSTM, CNN. These models are chosen because they are commonly used in similar classification problems, with good ability to solve problems such as nonlinear data (SVM, RF), imbalance (CatBoost, GNB), and time series (LSTM, 1D-CNN). Limiting the number of models to 10 helps us optimize resources and experimental time while still meeting the diversity of different algorithms from machine learning to deep learning when evaluating performance. These models are

Model	**Accuracy**	**F1-score**	**Precision**	**Recall**
SVM	0.9612	0.7958	0.7266	0.9110
KNN	0.9692	0.8592	0.9106	0.8205
RF	0.9730	0.8695	0.9625	0.8052
DT	0.9519	0.7919	0.7920	0.7918
LDA	0.8906	0.6346	0.6364	0.6677
LR	0.9079	0.6085	0.7766	0.5369
NB	0.1853	0.1638	0.3240	0.4626
CatBoost	0.9807	**0.9035**	0.8625	0.9570
LSTM	0.9745	**0.8711**	0.8360	0.9150
CNN	0.9775	**0.8896**	0.8667	0.9157

Table 2 Summary of candidate models evaluation results on the val set

then trained on the training set and evaluated on the validation set. Because the dataset we used in this study is imbalanced, we use the F1-score as the evaluation measure, the 3 models with the largest F1-score will be selected as the base model. The detailed experimental results are shown in Table 2. After the experiment, the 3 models with the highest F1-score are CatBoost, LSTM, and CNN, respectively.

CatBoost is a boosting-based ensemble learning algorithm, optimized for tabular and imbalanced data. By adopting a gradient-based evaluation strategy and symmetric leaf splitting, CatBoost is more robust and generalizes better than traditional boosting algorithms, even with higher efficiency and better accuracy when processing categorical features. In addition, CatBoost can automatically handle missing values and handle large and high-dimensional data sets, while reducing the risk of overfitting. LSTM is a variant of RNN, capable of learning long-term dependencies in sequential data thanks to its special architecture that helps mitigate the vanishing gradient problem. A basic unit of LSTM is a cell with gates called input gate, output gate, and forget gate, respectively. These gates help LSTM networks selectively retain or discard information as it passes through the network, allowing them to memorize long-term dependencies. Thanks to this special architecture, LSTM has partly overcome the vanishing gradient problem in RNN. CNN is a very successful model in the field of image processing and computer vision thanks to its ability to represent and capture local patterns in data. This also makes CNN effective in capturing short-term dependencies in sequences. Sharing weights in convolutional layers reduces hyperparameters compared to fully connected networks, making them more effective for long sequences. In addition, CNN does not depend on the previous state, allowing the model to process multiple parts of the sequence at the same time, speeding up the sequence processing.

From the architectural features of the models that are particularly suitable for the data characteristics in this study and the experimental results, the three models, CatBoost, LSTM, and CNN, are the most suitable models to become base models. However, before these base models are combined with the meta model stacked model, they will need hyperparameter tuning. This helps the base models determine the architecture to fit data and enhance performance on a classification task. Optimizing the performance of the base models will also lead to improved performance of the stacked model.

2.5 Genetic algorithm

Genetic algorithm, GA for short, is a stochastic global search optimization algorithm. This algorithm is inspired by the adaptive evolution of biological populations based on Charles Darwin's Theory of Evolution through natural selection. Specifically, this algorithm combines the understanding of genetics with evolutionary theory. This algorithm works by simulating the natural evolutionary process, which includes steps such as selection, crossover, and mutation to create new generations of potential solutions. Each new generation is expected to be closer to the optimal solution for the problem being solved. Since GA does not require prior knowledge of the search area such as gradients or mathematical properties of the objective function, it is suitable for complex, non-differentiable problems. In addition, the selection, crossover, and mutation operations performed simultaneously on the entire population not only improve the search speed but also increase the ability to explore the entire space to avoid falling into local extremes. With these advantages, GA has been widely used in practical situations.

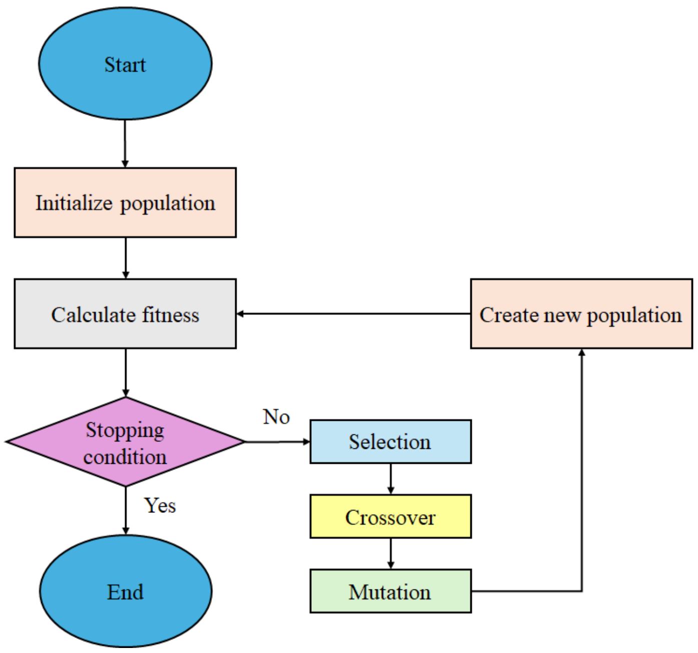


Fig. 5 Genetic algorithm flowchart

Basically, when applying the traditional genetic algorithm, the implementation process in this study is carried out as shown in Figure 5. First, we construct an initial population with 10 individuals randomly initialized. Next, due to the imbalanced nature of the data, we calculate the fitness of each individual in the population using the F1-score. We then select the two individuals with the best performance to breed the next generation. Since offspring may share identical genomes if the parents are similar in one or several hyperparameters, we filter out such duplicate individuals to avoid wasting training resources on individuals with identical genomes. Thus, the number of new offspring created will be $2^n - 2$, where n is the number of hyperparameters that differ between the parents. Each offspring is created with a certain mutation rate applied to one or possibly all hyperparameters, to enhance population diversity and improve the ability to search for the optimal genome. The initial mutation rate is set at 0.1. Finally, through these steps, we obtain a new generation inheriting the genomes of the two most elite individuals from the previous generation. This loop continues until a stopping criterion is met, with a maximum limit of 50 generations in this study.

However, during the experiments, we encountered many difficulties and challenges. One of the greatest challenges was maintaining population diversity. Let us consider the initial population of 10 individuals, where the best-performing individuals are selected in each generation to breed and form a new population. This process repeats until the stopping criteria are met. However, subsequent populations in the algorithm often depend heavily on the initial population, leading to a lack of diversity, even when the mutation step is applied. Relying solely on the initial mutation rate does not provide the necessary diversity within the population, resulting in significant resource and time waste during the search for the optimal hyperparameters for machine learning models. To address this issue, we improved the genetic algorithm by optimizing the crossover, mutation, and generation construction processes while implementing appropriate stopping conditions to enhance population diversity and the effectiveness of the algorithm.

In the crossover step, the removal of duplicate individuals, leaving only a single unique individual, may result in parent pairs failing to produce new offspring or generating too few offspring, falling short of expectations. This leads to a waste of computational resources when populations consist solely of parent pairs or have too few individuals, causing the stopping condition to be met earlier than anticipated and preventing optimal results from being achieved. To address this issue, we randomly introduced entirely new individuals into populations that failed to meet the expected size. This approach is akin to migration in natural evolution or adoption when parent pairs cannot produce offspring. These new individuals are still included in the lineage of subsequent generations. This method not only ensures population size but also significantly enhances genetic diversity alongside mutation, while maintaining the logical framework of a genetic system. Detailed improvements are shown in Figure 6.

Although the number of offspring generated by parent pairs may be very limited, especially in the middle and later generations, it cannot be denied that these offspring inherit the genome, i.e., the hyperparameters of the most superior pair of individuals up to that time. However, if the parent pairs remain unchanged across generations, the offspring produced will also remain identical. Therefore, we progressively increase the mutation rate each time the parent pairs are repeated, applying it to the offspring to alter parts of their genome. This approach not only leverages the best genome at the time but also enhances the likelihood of finding optimal hyperparameters for the model, while improving the genetic diversity of the population. Thus, the mutation rate is set as 0.1 x k, where k represents the number of times the parent pair is repeated across generations. The maximum mutation rate is 1.0. Detailed improvements are shown in Figure 7.

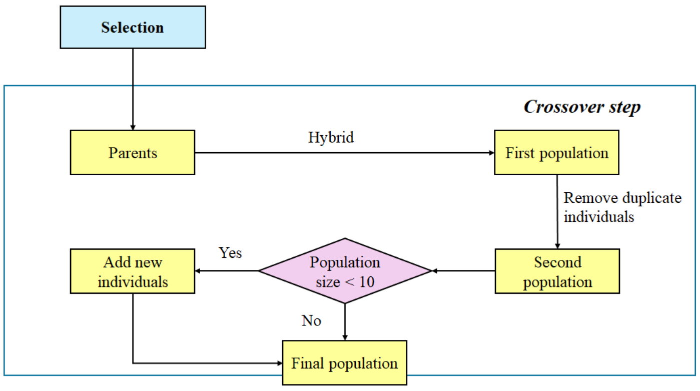


Fig. 6 Improved details of the Crossover step

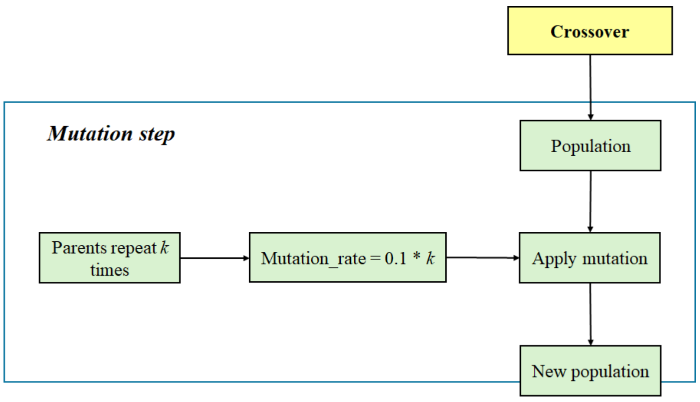


Fig. 7 Improved details of the Mutation step

Finally, we directly include the parent pair from the previous population into the next population. As a result, the new population reaches a size of 2^n, where n represents the number of hyperparameters that differ between the parents. This approach aims to mitigate the risk of the next population being of lower quality than the previous one, as there is no guarantee that the offspring will always perform better than their parents.

Thanks to improvements in critical steps of the algorithm, the populations diversity was maintained at a very high level, enabling the optimal hyperparameters to be found significantly faster compared to the traditional genetic algorithm. To further optimize training time and resources, we implemented an early stopping mechanism: if the parent pair remains unchanged for 15 consecutive generations, the best individual in the final population is considered to contain the optimal hyperparameters, and the algorithm terminates at that point.

2.6 Stacking ensemble learning

Stacking Ensemble Learning, also known as Stacked Generalization, is an algorithm within the Ensemble Learning group, similar to Bagging and Boosting, but with a unique approach to combining multiple machine learning models to improve predictive performance. While Bagging combines predictions from multiple models, often of the same type, trained on different subsets of the training dataset, and Boosting uses a sequence of typically similar models to iteratively correct errors, Stacking focuses on leveraging the strengths of different machine learning models and using a meta model to enhance the overall predictive power.

The Stacking method is typically divided into two levels: Level-0 Models (Base Models) and Level-1 Model (Meta Model). The Level-0 models are trained directly on the original dataset to produce output predictions. This output is then used as input for a meta model, the Level-1 model. The meta model learns how to combine the predictions from the base models to provide the best final result. The meta model does not learn directly from the original data but from the outputs of the base models. However, in some cases, the original data may be supplemented to provide additional information for the meta model. The strength of Stacking lies in its ability to combine complex and diverse models such as Decision Trees, SVMs, Neural Networks, or even other ensemble models like Random Forest or Gradient Boosting. Although the base models are often complex, the meta model is typically designed to be simple, such as using Logistic Regression for classification problems or Linear Regression for regression tasks, because the data the meta model needs to learn from is often reduced to either real values for regression tasks or probabilities of labels for classification tasks. To put it simply, Stacking is like having a group of friends, each an expert in a specific field. Instead of learning everything from scratch, you listen to each friend's insights about their area of expertise, then combine these insights to achieve a certain level of knowledge for your goal. This is essentially how Stacking works—maximizing the strengths of each model to create a more comprehensive solution to the problem.

Although Stacking is a brilliant idea for combining machine learning models, it suffers from a significant drawback: susceptibility to overfitting. The primary cause lies in the method of constructing the training dataset for the meta model. If the base models are trained on the training set and then make predictions on the same training set to generate input data for the meta model, these predictions become "artificial" because the base models have already seen this data during training. This leads to the meta model not only learning from the predictions but also indirectly learning the original training samples, resulting in overfitting. For example, if one of the base models, such as an SVM, overfits the training data, the meta model is likely to overly rely on these flawed predictions. Consequently, the entire Stacking system is at risk of overfitting as well. To address the overfitting issue in Stacking, the k-fold cross-validation method is commonly applied. This approach splits the dataset into two main parts: the training set and the validation set, but with a unique twist-it creates multiple validation sets from the original data. Specifically, the dataset is divided into k equal parts (folds). The training process is carried out k times, each iteration referred to as a run. In each run, k - 1 folds are used to train the base models, while the remaining fold is used for predictions, serving as the validation set. These predictions, known as Out-of-Fold Predictions (OOF), are independent of the data on which the models were trained, ensuring objectivity and preventing the meta model from directly learning from the original training data. Once this process is complete, the predictions made by the base models on unseen parts of the dataset are combined to form the input dataset for the meta model. An illustration of the application of Stacking with the cross-validation method is shown in Figure 8.

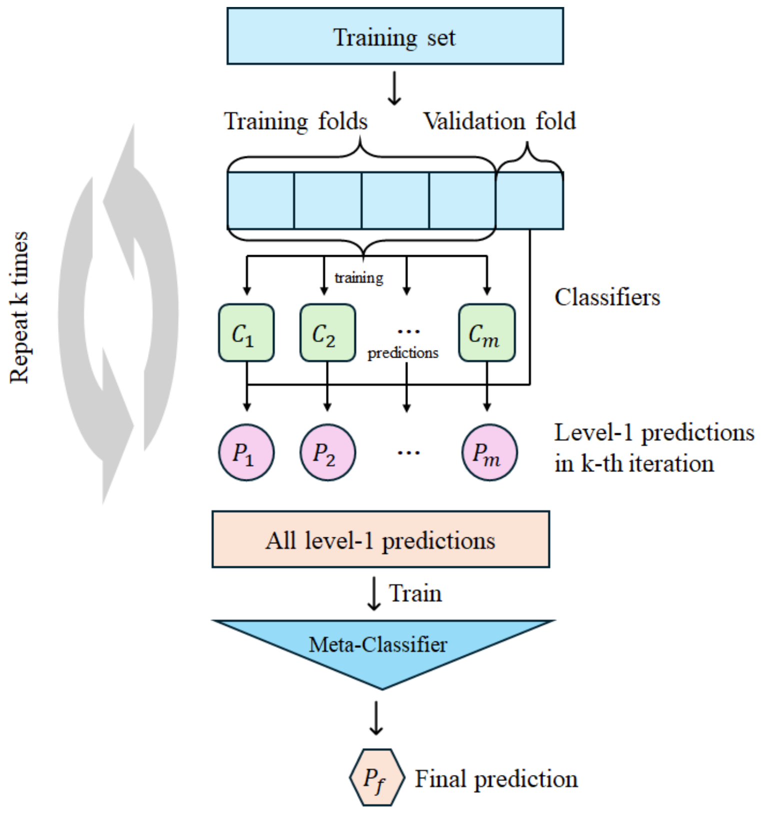


Fig. 8 Stacking Ensemble Learning with Cross-Validation Workflow

3 Experimental results

3.1 Experiment configs and evaluation metrics

During the experiments, we utilized a single P100 GPU with 16 GB of RAM to train the models. The system was powered by an Intel Xeon 2.20 GHz CPU with 4 virtual CPUs (vCPUs) and 32 GB of RAM, ensuring smooth data processing, visualization, and model execution. The operating system used was Linux with kernel version 5.15.109+, and all experiments were conducted in a Python 3.10.14 environment. For the software setup, the primary libraries included PyTorch 2.4.0 for deep learning models, along with scikit-learn 1.2.2 and CatBoost 1.2.7 for classification tasks. All deep learning models were trained with a batch size of 32 and a random seed of 42, ensuring consistency during the split of the dataset into three subsets: train, validation, and test. The optimal learning rate and optimizer for the models were selected using a genetic algorithm.

In this study, we utilized evaluation metrics including Accuracy, F1-score, Precision, and Recall, each providing a unique perspective on the model's classification performance.

Accuracy is calculated as the ratio of correctly predicted samples to the total number of samples, and its formula is:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision measures the proportion of correctly predicted samples among all samples that the model predicted as belonging to a specific class. The formula is:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

Recall, also known as Sensitivity, measures the model's ability to correctly identify samples that truly belong to a specific class. The formula is:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

Where:

- TP (True Positive) is the number of positive samples correctly predicted as positive.
- TN (True Negative) is the number of negative samples correctly predicted as negative.
- FP (False Positive) is the number of negative samples incorrectly predicted as positive.
- FN (False Negative) is the number of positive samples incorrectly predicted as negative.

Finally, the F1-score is the average of the F1-scores across all classes, and it is used to balance between precision and recall:

$$\text{F1-score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{4}$$

We use F1-score as the primary performance evaluation metric for the models, as the training dataset has a significant class imbalance. The F1-score helps balance between precision and recall, providing an effective measure of performance in this scenario.

3.2 Results of Genetic algorithm

The GA algorithm ends, and the base models have the best hyperparameters for each model for the dataset used in this study. CNN has 5 convolution layers, 509 neurons, and approximately 0.000513 learning rate, optimizer is Nadam. LSTM has 2 hidden layers, 512 neurons, and approximately 0.01963 learning rate, optimizer is Adamax. CatBoost has approximately 0.482 learning rate, Depth is 9, regularization hyperparameter for Leaves (l2_leaf_reg) is 4 and the number of Splits for Continuous Features (border_count) is 47.

Figure 9, 10, 11 represent the best fitness across generations. The improvements to GA not only enhance its ability to find optimal hyperparameters but also demonstrate significant computational efficiency, as evidenced by the early stopping mechanism. We can see that the fitness of later generations tends to be better than previous generations, making it easier to find a set of hyperparameters, which can be easily seen in Figure 10. At the same time, it also saves computational resources when it is impossible to find a new, better set of hyperparameters, Figure 9 demonstrates the fitness stagnation during hyperparameters tuning for CatBoost. However, we still see abnormalities in the results of CatBoost and CNN. The general trend is that we still see that the fitness of the later generation is better than the previous generations, but it is not stable when the fitness value fluctuates quite strongly in CNN in Figure 11 and is smaller in CatBoost.

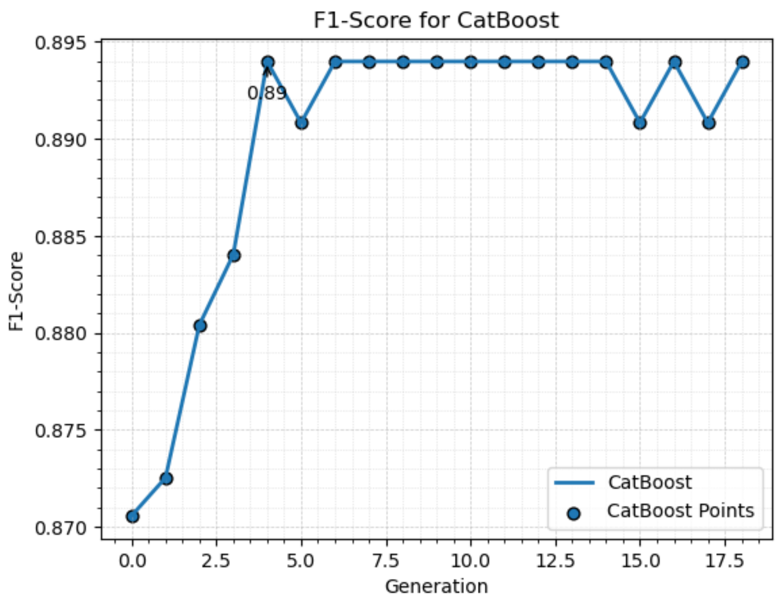


Fig. 9 Fitness score over generations when optimizing hyperparameters for CatBoost model

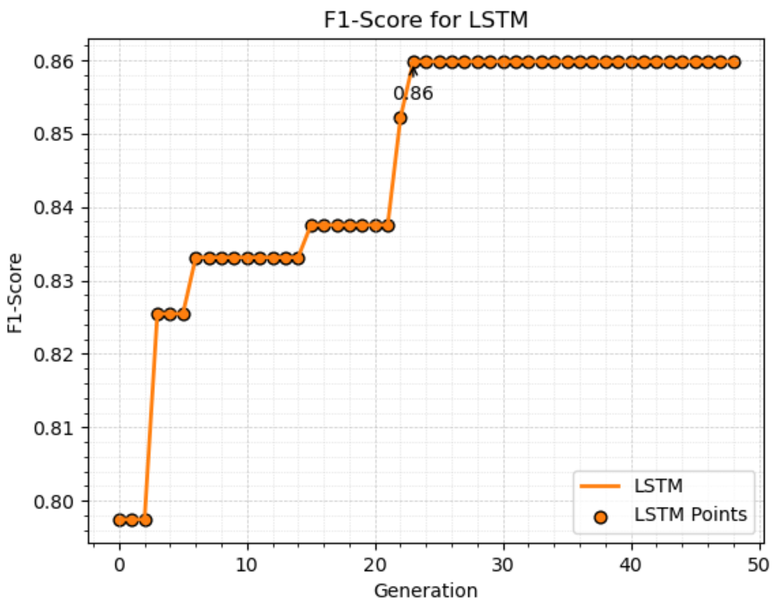


Fig. 10 Fitness score over generations when optimizing hyperparameters for LSTM model

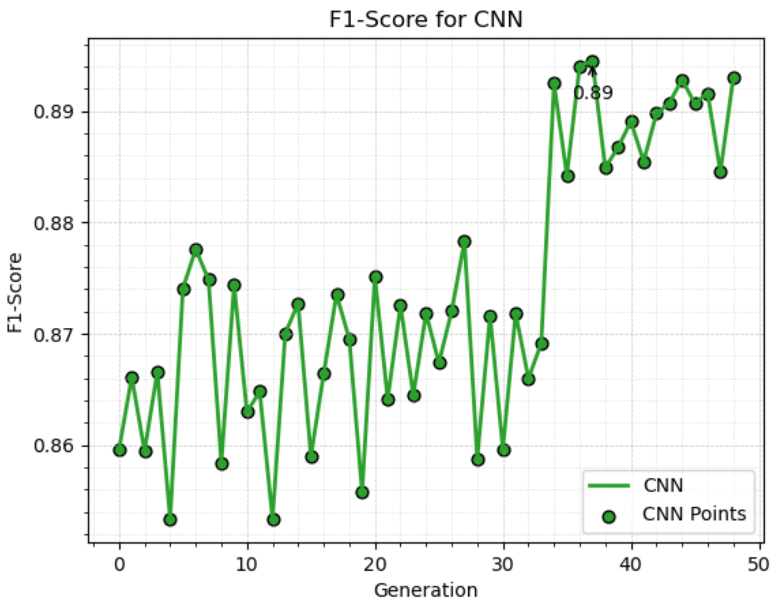


Fig. 11 Fitness score over generations when optimizing hyperparameters for CNN model

3.3 Results of stacking ensemble

After applying the Stacking ensemble learning method in section 2.6 to the training dataset, we constructed a new training dataset for the meta model. This dataset was organized into a DataFrame consisting of 16 columns: 15 feature columns representing the prediction probabilities of 3 base models across 5 labels and 1 column for the actual labels. Next, we trained the base models on the test set and collected the prediction probabilities for each label. These probabilities were aggregated and combined with the actual labels of the test set, forming a complete test set for the meta model. The training configuration for this phase included 20 epochs for the LSTM and CNN models, 5000 iterations for the CatBoost model, and hyperparameters optimized using a Genetic algorithm.

The comparison of models in Table 3 shows that the Stacking Ensemble model with Logistic Regression as the meta model outperforms all individual models optimized by the Genetic algorithm across most metrics: accuracy, F1-score, and precision. The prefix "GA_" before the model names indicates that these models are using hyperparameters optimized by the Genetic algorithm.

Focusing on the F1-score - the primary metric for performance comparison, GA_CatBoost achieved 92.28%, ranking second in performance, highlighting that CatBoost is one of the most robust standalone models on this dataset after optimization. However, compared to the Stacking ensemble, the performance of GA_CatBoost is still 0.88% lower. Nonetheless, GA_CatBoost achieved the highest Recall at 97.31%, demonstrating its superior ability to accurately identify positive labels compared to other models, even surpassing the combined performance of the three models using the Stacking method. GA_LSTM, despite being designed to handle time-series data effectively, achieved an F1-score of only 89.99%, significantly lower than GA_CatBoost and GA_CNN. However, the integration of information from LSTM with other models in the Stacking Ensemble contributed to the overall performance improvement, showing the additional value that LSTM brings in diversifying input information for the meta model. GA_CNN, with an F1-score of 91.38%, lies between LSTM and CatBoost, suggesting that CNN may not be fully suited to the characteristics of the data in this problem compared to CatBoost. Nevertheless, CNN, along with LSTM and CatBoost, still plays an important role in stacking by providing distinct feature representations, adding value to the final model. The Stacking ensemble model, with an F1-score of

Model	**Accuracy**	**F1-score**	**Precision**	**Recall**
GA_CatBoost	0.9853	0.9228	0.8838	**0.9731**
GA_LSTM	0.9821	0.8999	0.8704	0.9375
GA_CNN	0.9856	0.9138	0.9003	0.9311
Stacking (LR)	**0.9883**	**0.9316**	**0.9013**	0.9699

Table 3 Performance Evaluation of Models on Test Set

93.16%, demonstrates its superiority through its ability to combine information from multiple base models. This combination not only leverages the individual strengths of each model but also mitigates weaknesses, ensuring stable and consistent performance across metrics. These results affirm the strength of the Stacking method in enhancing overall performance compared to standalone models, while also highlighting the potential for further performance improvements if the base models are more effectively optimized in the future.

3.4 Comparative analysis

3.4.1 GA Optimization vs. Baseline Model Performance

To evaluate and compare the performance of baseline models with models optimized using the genetic algorithm, we analyzed the predictions of the baseline models on the test set. The results, presented in Table 4, show that the performance of the baseline models improved significantly through the optimization process using the genetic algorithm, demonstrating the potential of this method in enhancing classification effectiveness. The initial models, such as CatBoost, LSTM, and CNN, achieved F1 scores of 90.05%, 88.48%, and 88.19%, respectively. However, after applying the genetic algorithm to identify optimal hyperparameters, the baseline models showed impressive growth: the F1 score of CatBoost increased by 2.23%, LSTM by 1.51%, and CNN achieved the highest improvement with a 3.19% increase. This improvement not only highlights the optimization capability of the algorithm but also underscores the importance of hyperparameter tuning in achieving higher accuracy and overall effectiveness.

Model	**Accuracy**	**F1-score**	**Precision**	**Recall**
Base CatBoost	0.9807	0.9005	0.8524	0.9675
GA_CatBoost	0.9853	0.9228	0.8838	0.9731
Base LSTM	0.9766	0.8848	0.8498	0.9295
GA_LSTM	0.9821	0.8999	0.8704	0.9375
Base CNN	0.9771	0.8819	0.8578	0.9107
GA_CNN	0.9856	0.9138	0.9003	0.9311

Table 4 Comparing GA-Optimized Model and Baseline Model Performance on Test Set

3.4.2 Comparison with State-of-the-Art Technologies

The results presented in Table 5 show that the GA combined with Stacking method achieves an F1-score of 0.9316, confirming its competitiveness in the ECG classification task on the MIT-BIH dataset. Although it has not surpassed the most advanced methods such as DCETEN 1D by Jiang, Fan, et al. [29] with an F1-score of 0.9967, or BiTrans with Context-Aware Loss by El-Ghaish, H, et al. [30] with an F1-score of 0.9426, this method still outperforms many other approaches. We did not use data balancing techniques such as SMOTE or label-specific weighting, which shows that the model can maintain high performance without the need for additional adjustments through class balancing.

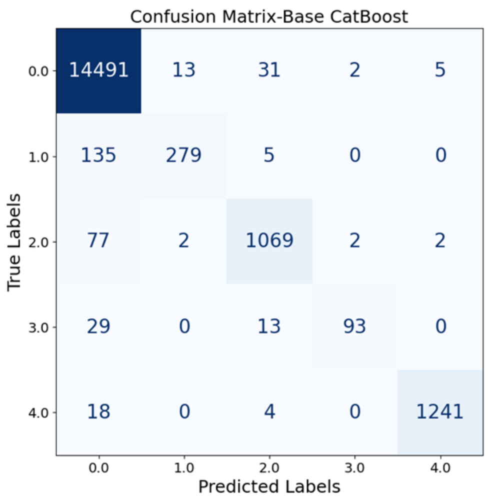
Works	**Year**	**Approach**	**Balanced**	**F1-score**
Jiang, Fan, et al. [29]	2024	DCETEN(1D)	True	**0.9967**
El-Ghaish, H, et al. [30]	2024	Bidirectional Transformer (BiTrans) Context-Aware Loss (CAL)	True	0.9426
Farag, M. M. [31]	2023	Matched Filter-Based CNN	True	0.9217
Jahan, Masud Shah, et al. [32]	2022	AdaBoost, Leave-One-Group-Out Cross Validation (LOGO-CV)	False	0.8801
This paper	2025	GA + Stacking	False	0.9316

Table 5 Comparison of ECG Classification Techniques

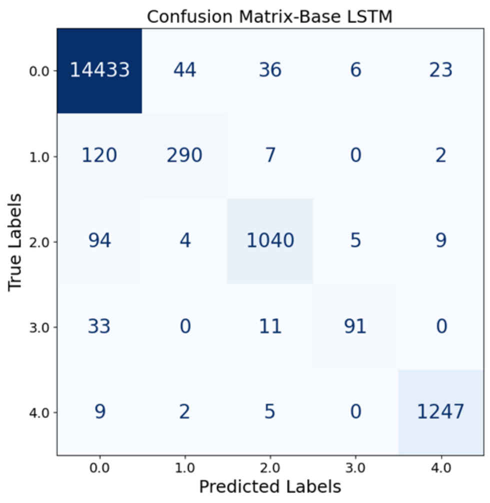
4 Discussion

From the experimental results, we can verify the benefits of GA improvements. Not only does it help find the optimal hyperparameters easily, but it also saves computational resources by stopping the algorithm early when there are no better new hyperparameters. However, there are also abnormalities when the hyperparameters optimization process in CatBoost and CNN is unstable. Through several reruns of GA to investigate these abnormalities, we realized that LSTM is stable in the hyperparameters optimization process, CatBoost appears in this situation several times in some experiments, and CNN always appears in this situation in our experiments. We realized that this may be due to the sequential nature of LSTM, which makes it unable to take full advantage of the parallel processing capability of the GPU but limits randomness, making the hyperparameters optimization process more stable. Meanwhile, CNN and CatBoost take advantage of the parallel processing capabilities of GPUs better, which leads to parallel computations with small inconsistencies. Bootstrapping, random splitting (CatBoost) or inappropriate batch size (CNN) also increase the variability of the results. These are just assumptions we made after several experiments, and this issue still needs to be studied in detail.

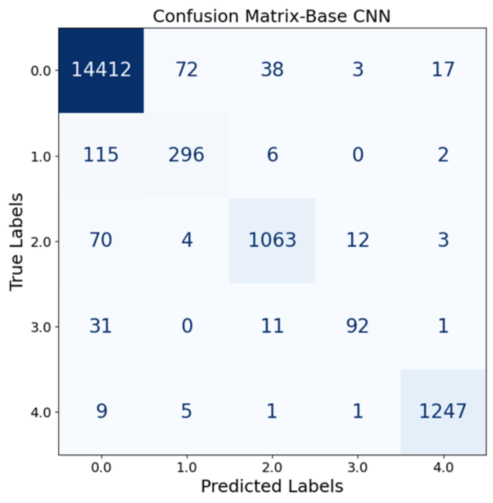
Figures 12 and 13 illustrate the confusion matrices of baseline models and models optimized with the Genetic Algorithm (GA) on the test set. The GA-optimized models exhibit significantly reduced error rates across most classes, reflecting substantial improvements in classification performance. While both baseline and optimized models achieve high overall recognition accuracy, they share a common challenge in accurately classifying labels 1.0 (Supraventricular premature contraction) and 3.0 (Fusion of ventricular and normal). Prediction accuracy for these labels ranges from approximately 74.37% (Base CNN) to 80.17% (Base CatBoost) and from 76.60% (GA_LSTM) to 86.93% (Stacking (LR)). This difficulty is likely due to the severe class imbalance in the dataset, which hampers the models' ability to learn and distinguish less frequent labels. A potential solution to address this issue could involve using data balancing techniques such as oversampling or undersampling, or adjusting the loss function weights to mitigate the impact of imbalance—approaches we are considering for future work.



a, Confusion matrix of Base CatBoost model

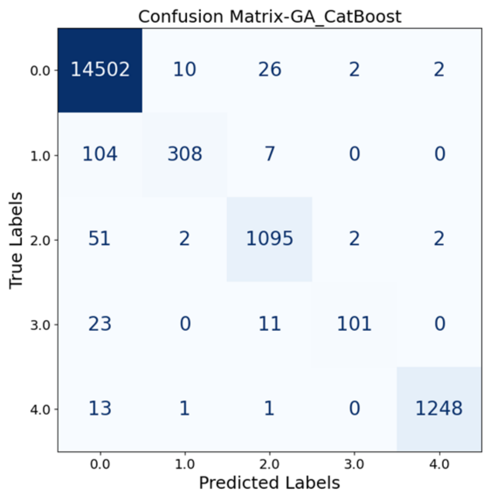


b, Confusion matrix of Base LSTM model

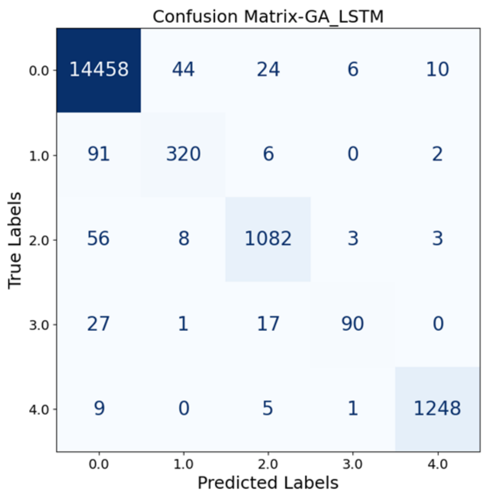


c, Confusion matrix of Base CNN model

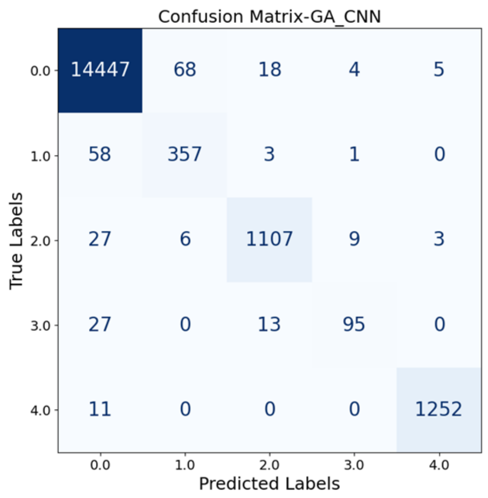
Fig. 12 Confusion Matrix of Baseline Models for Test Set. (a) represents Confusion Matrix of Base CatBoost; (b) represents Confusion Matrix of Base LSTM; (c) represents Confusion Matrix of Base CNN; (d) represents Confusion Matrix of Stacking (LR)



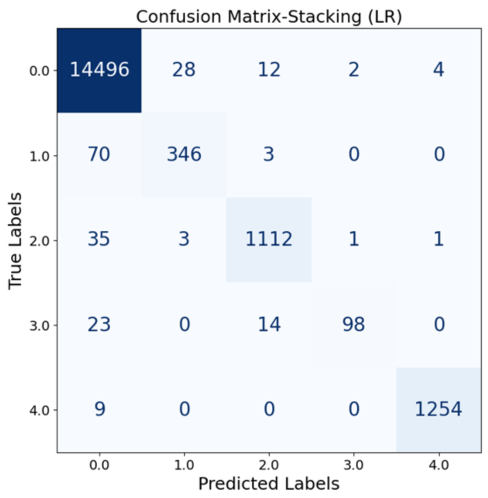
a, Confusion matrix of GA_CatBoost model



b, Confusion matrix of GA_LSTM model



c, Confusion matrix of GA_CNN model



d, Confusion matrix of Stacking (LR) model

Fig. 13 Confusion Matrix of GA-Optimized Model for Test Set. (a) represents Confusion Matrix of GA_CatBoost; (b) represents Confusion Matrix of GA_LSTM; (c) represents Confusion Matrix of GA_CNN; (d) represents Confusion Matrix of Stacking (LR)

5 Conclusion

In summary, we have experimented with and applied the Genetic algorithm to optimize the hyperparameters for various classification models, including CatBoost, CNN, and LSTM. During this process, we identified key challenges in using Genetic algorithms for hyperparameters optimization and proposed practical solutions to address them, such as incorporating mechanisms that ensure extremely high diversity in the algorithm. Additionally, we leveraged the Stacking ensemble technique to further enhance model performance, yielding significant improvements and promising results.

Competing Interests

The authors have no competing interests to declare that are relevant to the content of this article.

Data availability

The data and code that support the findings of this study are available from the corresponding author upon reasonable request.

Ethics approval

It is not applicable. The study does not include any medical tests, treatments, or interventions.

Acknowledgment

This research was supported by The VNUHCM-University of Information Technology's Scientific Research Support Fund.

References

[1] World Health Organization https://www.who.int/data/gho

[2] Zabihi, F., Safara, F., Ahadzadeh, B.: An electrocardiogram signal classification using a hybrid machine learning and deep learning approach. Healthcare Analytics **6**, 100366 (2024)

[3] Sharma, M., Singh, S., Kumar, A., San Tan, R., Acharya, U.R.: Automated detection of shockable and non-shockable arrhythmia using novel wavelet-based ecg features. Computers in Biology and Medicine **115**, 103446 (2019) https://doi.org/10.1016/j.compbiomed.2019.103446

[4] Papadogiorgaki, M., Venianaki, M., Charonyktakis, P., Antonakakis, M., Tsamardinos, I., Zervakis, M.E., Sakkalis, V.: Heart rate classification using ecg signal processing and machine learning methods. In: 2021 IEEE 21st International Conference on Bioinformatics and Bioengineering (BIBE), pp. 1–6 (2021). https://doi.org/10.1109/BIBE52308.2021.9635462

[5] Aziz, S., Ahmed, S., Alouini, M.-S.: Ecg-based machine-learning algorithms for heartbeat classification. Scientific reports **11**(1), 18738 (2021)

[6] Pham, T.-H., Sree, V., Mapes, J., Dua, S., Lih, O.S., Koh, J.E., Ciaccio, E.J., Acharya, U.R.: A novel machine learning framework for automated detection of arrhythmias in ecg segments. Journal of Ambient Intelligence and Humanized Computing, 1–18 (2021)

[7] Rajpurkar, P., Hannun, A.Y., Haghpanahi, M., Bourn, C., Ng, A.Y.: Cardiologist-level arrhythmia detection with convolutional neural networks. CoRR **abs/1707.01836** (2017) 1707.01836

[8] Acharya, U.R., Oh, S.L., Hagiwara, Y., Tan, J.H., Adam, M., Gertych, A., Tan, R.S.: A deep convolutional neural network model to classify heartbeats. Computers in Biology and Medicine **89**, 389–396 (2017) https://doi.org/10.1016/j.compbiomed.2017.08.022

[9] Kachuee, M., Fazeli, S., Sarrafzadeh, M.: ECG heartbeat classification: A deep transferable representation. CoRR **abs/1805.00794** (2018) 1805.00794

[10] Yildirim, O., Baloglu, U.B., Tan, R.-S., Ciaccio, E.J., Acharya, U.R.: A new approach for arrhythmia classification using deep coded features and lstm networks. Computer Methods and Programs in Biomedicine **176**, 121–133 (2019) https://doi.org/10.1016/j.cmpb.2019.05.004

[11] Ji, W., Zhu, D.: Ecg classification exercise health analysis algorithm based on gru and convolutional neural network. IEEE Access **12**, 59842–59850 (2024) https://doi.org/10.1109/ACCESS.2024.3392965

[12] Jiang, F., Xiao, J., Liu, L., Wang, C.: Dceten: A lightweight ecg automatic classification network based on transformer model. Digital Communications and Networks (2024) https://doi.org/10.1016/j.dcan.2024.11.003

[13] Zabihi, F., Safara, F., Ahadzadeh, B.: An electrocardiogram signal classification using a hybrid machine learning and deep learning approach. Healthcare Analytics **6**, 100366 (2024) https://doi.org/10.1016/j.health.2024.100366

[14] Elsayad, A., Nassef, A., Baareh, A.K.: Cardiac arrhythmia classification using boosted decision trees. International Review on Computers and Software **10**, 280–289 (2015) https://doi.org/10.15866/irecos.v10i3.5359

[15] Rabinezhadsadatmahaleh, N., Khatibi, T.: A novel noise-robust stacked ensemble of deep and conventional machine learning classifiers (nrse-dcml) for human biometric identification from electrocardiogram signals. Informatics in Medicine Unlocked **21**, 100469 (2020) https://doi.org/10.1016/j.imu.2020.100469

[16] Kunwar, P., Choudhary, P.: A stacked ensemble model for automatic stroke prediction using only raw electrocardiogram. Intelligent Systems with Applications **17**, 200165 (2023) https://doi.org/10.1016/j.iswa.2022.200165

[17] Din, S., Qaraqe, M., Mourad, O., Qaraqe, K., Serpedin, E.: Ecg-based cardiac arrhythmias detection through ensemble learning and fusion of deep spatial–temporal and long-range dependency features. Artificial Intelligence in Medicine **150**, 102818 (2024) https://doi.org/10.1016/j.artmed.2024.102818

[18] Khezripour, H., Mozaffari, S.P., Reshadi, M., Zarrabi, H.: Classification of electrocardiogram signals using deep learning based on genetic algorithm feature extraction. Biomedical Physics Engineering Express **9**(5), 055014 (2023) https://doi.org/10.1088/2057-1976/acdc2a

[19] Zhao, N., Li, X., Ma, Y., Wang, H., Lee, S.-J., Wang, J.: Improved stacked ensemble with genetic algorithm for automatic ecg diagnosis of children living in high-altitude areas. Biomedical Signal Processing and Control **87**, 105506 (2024) https://doi.org/10.1016/j.bspc.2023.105506

[20] Kachuee, M., Fazeli, S., Sarrafzadeh, M.: Ecg heartbeat classification: A deep transferable representation. In: 2018 IEEE International Conference on Healthcare Informatics (ICHI), pp. 443–444 (2018). https://doi.org/10.1109/ICHI.2018.00092

[21] Shanmugam, S., Dharmar, S.: Implementation of a non-linear svm classification for seizure eeg signal analysis on fpga. Engineering Applications of Artificial Intelligence **131**, 107826 (2024)

[22] Li, Z., Zhang, H.: Fusing deep metric learning with knn for 12-lead multi-labelled ecg classification. Biomedical Signal Processing and Control **85**, 104849 (2023)

[23] Peng, R., Lu, Z.: Semiparametric averaging of nonlinear marginal logistic regressions and forecasting for time series classification. Econometrics and Statistics **31**, 19–37 (2024)

[24] Deng, Z., Han, T., Cheng, Z., Jiang, J., Duan, F.: Fault detection of petrochemical process based on space-time compressed matrix and naive bayes. Process Safety and Environmental Protection **160**, 327–340 (2022)

[25] Zalewski, W., Silva, F., Maletzke, A.G., Ferrero, C.A.: Exploring shapelet transformation for time series classification in decision trees. Knowledge-Based Systems **112**, 80–91 (2016)

[26] Yang, M., Olivera, F.: Classification of watersheds in the conterminous united states using shape-based time-series clustering and random forests. Journal of Hydrology **620**, 129409 (2023)

[27] Zhao, J., Liang, H., Li, S., Yang, Z., Wang, Z.: Matrix-based vs. vector-based linear discriminant analysis: A comparison of regularized variants on multivariate time series data. Information Sciences **654**, 119872 (2024)

[28] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: Catboost: unbiased boosting with categorical features. Advances in neural information processing systems **31** (2018)

[29] Jiang, F., Xiao, J., Liu, L., Wang, C.: Dceten: A lightweight ecg automatic classification network based on transformer model. Digital Communications and Networks (2024)

[30] El-Ghaish, H., Eldele, E.: Ecgtransform: Empowering adaptive ecg arrhythmia classification framework with bidirectional transformer. Biomedical Signal Processing and Control **89**, 105714 (2024)

[31] Farag, M.M.: A tiny matched filter-based cnn for inter-patient ecg classification and arrhythmia detection at the edge. Sensors **23**(3), 1365 (2023)

[32] Jahan, M.S., Mansourvar, M., Puthusserypady, S., Wiil, U.K., Peimankar, A.: Short-term atrial fibrillation detection using electrocardiograms: A comparison of machine learning approaches. International Journal of Medical Informatics **163**, 104790 (2022)