Classification of Cardiac Arrhythmia Using GA Stacking in Ensemble Learning

Nguyen Le Vy[1], Nguyen Thi Mai Trinh[1], Nguyen Tat Bao Thien[2]

[1]Faculty of Information Science and Engineering, University of Information Technology, Vietnam National University, Ho Chi Minh City, Vietnam.

[1]Faculty of Computer Science, University of Information Technology, Vietnam National University, Ho Chi Minh City, Vietnam.

Contributing authors: 21522811@gm.uit.edu.vn; 21522718@gm.uit.edu.vn; thienntb@uit.edu.vn;

Abstract

In the era of rapid advancements in automation and artificial intelligence, applying modern techniques to healthcare presents opportunities to enhance diagnostic and treatment efficiency. To explore this field further, our team developed a hybrid model to classify arrhythmia types using the Cardiac Arrhythmia Database [1]. This dataset includes ECG phase characteristics across a cardiac cycle. For arrhythmia classification, we employed five base models (Random Forest, Gradient Boosting, XGBoost, SVC and CNN), a meta-model (LogisticRegression), followed by GA/ Grid Search to optimize classification results. The ensemble models show promising results, particularly in addressing challenges posed by imbalanced data. The accuracy of the ensemble models exceeds 0.70, and the model also improves the detection of arrhythmia types, as demonstrated by higher Precision, Recall, and F1-score compared to some individual models. However, the model's performance still falls short of expectations, suggesting room for further refinement. These results highlight the potential of ensemble models in classifying physiological signals and open new avenues for developing automated medical diagnostic systems that could significantly enhance healthcare outcomes.

Keywords: Random Forest, Gradient Boosting, XGBoost, SVC, CNN, Logistic Regression, GA, GridSearch

1 Introduction

Arrhythmia classification is crucial in healthcare due to the complexity and variability of ECG signals, which requires advanced analytical methods to ensure diagnostic precision. To address this challenge, we developed a Stacked Ensemble model aimed at enhancing classification performance on Cardiac Arrhythmia Database. This dataset contains intricate features of ECG signal, requiring a model capable of precise detection and classification.

The process is illustrated in the Fig.1.

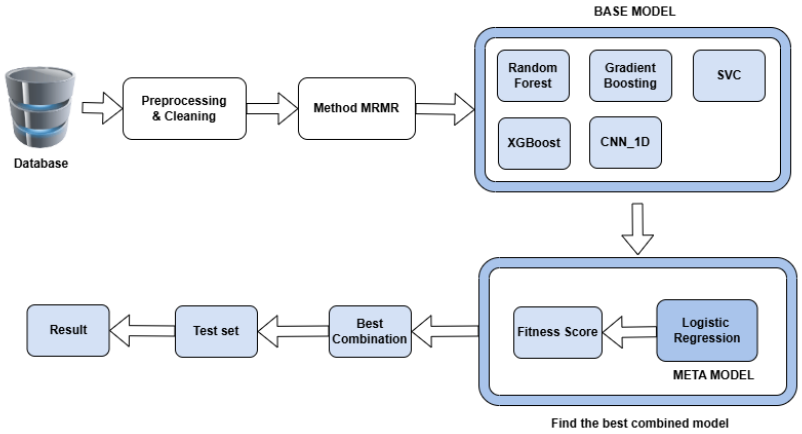

Fig. 1 The arrhythmia classification process using GA/ Grid Search. This figure illustrates the process of selecting the best ensemble model combination using GA/ Grid Search. Subsequently, experiments are performed to assess the performance of the models combined using these two methods.

Our approach to arrhythmia classification employs an ensemble model that integrates base models such as Random Forest, Gradient Boosting, SVC, XGBoost, and CNN. Logistic Regression serves as a meta-model. The biggest challenge is how to find out the best model combination. Therefore, we experimented with a genetic algorithm and a grid search based on fitness scores 5.2.5. Each method produces two different ensemble models (models with highest fitness), then we evaluate these ensemble models on the test set using evaluated using metrics such as Accuracy, Precision, F1-Score, and Recall.

The results indicated that the ensemble approach, optimized with GA and Grid Search, showed promising outcomes, with space for further fine-tuning and development. This finding not only highlights the effectiveness of structured ensemble methods in physiological signal analysis but also opens new opportunities for the development of automated medical diagnostic systems.

Our report is organized into six sections. Section 1 introduces the problem and the research objectives. Related work is discussed in Section 2. Section 3 details the dataset. Section 4 presents the methodology. The experimental pipeline and results are

discussed in Section 5. Finally, section 6 concludes the report with future development directions.

2 Related Work

In the field of healthcare, the detection and diagnosis of cardiovascular issues, particularly arrhythmias, have long been studied using machine learning (ML) and artificial intelligence (AI).

Previous research has predominantly relied on single models for arrhythmia classification. For instance, "A Study on Arrhythmia via ECG Signal Classification Using the Convolutional Neural Network"[2] employed CNNs, renowned for their capability to automatically extract features from ECG signals. Another approach, illustrated in "LSTM-Based Arrhythmia Classification in Electrocardiogram Signals" by Nandita and Mandala [3], leveraged LSTMs to classify eight arrhythmia types, showcasing their ability to capture sequential relationships in long ECG signals.

However, relying solely on individual models limits the ability to harness the strengths of multiple ML techniques, especially for complex datasets like ECGs. To address this, ensemble learning techniques such as bagging, boosting, and stacking have been introduced to combine the strengths of diverse models, improving classification performance. Most recently, Mohapatra et al. [4] developed an arrhythmia detector based on the stacking technique, utilizing a Deep Learning Bagging Model as the base learner. The meta-learner in this study was a combination of CNN-LSTM and RRHOS-LSTM models, highlighting the versatility of stacking in combining deep learning architectures. This chronological review underscores the evolution and continued relevance of the stacking technique in the context of arrhythmia detection. Similarly, N. Zhao's work [5] on ECG diagnosis used a GA-Stacking ensemble with XGBoost, LightGBM, and CatBoost, achieving F1 scores of 0.8710 and 0.8459 for ECG abnormality detection and arrhythmia classification, respectively. Although the stacked model slightly sacrificed precision, it significantly enhanced recall, improving abnormality detection, demonstrating a balanced and superior performance. On the other hand, Dalal and Ingale [6] and Yakut and Bolat [7] proposed arrhythmia detection methods based on the ensemble of Multi-Layer Perceptron and Random Forest. One distinction between the two studies is the use of meta-learners, where Yakut and Bolat [7] incorporated a meta-learner while Dalal and Ingale [6] did not.

Recent research highlights the effectiveness of ensemble methods, particularly GA-Stacking. Studies like "Early Risk Prediction of Diabetes Based on GA-Stacking" [8] and "Improved Stacked Ensemble with Genetic Algorithm for Automatic ECG Diagnosis of Children Living in High-Altitude Areas" [9] demonstrate its potential. Zhao et al. [9] introduces an interesting architecture based on Genetic Algorithm (GA). In their experiments, the authors utilized GA to search for the best combination model from the results of base models, thereby significantly improving the overall performance. Therefore, our research team has developed an ensemble model approach using Genetic Algorithms (GA) to conduct the experiment.

Regarding the base models, Iyer et al. [10] conducted experiments on the Cardiac Arrhythmia Database, testing and evaluating the performance of various machine

learning models for the task of cardiac arrhythmia classification. We will build upon these results and select the top 4 performing machine learning models, which include: Random Forest, Gradient Boosting, SVC, and XGBoost, as the base models for our task. Additionally, according to Chen et al. [11], we observed that deep learning models also delivered performance on par with machine learning models in classifying heart rhythms. Specifically, the CNN model used by the authors achieved impressive results. To optimize performance, we will combine 5 base models, including CNN and the four aforementioned machine learning models.

The paper "A Stacking Classifiers Model for Detecting Heart Irregularities and Predicting Cardiovascular Disease" [12] also conducted experiments on the Cardiac Arrhythmia Database with the goal of classifying heart rhythms, similar to our work. This study experimented with 12 machine learning models as meta-learners, with machine learning models as base learners, and the results showed that Logistic Regression (LR) achieved high performance, with accuracy, precision, and sensitivity all exceeding 0.80. Similarly, Zaini and Awang [13] found that the logistic regression meta-classifier yielded a result of 90.16%, outperforming all base classifiers. Based on these findings, we have decided to use LR as the meta-learner for our problem.

Building on these advancements, our project applies GA/ Grid Search to optimize base model classifications (Random Forest, Gradient Boosting, SVC, XGBoost and CNN), with meta-model (Logistic Regression) aiming to develop a more effective prediction system for the detection of arrhythmias. By integrating the strengths of multiple base models, our approach aims to achieve improved diagnostic accuracy and efficiency in identifying arrhythmias.

3 Dataset

3.1 Cardiac Arrhythmia Database

The team utilized the Cardiac Arrhythmia Database, available on UCI [1]. Its primary goal is to distinguish, identify, and classify different types of arrhythmias. The dataset comprises 452 patient records with 279 features, including attributes like age, gender, height, weight, heart rate, and ECG parameters such as Q, R, S waves on DI, DII, DIII, and other channels. Approximately 0.33% of the feature values are missing.

The target variable, 'diagnosis', encompasses a total of 16 distinct labels, each representing a specific clinical condition: Label 01: Normal ECG; Label 02: Ischemic changes (Coronary Artery Disease); Label 03: Old Anterior Myocardial Infarction; etc. Each label corresponds to a unique diagnostic category, capturing variations in electrocardiographic patterns for precise classification. The dataset exhibits class imbalance, with label 01 being the most common. Additionally, labels 11, 12, and 13 have no samples. While some patients's ECGs may display characteristics of multiple arrhythmias, the dataset assumes that each patient only has one type of arrhythmia.

The dataset provides rich diagnostic information, enabling the development of machine learning models to classify and detect anomalies accurately.

3.2 Data Preprocessing

The preprocessing steps included:

- Handling missing values.
- Converting data types for attributes like 'T', 'heart-rate', 'P', 'QRST', and 'J'.
- Revising target labels: Removed labels without values (11, 12, 13 corresponding to atrioventricular blocks grade 1–3).
- The label values were decreased by 1 unit: 1 became 0, 2 became 1, and so on.

After cleaning, the dataset retained 452 rows and 279 features, all cleaned for arrhythmia classification. The target variable 'diagnosis' was reduced from 16 labels to 13 labels:

- Label '0': Normal ECG
- Label '1': Ischemic changes (Coronary Artery Disease)
- Label '2': Old Anterior Myocardial Infarction
- Label '3': Old Inferior Myocardial Infarction
- Label '4': Sinus Tachycardia
- Label '5': Sinus Bradycardia
- Label '6': Ventricular Premature Contraction (PVC)
- Label '7': Supraventricular Premature Contraction (PVC)
- Label '8': Left Bundle Branch Block
- Label '9': Right Bundle Branch Block
- Label '10': Left Ventricular Hypertrophy
- Label '11': Atrial Fibrillation or Flutter
- Label '12': Other conditions

The dataset suffers from a severe class imbalance Fig.2, with class 0 dominating, accounting for over 240 data points, while other classes such as 6, 7, and 10 have fewer than 10 data points each. To mitigate the negative impact of class imbalance on model performance, the team employed a label weighting approach.

3.3 Input Feature selection - mRMR

The mRMR (Maximum Relevance Minimum Redundancy)[14] method is used to select the optimal features from a dataset based on two main criteria:

- Maximum Relevance: This criterion selects features that are most relevant to the target variable (label). The relevance is typically measured using mutual information (MI), which quantifies the relationship between a feature and the label. It is calculated using the following formula:

$$Relevance(X_i, y) = \frac{1}{|S|} \sum_{X_i \in S} I(X_i; y)$$

Where $I(X_i; y)$ is the mutual information between the feature X_i and the target variable y, and S is the set of features.

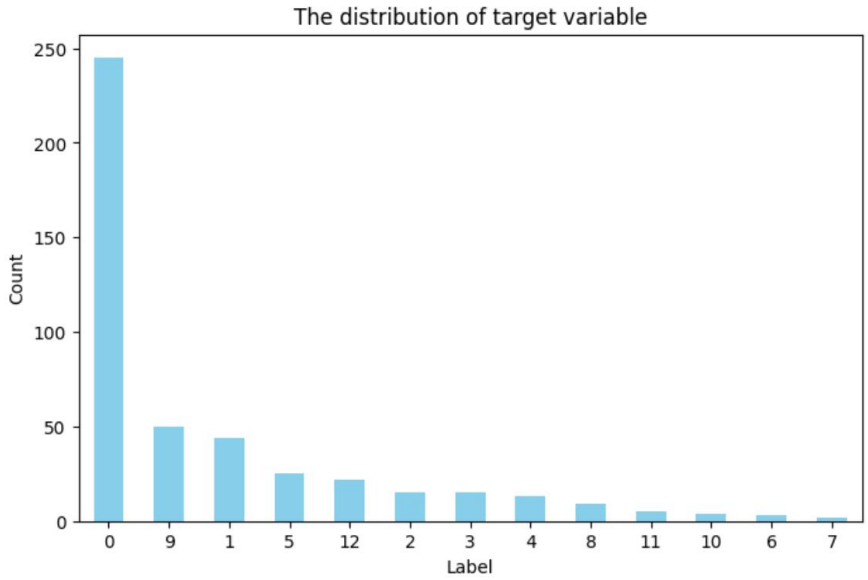

Fig. 2 The 'diagnosis' label distribution chart. This figure illustrates the number of data points allocated to each label in the dataset.

- Minimum Redundancy: This criterion ensures that the selected features are not redundant, meaning that features with similar information are avoided. The redundancy between two features X_i and X_j is also measured using mutual information:

$$Redundancy(X_i, X_j) = \frac{1}{|S|^2} \sum_{X_i, X_j \in S} I(X_i; X_j)$$

Where $I(X_i; X_j)$ is the mutual information between the two features X_i and X_j, and S is the set of features.

The value of Mutual Information between two discrete variables X and y is calculated using the following formula:

$$I(X; Y) = \sum_{x \in X, y \in Y} P(x, y) \log \frac{P(x, y)}{P(x)P(y)}$$

Where: $P(x, y)$ is the joint probability of X and Y; $P(x)$ and $P(y)$ are the marginal probabilities of X and Y, respectively.

We select the optimal features based on the mRMR objective function (MIQ - Mutual Information Quotient). The mRMR method optimizes the objective function by balancing relevance and redundancy, typically represented as follows:

$$J(X_k) = \frac{Relevance(X_k, y)}{\frac{1}{|S|} \sum_{X_j \in S} I(X_k; X_j)}$$

Where:

- $Relevance(X_k, y)$ is the relevance of feature X_k with respect to the label Y,
- $I(X_k; X_j)$ is the redundancy between feature X_k and other features in the set S.

The mRMR method from the pymrmr library was applied to the dataset with the target variable 'diagnosis'. As a result, the top 100 features were selected from the 279 available in the Cardiac Arrhythmia Database. These selected features were subsequently used as inputs for the model-building process.

Minimizing unimportant features helps the model focus on the key factors, improving accuracy and reducing the risk of overfitting. Additionally, it enhances computational efficiency, simplifies the model, and makes deployment and maintenance easier.

4 Methodology

The dataset was split into a train set and a test set in a 7:3 ratio to ensure robust evaluation of the models. After, we present an ensemble approach for arrhythmia classification, integrating base models such as Random Forest, Gradient Boosting, SVC, XGBoost, and CNN. Logistic Regression is used as a meta-model to evaluate different combinations of these base models on the train set, with fitness scores 5.2.5. To identify the optimal model combination, we employed two methods: GA and Grid Search, running them simultaneously to compare their effectiveness and pinpoint the best solution (highest fitness score). Each method produces two different ensemble models, then we evaluate these ensemble models on the test set using evaluation metrics.

4.1 Base Models

4.1.1 Random Forest

Random Forest[15] is an ensemble learning algorithm that combines multiple decision trees to form a "forest." Each tree is trained on a random subset of the data using a technique called bagging (Bootstrap Aggregating), which helps mitigate overfitting and enhances generalization.

The model splits data at each node based on the most informative features, and the final output is determined by majority voting (for classification) or averaging the predictions (for regression).

Random Forest is highly effective in handling missing values, scaling issues, and high-dimensional data. It is commonly used in applications such as fraud detection, medical diagnosis, and recommendation systems. While it can be computationally expensive for very large datasets, its strong performance and robustness make it a popular choice for a wide range of machine learning tasks.

4.1.2 Gradient Boosting

The term Gradient Boosting originates from Friedman's paper "Greedy Function Approximation: A Gradient Boosting Machine"[16]. This powerful machine learning algorithm combines weak learners to form a strong predictive model through boosting. It works by sequentially constructing decision trees, each correcting the errors of

the previous one, guided by a loss function that minimizes overall errors. This additive approach optimizes differentiable loss functions and fits regression trees on the negative gradient of the loss, such as binary or multiclass log loss. In binary classification, only one regression tree is induced.

The method excels at handling non-linear data and delivers high performance in both classification and regression tasks. Parameter tuning and early stopping help mitigate overfitting risks.

Gradient Boosting is widely used in fields like text classification, financial risk prediction, fraud detection, and other areas requiring high accuracy.

4.1.3 SVC

Support Vector Classifier (SVC)[17] is a supervised machine learning algorithm designed for classification tasks. Its primary goal is to identify the optimal hyperplane that separates data points from different classes, ensuring robust performance for both binary and multiclass problems.

SVC works by mapping input data into a higher-dimensional space using kernel functions, allowing it to find a linear hyperplane that separates classes. It focuses on the nearest data points, called support vectors, to maximize the margin between classes. This approach enables SVC to handle both linear and nonlinear datasets effectively.

The algorithm is robust in high-dimensional spaces and performs well on small to medium-sized datasets. By utilizing kernels such as linear, polynomial, and radial basis functions (RBF), SVC can model complex relationships in data while avoiding overfitting.

SVC is widely used in applications such as: text classification, image recognition, spam detection, handwriting analysis, bioinformatics, and anomaly detection, making it a versatile tool for solving real-world classification challenges.

4.1.4 XGBoost

XGBoost (Extreme Gradient Boosting)[18] is an advanced implementation of Gradient Boosting (4.1.2), designed for enhanced computational efficiency and predictive performance.

Similar to Gradient Boosting, XGBoost builds decision trees iteratively, but incorporates optimizations such as reduced tree complexity, parallel processing, and shrinkage-based learning. These improvements enable XGBoost to excel at handling large and complex datasets with exceptional speed and accuracy.

XGBoost stands out for its automatic handling of missing values, built-in regularization to prevent overfitting, and high computational efficiency. As a result, it has become a go-to algorithm in data science, frequently applied to tasks such as fraud detection, ranking, weather forecasting, and more. It is also a favorite tool in global data science competitions.

4.1.5 Convolutional Neural Network 1D (CNN)

The Convolutional Neural Network (CNN)[19][20] is one of the most remarkable architectures in artificial neural networks (ANN). CNN is specifically designed to

process one-dimensional or sequential signal data. CNN has multiple layers; including convolutional layer, non-linearity layer, pooling layer, and fully-connected layer. Convolutional and fully connected layers contain parameters, whereas non-linearity and pooling layers do not. Convolutional layers use filters to extract important features from sequential data, while pooling layers reduce data dimensions to enhance computational efficiency.

CNNs automatically learn essential features from data, minimizing the need for manual feature engineering. They also reduce the number of parameters, thus mitigating overfitting risks.

4.2 Meta Model

Stacking is particularly useful when multiple models perform well but in different ways, resulting in uncorrelated or weakly correlated predictions. In stacking, multiple base models are trained on the data, and their predictions are combined by a meta-model. The dataset is split into training and validation subsets, with the base models trained on the training data and the meta-model trained on predictions from the base models using the validation data. For this paper's classification task, the inputs to the meta-model are the probabilities from the base models.

To prepare the training data for the meta-model, k-fold cross-validation is used for the base models, and the out-of-fold predictions serve as training data for the meta-model. Once prepared, the meta-model is trained independently, while the base models are trained on the full dataset.

As mentioned in Section 2, we chose Logistic Regression (LR) as the meta-model for our problem. This choice is due to LR's ability to effectively combine the predictions from various base models. Furthermore, a significant amount of research has successfully demonstrated the use of LR as a meta-model in ensemble models, including studies by Simmonds and Higgins [12], Mohapatra et al. [4], Zaini and Awang [13], and others.

4.3 Determine the Optimal Model Combination

4.3.1 GA Method

Genetic Algorithm (GA)[21][9] is an optimization technique that simulates the natural evolutionary process. It represents solutions to a problem as chromosomes and uses genetic operations such as mutation, selection, and crossover to improve and evolve the population, aiming to find the optimal solution. Since GA does not require prior knowledge of the search space and can run concurrently across multiple threads, it has been widely applied in various real-world problems.

In this study, the process of selecting a combination of base classifiers using an enhanced GA is depicted in Fig.3. Each potential base classifier is represented by a binary digit (0 for exclusion and 1 for inclusion), with the selected combination of base classifiers encoded as the individual's chromosome. Initially, the population is generated randomly with a set number of individuals. For each evolutionary iteration, the default meta-classifier is Logistic Regression (LR), and the F1-Score of the stacked ensemble model is calculated for each individual corresponding to the base classifier

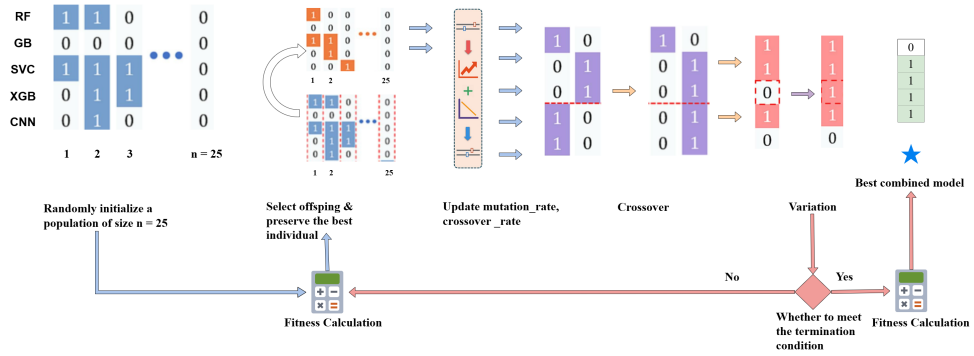

Fig. 3 The arrhythmia classification process using GA Stacking. This figure illustrates the process of selecting the best ensemble model combination using GA.

combination on the train set. The fitness score 5.2.5 of each individual is evaluated based on their F1-score, providing an objective performance measure. The genetic algorithm uses tournament selection to retain individuals with the highest fitness scores, ensuring that the best base classifier combination survives in each generation. The 'crossover_rate' and 'mutation_rate' are carefully tuned to maintain a balance between exploration and exploitation. The crossover operation merges the genetic information of two individuals, creating offspring with inherited strengths from both parents. In contrast, mutation introduces small random changes to an individual's chromosome, which ensures diversity and helps avoid the algorithm from becoming trapped in local optima. These genetic operations lead to the creation of new individuals, which replace older ones in the population. The process continues with each new generation evolving and potentially outperforming the previous one. This iterative process is repeated until a stopping condition is satisfied, such as reaching the maximum number of generations or achieving an optimal fitness score. Finally, after evaluating the fitness of all individuals in the final population, the individual with the highest fitness score is selected, ensuring the optimal base classifier combination for the ensemble model.

4.3.2 Grid Search Method

We observed that the process of searching for a model combination is similar to examining all the points in the data space, so they proposed using Grid Search[22] to perform this task. Model selection with Grid Search is a method aimed at discovering the best combination of models within an ensemble by systematically exploring a predefined set of possible configurations. The core principle involves an exhaustive and structured search through all potential combinations of base models. Each base model can either be included or excluded from the ensemble, depending on the outcomes of the search process.

First, we define the search space by listing all possible combinations of the base models. Each combination is represented as a binary vector, where each element is either 1 (indicating the inclusion of a base model) or 0 (indicating its exclusion).

For example, if there are 5 base models, the search space would consist of all binary combinations such as [0, 0, 0, 0, 0], [1, 0, 0, 0, 0], ... [1, 1, 1, 1, 1]. Next, we assess each model combination within the search space through the meta-model (LR) on the train set using an objective function, also referred to as the fitness function 5.2.5. This function measures the performance of the model generated by the chosen base models and the meta-model. We assess the combinations based on fitness to identify the one with the best performance (highest fitness). The search continues until all combinations in the search space have been tested, and the combination with the highest performance is selected. Ultimately, we arrive at the best model combination, which is the result of Grid Search, helping to optimize the performance of the ensemble model in solving the problem at hand.

While Grid Search ensures that the best solution is found by exhaustively exploring all options, it can be computationally expensive, especially when the number of classifiers increases, as the search space grows exponentially. Additionally, it is limited to the predefined search space and may miss optimal configurations outside of it. Nonetheless, Grid Search is a simple and effective method for model selection, ensuring that the most effective ensemble configuration is identified based on the chosen fitness criteria.

5 Experimental Design

To address the arrhythmias classification problem, we followed the workflow illustrated in Fig.1.

Our approach to arrhythmia classification involves using ensemble models that integrate base models: Random Forest, Gradient Boosting, SVC, XGBoost, and CNN. The dataset was split into a train set and a test set in a 7:3 ratio to ensure robust evaluation of the models. After, we present an ensemble approach for arrhythmia classification, integrating base models. Logistic Regression serves as a meta-model to evaluate different combinations of these base models on the train set, with fitness scores 5.2.5. The fitness of each individual is evaluated based on their F1-score, providing an objective performance measure. To identify the optimal model combination, we employed two methods: GA and Grid Search. Both methods were implemented simultaneously during the phase of searching for the best-performing combination to compare their effectiveness and determine the most suitable approach for this task. The general principle of both methods is to select the best model combination based on fitness score (the highest score is chosen). Each method produces a different ensemble model, then we evaluate these ensemble models on the test set using various evaluation metrics (Accuracy, Precision, F1-Score, Recall).

5.1 Hyperparameters of the Models

The parameters employed for the models in this paper are outlined in Table 1. While Grid Search explores all model combinations, we set specific limits for GA, including max-generation = 10 and population size = 25. GA will stop and conclude the best combination once the max-generation limit is reached, regardless of whether all possible scenarios have been explored.

Table 1 Model Hyperparameters

Model	Parameters
Random Forest	Number of estimators (`n_estimators`) = 100, random state (`random_state`) = 42.
Gradient Boosting	Default parameters used.
SVC	Probability estimation enabled (`probability=True`), random state (`random_state`) = 42.
XGBoost	Objective function = `'multi:softmax'`, number of classes (`num_class`) = 13.
CNN	Optimizer = `Adam`, learning rate (`lr`) = 0.001, weight decay (`weight_decay`) = 1×10^{-4}, loss function = `CrossEntropyLoss`, class weights (`class_weights`).
GA (Genetic Algorithm)	Population size (`population_size`) = 25, max generations (`max_generations`) = 10, mutation rate (`mutation_rate`) = 0.1, crossover rate (`crossover_rate`) = 0.7.

5.2 Evaluation Metrics

5.2.1 Accuracy

Accuracy is the percentage of correct predictions out of the total predictions. It is calculated using the following formula:

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total Instances}}$$

High accuracy indicates good classification performance, but it may not be sufficient for evaluation when the data is imbalanced.

5.2.2 Precision

Precision is the ratio of the number of correct predictions (True Positive) to the total number of predictions for that class (including both True Positive and False Positive). It reflects the model's ability to avoid making false predictions for a particular class:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

5.2.3 Recall

Recall, also known as sensitivity, is the ratio of the number of correct predictions to the total number of actual samples that belong to that class (including both True Positive and False Negative). Recall measures how well the model can detect actual instances of a specific class:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

5.2.4 F1-score

F1-score is the harmonic mean of Precision and Recall, balancing the two metrics. It is especially useful when evaluating models in situations where there is a trade-off between Precision and Recall:

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

5.2.5 Fitness Score

The fitness score is computed by selecting base classifiers based on the input 'individual' (a binary vector), where each element corresponds to whether a particular base classifier is included in the ensemble or not. If no classifiers are selected, the function returns a score of 0.0. This process is carried out within the 'fitness_function' method.

Next, we evaluate the selected classifier combinations using K-Fold cross-validation (with 3 splits). In each split, the train set is divided into train set and validation set, then the base classifiers (Random Forest, Gradient Boosting, SVC, XGB, and CNN) are trained on the train set. For each base classifier, the predicted probabilities on the validation set are calculated. These predictions are stored in a list for subsequent stacking.

Specifically, for classifiers like Random Forest and XGB, if there are missing classes in the training data (classes that do not appear in the K-Fold train set), we add dummy samples to ensure that all classes are represented during training. This prevents the classifiers from failing to learn about certain classes.

After training, the predictions from all base classifiers are stacked together into a feature matrix, and this matrix is passed to the 'meta_classifier' (meta-model) for training. The meta-model learns how to combine the predictions of the base classifiers to make the final prediction.

The final predictions from the meta-model (LR) are compared with the actual labels in the validation set, and the F1-score (average="macro") is calculated to assess the accuracy of the predictions. The F1-score reflects the model's classification performance, particularly in imbalanced datasets. Finally, the average F1-score across all K-Fold splits is returned as the fitness score for the current classifier combination.

5.3 Experimental results

To select the best model combination, the team trains different model combinations using the meta-model (Logistic Regression), applies cross-validation on train set, and calculates the fitness score 5.2.5. The specific process is described according to the model selection methods (GA/ GS) mentioned in Section 4.3.

The model selection method using GA yielded the optimal configuration, with the matrix being [0, 1, 1, 1, 1], meaning the selection of the following four base models: Gradient Boosting, SVC, XGBoost, and CNN. GA determined this combination with a highest fitness value of 0.9708, achieved in the 9th generation.

Using Grid Search, the optimal configuration was found to include all five models: Random Forest, Gradient Boosting, SVC, XGBoost, and CNN (the matrix being [1, 1, 1, 1, 1] with a maximum fitness value of 0.723).

The two methods produce differing results, attributed to the GA algorithm's randomness, which can lead to some cases being overlooked.

After obtaining the best model combination with the two methods, GA and Grid Search, we use four fundamental metrics—Accuracy, Precision, Recall, and F1-Score 5.2—to evaluate the performance of the models on testset. The table 2 below illustrates the performance of the models:

Table 2 Experimental Results on Testset

	Base Model					Ensemble Model	
	RF	**GB**	**SVC**	**XGB**	**CNN**	**Ensemble GA**	**Ensemble GS**
Accuracy	0.71	0.72	0.60	0.72	0.61	0.70	0.71
Precision	0.40	0.49	0.22	0.50	0.33	0.45	0.48
F1-Score	0.34	0.42	0.16	0.43	0.29	0.37	0.40
Recall	0.36	0.44	0.15	0.45	0.28	0.39	0.43

Table 2 compares the performance of base and ensemble models on the test set. Among the base models, Gradient Boosting (GB) and XGBoost (XGB) lead with the highest accuracy (0.72), with XGB excelling in Precision (0.50) and F1-Score (0.43), indicating strong performance in handling imbalanced data.

The ensemble models (Ensemble GA and Ensemble GS) outperform most base models across multiple metrics, with Ensemble GS achieving the highest accuracy (0.71), comparable to GB (0.72) and XGB (0.72), and surpassing RF, SVC, and CNN in accuracy, precision, recall, and F1-Score. However, GB and XGB still outperform in certain metrics, suggesting potential for further optimization.

All ensemble models perform well, with accuracy above 0.70. Notably, Ensemble GS outperforms Ensemble GA by 0.01%-0.04% in all metrics, showing better accuracy in predicting and identifying positive samples.

In conclusion, while XGB is the top base model, ensemble models, especially Ensemble GS, offer significant improvements in balanced metrics, showcasing the power of ensemble learning in providing more reliable predictions.

6 Conclusion and Future development

We successfully applied both GA and Grid Search methods to build ensemble models for arrhythmia classification on the Cardiac Arrhythmia Database, aiming to accurately classify various types of arrhythmia. The dataset was divided into a train set and a test set in a 7:3 ratio. The ensemble model integrated five base models (Random Forest, Gradient Boosting, SVC, XGBoost, and CNN). Logistic Regression served as a meta-model to evaluate different model combinations on the train set, producing Fitness scores. Then, GA and Grid Search were utilized to determine the optimal model combinations yielding the best performance. Subsequently, each method produces two

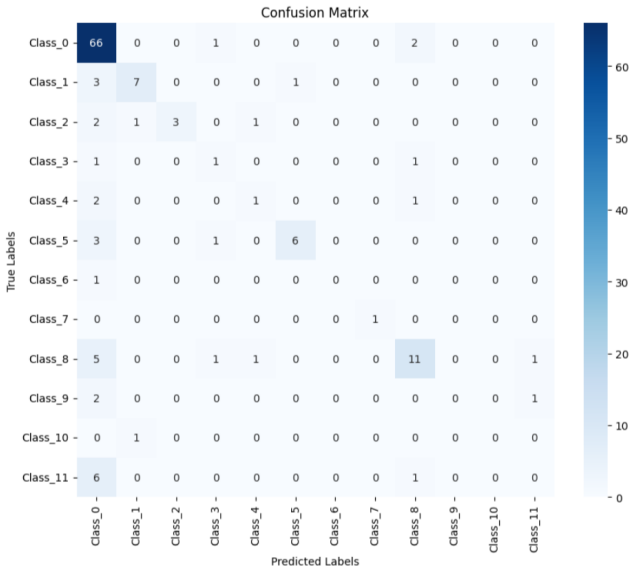

Fig. 4 Confusion matrix of ensemble model (GA). This figure illustrates the confusion matrix of the ensemble model (GA) used in the classification task.

different ensemble models, then we evaluate these ensemble models on the test set using various performance metrics.

Experimental results demonstrated that all ensemble models delivered strong performance, achieving accuracy above 0.70. Additionally, Grid Search outperformed GA in this case due to the small size of the search space associated with the Cardiac Arrhythmia dataset, which contains only 452 records. Grid Search exhaustively evaluated all combinations of base models, yielding better results and shorter training times compared to the random exploration of GA. However, GA remains advantageous for larger search spaces and more complex datasets.

Additionally, GB and XGB stand out in certain metrics, with an accuracy of 0.72, surpassing even the ensemble models. This highlights the potential for further optimization of the ensemble models to fully leverage the strengths of these high-performing base models.

This study underscores the potential of Stacked Ensemble models, optimized with both GA and Grid Search, in physiological signal classification, paving the way for the development of advanced automated medical diagnosis systems.

In the future, we plan to evaluate and test the proposed model on various datasets. A major challenge is the limited amount of data, and collaborating with hospitals and medical organizations to collect high-quality data will help improve the model. Additionally, we aim to enhance the model by using GA-Stacking to improve feature extraction, reduce classification errors, and boost overall performance.

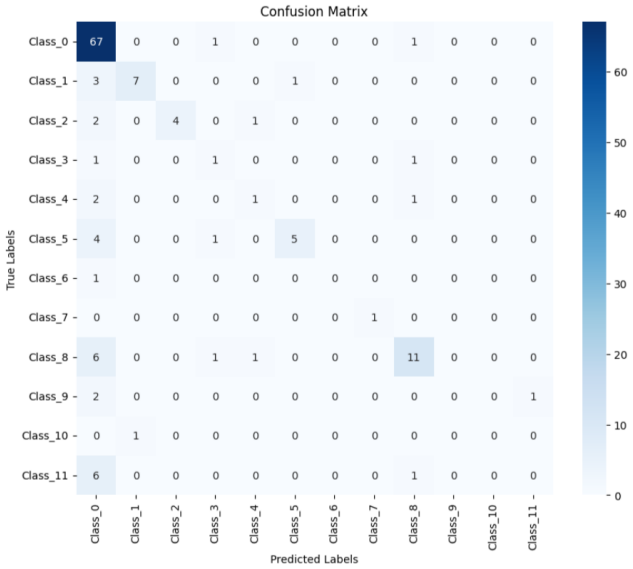

Fig. 5 Confusion matrix of ensemble model (GS). This figure illustrates the confusion matrix of the ensemble model (GS) used in the classification task.

References

[1] Cardiac Arrhythmia Database. Cardiac Arrhythmia Database. Accessed: 2025-01-03 (2024)

[2] Wu, M., Lu, Y., Yang, W., Wong, S.Y.: A study on arrhythmia via ecg signal classification using the convolutional neural network. Frontiers in Computational Neuroscience **14**, 564015 (2021) https://doi.org/10.3389/fncom.2020.564015

[3] Nandita, N.Y., Mandala, S.: Lstm-based arrhythmia classification in electrocardiogram signals. In: 2023 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob), pp. 78–83 (2023). https://doi.org/10.1109/APWiMob59963.2023.10365617

[4] Mohapatra, S., Maneesha, S., Mohanty, S., Patra, P.K., Bhoi, S.K., Sahoo, K.S., Gandomi, A.H.: A stacking classifiers model for detecting heart irregularities and predicting Cardiovascular Disease. Published in Healthcare Analytics, vol 3, article 100133. https://doi.org/10.1016/j.health.2022.100133 (2023)

[5] Zhao, Z.: Transforming ecg diagnosis: An in-depth review of transformer-based deep learning models in cardiovascular disease detection. ArXiv **abs/2306.01249** (2023)

[6] Dalal, F., Ingale, V.V.: Arrhythmia identification and classification using ensemble learning and convolutional neural network. In: 2021 2nd Global Conference for

Advancement in Technology (GCAT), pp. 1–8 (2021). https://doi.org/10.1109/GCAT52182.2021.9587596

[7] Yakut, , Bolat, E.D.: A high-performance arrhythmic heartbeat classification using ensemble learning method and psd-based feature extraction approach. Biocybernetics and Biomedical Engineering **42**(2), 667–680 (2022) https://doi.org/10.1016/j.bbe.2022.05.004

[8] Tan, Y., Chen, H., Zhang, J., Tang, R., Liu, P.: Early Risk Prediction of Diabetes Based on GA-Stacking. Published in Applied Sciences, vol. 12, no. 2, article number 632. (2022). https://doi.org/10.3390/app12020632

[9] 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. Published in Biomedical Signal Processing and Control, vol. 87, article number 105506. (2024). https://doi.org/10.1016/j.bspc.2023.105506

[10] Iyer, T.J., Kishan, B., Nersisson, R.: Prediction and Classification of Cardiac Arrhythmia Using a Machine Learning Approach. Paper presented at the Advances in Automation, Signal Processing, Instrumentation, and Control conference (i-CASIC 2020), Lecture Notes in Electrical Engineering, vol 700, Springer, Singapore. https://doi.org/10.1007/978-981-15-8221-9_53 (2021)

[11] Chen, T.-M., Huang, C.-H., Shih, E.S.C., Hu, Y.-F., Hwang, M.-J.: Detection and Classification of Cardiac Arrhythmias by a Challenge-Best Deep Learning Neural Network Model. Published in iScience, vol 23, no 3, pages 100886. Available at: https://doi.org/10.1016/j.isci.2020.100886 (2020)

[12] Simmonds, M.C., Higgins, J.P.: A general framework for the use of logistic regression models in meta-analysis. Statistical Methods in Medical Research **25**(6), 2858–2877 (2016) https://doi.org/10.1177/0962280214534409

[13] Zaini, N.A.M., Awang, M.K.: Performance comparison between meta-classifier algorithms for heart disease classification. International Journal of Advanced Computer Science and Applications **13**(10) (2022) https://doi.org/10.14569/IJACSA.2022.0131039

[14] feature-engine.trainindata.com: MRMR. https://feature-engine.trainindata.com/en/latest/api_doc/selection/MRMR.html#feature_engine.selection.MRMR. Accessed: 2025-01-03

[15] Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001) https://doi.org/10.1023/A:1010933404324

[16] Friedman, J.H.: Greedy function approximation: A gradient boosting machine. Annals of Statistics **29**, 1189–1232 (2001)

[17] geeksforgeeks.org: Support Vector Machine (SVM) Algorithm. https://www.geeksforgeeks.org/support-vector-machine-algorithm/. Accessed: 2025-01-03

[18] xgboost.readthedocs.io: Introduction to Boosted Trees. https://xgboost.readthedocs.io/en/stable/tutorials/model.html. Accessed: 2025-01-03

[19] Saxena, A.: An introduction to convolutional neural networks. International Journal for Research in Applied Science and Engineering Technology (2022)

[20] Albawi, S., Mohammed, T.A., Al-Zawi, S.: Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET), pp. 1–6 (2017). https://doi.org/10.1109/ICEngTechnol.2017.8308186

[21] Ledezma, A., Aler, R., Sanchis, A., Borrajo, D.: Ga-stacking: Evolutionary stacked generalization. Intell. Data Anal. **14**(1), 89–119 (2010)

[22] scikit-learn.org: GridSearchCV. https://scikit-learn.org/1.5/modules/generated/sklearn.model_selection.GridSearchCV.html. Accessed: 2025-01-03