BAREKENG: Journal of Mathematics and Its Applications
June 2024 Volume 18 Issue 2 Page 1053–1066
P-ISSN: 1978-7227 E-ISSN: 2615-3017
https://doi.org/10.30598/barekengvol18iss2pp1053-1066

OPTIMIZING HEART ATTACK DIAGNOSIS USING RANDOM FOREST WITH BAT ALGORITHM AND GREEDY CROSSOVER TECHNIQUE

Safrizal Ardana Ardiyansa[1], Natasha Clarissa Maharani[2], Syaiful Anam[3*], Eric Julianto[4]

[1,2,3]*Department of Mathematics, Faculty of Mathematics and Natural Science, Universitas Brawijaya*
[1,2,3]*Jalan Veteran, Malang, 65145, Indonesia*

[4]*Braincore Indonesia*
Jl. Letjen S. Parman No.28, Jakarta, 11470, Indonesia

*Corresponding author's e-mail: *syaiful@ub.ac.id*

Article History:

Received: 30th December 2023
Revised: 19th January 2024
Accepted: 22nd March 2024
Published: 1st June 2024

Keywords:

Heart Attack;
Bat Algorithm;
Random Forest;
Greedy Crossover.

ABSTRACT

Cardiovascular disease stands as one of the primary contributors to global mortality, with the World Health Organization (WHO) reporting approximately 17.9 million deaths annually. Swift and accurate diagnosis of heart attacks is crucial to ensure timely and specialized intervention for patients afflicted by this ailment. A machine learning algorithm that can be employed for addressing such issues is the Random Forest algorithm. However, the efficacy of the model is significantly influenced by the features selected during the training phase. To mitigate this, the Binary Bat Algorithm (BBA) with greedy crossover has been utilized to enhance feature selection within the model. This approach is particularly adept at preventing convergence issues often associated with local minima. The optimal parameters for BBA with greedy crossover are determined to be $\alpha = 0.9$, $\gamma = 0.2$, $n = 25$, *and* $t_{max} = 40$. *With these parameters, the proposed algorithm identifies the most relevant features, including age, gender, cp, chol, thalach, oldpeak, slope, and ca, achieving an accuracy of 94.19% on the training data and 91.8% on the test data. Furthermore, the precision and recall values for both classes range from 0.87 to 0.96, contributing to an approximate* f_1*-score of 0.92. The proposed method has increased its* f_1*-score by 0.05 if compared with the regular Random Forest model. These results underscore the effectiveness of the proposed algorithm in providing accurate and reliable predictions for heart disease diagnosis. As such, this model makes diagnosing heart attack more convenient and effective because it does not require too much medical features or patient data. Hopefully, the results of this research help medical practitioners make better and timely decisions in the diagnosis and treatment of heart attacks, as well as assist in planning more effective public health programs for heart attack prevention.*

How to cite this article:

S. A. Ardiyansa, N. C. Maharani, S. Anam and E. Julianto., "OPTIMIZING HEART ATTACK DIAGNOSIS USING RANDOM FOREST WITH BAT ALGORITHM AND GREEDY CROSSOVER TECHNIQUE," *BAREKENG: J. Math. & App.*, vol. 18, iss. 2, pp. 1053-1066, June, 2024.

Journal homepage: *https://ojs3.unpatti.ac.id/index.php/barekeng/*
Journal e-mail: *barekeng.math@yahoo.com; barekeng.journal@mail.unpatti.ac.id*

Research Article · Open Access

1. INTRODUCTION

The heart is a remarkable organ that plays a vital role in sustaining human life. Its significance lies in its complex and orchestrated functions, which are indispensable for the body's overall well-being. A poorly functioning heart can seriously affect other organs, such as the brain and kidneys [1]. Heart attacks, also known as myocardial infarctions, pose a significant threat to global health and are a leading cause of death worldwide. The severity of a heart attack lies in its potential to cause irreversible damage to the heart muscle, often with life-threatening consequences. The World Health Organization's findings indicate that approximately 17.9 million individuals succumb to heart attacks annually [2]. This data underscores the importance of diagnosing heart attacks to prevent severe consequences and save lives. The disease diagnosis process involves utilizing patient data as a source of medical information, requiring credible and relevant features [3]. Patient data plays a crucial role in the decision-making process for medical experts, making it a valuable resource for diagnosing illnesses [4]. Diagnosing processes based on patient data can be challenging and time-consuming. Machine learning algorithms can automate the diagnosis process and assist medical professionals. Support Vector Machines (SVM) [5], K-Nearest Neighbors (KNN) [6], Naive Bayes [7], and Random Forest [8] are machine learning algorithms that can help to predict heart diagnosis.

Random Forest has proven successful in diverse applications, including surface water detection [9], human gait recognition [10], supplier selection criteria classification [11], glaucoma detection [12], and many others. Past studies have shown that Random Forest exhibits superior generalization performance, faster learning speed, and a more straightforward implementation process without requiring extensive and time-consuming parameter tuning for diagnosing purposes [13]. Random Forest also offers effectively handles high-dimensional medical data [14], supports multi-class tasks [15], and suits parallel processing [16]. It also exhibits robustness in learning, even when dealing with significant data errors, while other algorithms are greatly affected by inaccuracies in the data [17]. Random Forest also demonstrates comparable predictive accuracy and computational efficiency [18], making it an excellent choice for real-world prediction tasks [19].

The performance of the Random Forest model heavily relies on the features selected as training data [20]. However, patient data typically presents a challenge with irrelevant or redundant features. Irrelevant features in the dataset can lead to various issues, requiring a feature selection process [21]. Feature selection is crucial to prevent the model from overfitting, reduce computational time, improve model accuracy, and it is very beneficial for high-dimensional data [22]. Therefore, features of the dataset must be carefully selected before model training to achieve better results [23].

Selecting the right features is a complex problem; it involves navigating through a multitude of factors that can influence the performance of a machine learning model [24]. Trying all feature combinations is time-consuming [25]. The bat algorithm has proven to outperform other optimization methods in feature selection. This algorithm is inspired by bat echolocation behavior [26]. The bat algorithm employs intelligent and efficient search strategies in exploring the search space [27]. The bat algorithm's dynamic frequency balances exploration and exploitation [28]. However, bats attract others quickly in high-dimensional data, leading to a sharp decrease in population diversity [29]. This problem causes premature convergence due to reduced diversity with increasing iterations [30]. This problem also led to various developments and modifications of the bat algorithm [31].

Modification techniques are needed for the bat algorithm to address rapid convergence. The greedy crossover technique can overcome this problem and prevent the algorithm from converging to local optima [32]. The concept of genetic recombination in biological reproduction inspired this technique. The greedy crossover will generate new offspring from the two best parents [33]. This technique offers more significant potential to explore and find more optimal solutions. The crossover process, relying solely on the two best parents, enhances computational time efficiency and is expected to yield more optimal offspring. Fast computational time is necessary to allow iterative training processes for machine learning models in the presence of new medical data additions. Fast computational time is crucial for swift patient diagnoses. It catalyzes for a paradigm shift in healthcare delivery, where timely and precise diagnostics become the cornerstone of effective medical intervention, ultimately enhancing the quality of care and saving lives. Given the significant increase in data over time, the model is trained iteratively. Therefore, to implement the proposed algorithm, finding the optimal parameters is essential for achieving the best features and minimizing computation time. In searching for the best parameters, analyzing the impact of modifications in bat population, maximum iterations, and other parameters on the model's performance and computation time is

essential. This background encourages research into incorporating greedy crossover techniques into the bat algorithm to improve its performance in finding the best parameters with fast computing time and maximum accuracy. As such, this study proposes a random forest with a bat algorithm and greedy crossover technique to optimize heart attack diagnosis.

2. RESEARCH METHODS

The research process involves several steps, as depicted in **Figure 1**. Initial steps include importing the dataset into Google Colaboratory, pre-processing it, and splitting it into training and testing data. Move to the next stage, parameter initialization for the bat algorithm is performed. As the results will vary in each experiment, conducting repeated experiments is essential **[34]**. Therefore, the program is executed 25 times to observe the consistency and variation in computing time during training and accuracy during evaluation. The bat algorithm is executed in each running process to select features and obtain the best random forest model. The performance of the random forest model is then assessed using the test data, and results for accuracy and computation time are recorded. The average and distribution of accuracy and computation time are calculated upon completing the iterations. This data will be used to identify optimal parameters and models for diagnosing heart attacks.

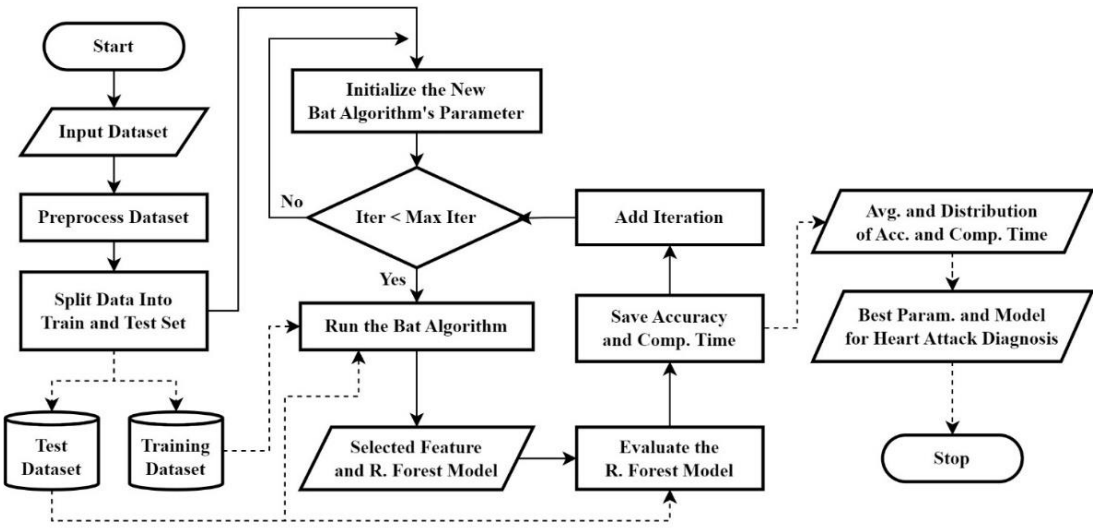


Figure 1. The Research Steps

2.1 Input and Pre-processing Data

The data for this research study was acquired from the Kaggle platform and consists of patient medical data. It comprises 1025 rows and 13 features that provide insights into various aspects of the patient's health. These features include age, gender, type of chest pain experienced (cp), resting blood pressure (trestbps), cholesterol level (chol), fasting blood sugar (fbs), resting electrocardiographic results (restecg), maximum heart rate achieved (thalach), presence of exercise-induced angina (exang), ST depression induced by exercise relative to rest (oldpeak), slope of the peak exercise ST segment (slope), number of major vessels (ca), and a parameter called 'thal' as shown in **Table 1**. The dataset also includes a target variable indicating heart disease. It has two classes: zero, indicating that patients are not at risk of a heart attack, and one, indicating a risk of experiencing a heart attack.

Table 1. Sample of the Dataset

No	Age	Sex	Cp	Trestbps	Chol	Fbs	Restecg	Thalach	Exang	Oldpeak	Slope	Ca	Thal	Target
1	52	1	0	125	121	0	1	168	0	1	2	2	3	0
2	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1025	54	1	0	120	188	0	1	113	0	1.4	1	1	3	1

***Data source**: Kaggle*

In this research, the preprocessing stage aims to ensure the dataset's quality before further analysis. After checking for missing values, it was found that the dataset used is clean, with no missing values requiring imputation. However, upon closer inspection, 723 duplicate entries were identified. Removing these duplicates was necessary to maintain the analysis's integrity and avoid potential bias. The next step involves splitting the data into training and test sets. The training data comprises 80%, while the test data is 20%. This proportion is chosen to balance training the model with a dataset that includes diverse patterns and characteristics while providing sufficient testing to measure the model's performance objectively.

2.2 Initialize Bat Algorithm's Parameter

The next step involves initializing parameters. To ensure unbiased identification of optimal parameters, the program goes through a looping process that runs multiple times to compare the algorithm's performance. Parameters like bat population, loudness decrease rate (α), and pulse rate increase speed (γ) must be initialized with various values. This iterative approach ensures a thorough exploration of the parameter space, leading to a more robust evaluation of the algorithm's effectiveness. The best parameters α and γ actually depend on its objective function, but most objective functions with parameters $\alpha = 0.9$ and $\gamma = 0.1$ have best performance **[35]**. It encourages this research to use alpha value (α) between range $\alpha \in [0.7, 0.9]$, because low alpha value would hinder the bat algorithm's effectiveness in exploring the search space. At the same time, the gamma value (γ) under consideration spans from 0.1 to 0.3, aiming to strike a balance between exploration and exploitation in the search for optimal solutions.

2.3 Binary Bat Algorithm (BBA)

The algorithm employed for feature selection in this research is the Binary Bat Algorithm (BBA), as introduced by Mirjalili et al. **[36]**. BBA is chosen for its ability to operate in a binary space $\{b_1, b_2, b_3, \dots, b_m\}$, where this binary space can effectively represent the feature set in the dataset. Given the research goal of selecting or not selecting a particular feature, each entry in this space holds a value of $b_j = 0$ or $b_j = 1$ where $j \in \{1, 2, 3, \dots, m\}$, and m is the number of features. Here, the value $b_j = 0$ denotes an unselected feature, while $b_j = 1$ signifies a selected feature. In this context, the bat's position is denoted by a binary vector, and the constraints on the new bat position are made binary using **Equation (1)**, while the update of the i-th bat's position is determined by **Equation (2)**, where σ is a random number in range $\sigma \in [0, 1]$.

$$S(x_i^j) = \left[1 + \exp(-x_i^j)\right]^{-1} \quad (1)$$

$$x_i^j = \begin{cases} 1, & \text{if } S(x_i^j) > \sigma \\ 0, & \text{other} \end{cases} \quad (2)$$

In this research, each bat will perform exploration and exploitation in each iteration. The exploration process aims to find the best solution in the search space and avoid getting stuck in a local optimum. The update of frequency (f_i), movement speed ($\boldsymbol{v}_i^t$), and the position ($\boldsymbol{x}_i^t$) of the i-th bat during iteration t in the exploration process is represented by **Equation (3)**,

$$f_i = f_{min} + (f_{max} - f_{min})\beta, \quad \boldsymbol{v}_i^t = \boldsymbol{v}_i^{t-1} + (\boldsymbol{x}_i^t - \boldsymbol{x}_{best}^t)f_i, \quad \boldsymbol{x}_i^t = \boldsymbol{x}_i^{t-1} + \boldsymbol{v}_i^t \quad (3)$$

where $\beta \in [0,1]$ represents a random number, and $\boldsymbol{x}_{best}^{t}$ denotes the position of the bat with the best accuracy among all bats at iteration t. This research employs $f_{min} = 0$ and $f_{max} = 1$ since the solution search domain is in binary space.

During the exploitation process, bats move towards the neighborhood of their previous positions and formulated mathematically by **Equation (4)**, where $\varepsilon \in [-1,1]$ is a random number, and A^t is the average loudness of each bat at iteration t.

$$x_{new}^{j} = x_{old}^{j} + \varepsilon A^t \tag{4}$$

$$A_i^{t+1} = \alpha A_i^t, \quad r_i^{t+1} = r_{max_i}[1 - \exp(-\gamma t)] \tag{5}$$

The initial loudness of each bat in this research is randomly set within the range $A_i^0 \in [1,2]$, while the maximum pulse rate (r_{max}) for i-th bat also initialized randomly within the range $r_{max_i} \in [0,1]$. The loudness (A_i) and pulse rate (r_i) for each i-th bat will be updated as the iterations progress using **Equation (5)**, where loudness' depreciation factor (α) and pulse rates' increasing speed (γ) are constants with $0 < \alpha < 1$ and $\gamma > 0$. As the iteration (t) approaches infinity, the conditions $A_i^t \to 0$ and $r_i^t \to r_{max_i}$, as indicated in **Figure 2**. In **Figure 2**, the blue, yellow and green lines which represented the A^t value decreases from 2 to 0 while the red, purple and magenta lines which represented the r^t value increases from 0 to 1. It can be inferred that as the α value increases, the time taken for the value to converge to 0 extends, and conversely, a smaller α leads to faster convergence. Similarly, a larger γ results in quicker convergence to 1, while a smaller γ leads to a slower convergence. To enhance the exploration process of bats, the α parameter should be minimized to ensure a gradual convergence of the A^t value to zero. Conversely, for the r^t value to converge slowly to one, the γ parameter should be maximized, obtaining the highest possible value.

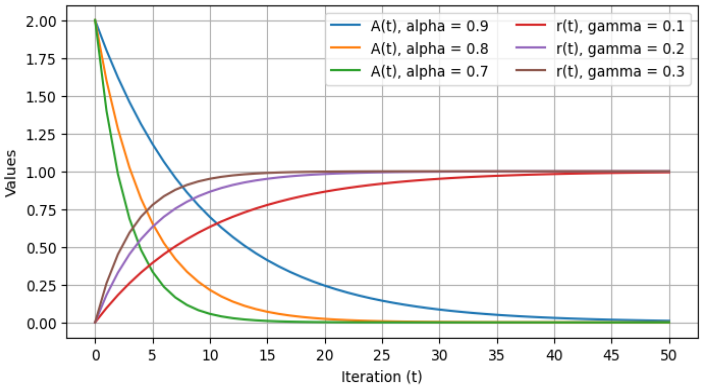


Figure 2. Graph of loudness and pulse rate

2.4 Greedy Crossover Technique in BBA

Crossover operators play a significant role in maintaining a balance between exploitation and exploration for feature selection **[37]**. This technique involves combining two sets of features as parents to create two new sets or offspring. The offspring inherit elements from both parents **[38]**. Greedy crossover is a specific method that focuses on selecting the two best parents at each iteration, aiming for the global optimum. In this research, the double-point crossover method is employed. In the double-point crossover, two random locations along the chromosome are chosen, and the genetic material between these points is exchanged between the parents **[39]**. For example, as depicted in **Figure 3**, the first and second parent are separated into three sections by two lines, and the green segments will undergo the crossover. To produce the first offspring, the green segments of the first parent are replaced by the green segment of the second parent, and the second offspring is obtained with the same process as the first offspring. This process introduces diversity by combining segments from both parents. The variability within this diversity will be leveraged to enhance the algorithm's optimization process.

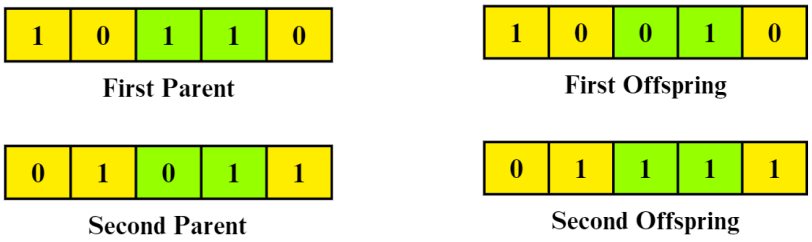


Figure 3. Double Point Crossover

Upon applying greedy crossover in the BBA, the position of the bat that produces the best fitness value (**Gbest**) is stored in a data frame called the '**Gbest dataframe**'. All rows in this data frame are then sorted in descending order based on the test accuracy generated by the bat's position. In this research, **Gbest dataframe** will only store unique bat position data to avoid crossover between identical parents. Thus, the deletion of duplicate **Gbest** data is necessary. If **Gbest dataframe** already has more than two unique bats, the greedy crossover process can be carried out. This process involves the two best and distinct bats. Next, the '**and**' operator and '**bitwise**' operator are performed to generate '**Crucial features**' and '**Semicrucial features**' between the two best bats to maintain truly important features and not reconsider truly unimportant features. Subsequently, greedy crossover is applied to **Crucial features** and **Semicrucial features** to generate new offspring. The new offspring will be evaluated, and the position of the worst bat will be replaced by the offspring and stored in the **Gbest dataframe**.

Algorithm 1. Greedy Crossover Technique in BBA

1: Add the best bat on the ***Gbest_dataframe***
2: Sort ***Gbest_dataframe*** based on test acc. (descending)
3: Remove duplicates row on the ***Gbest_dataframe***
4: *if* |***Gbest_dataframe***| > 2, *then*
5: Set ***First_Parent*** (Fp) as the first element of the ***Gbest***
6: Set ***Second_Parent*** (Sp) as the second element of the ***Gbest***
7: Compute ***Crucial_Feature*** with and operator between (Fp) and (Sp)
8: Compute ***Semicrucial_Feature*** with bitwise operator between (Fp) and (Sp)
9: Apply crossover on ***Crucial_Feature*** and ***Semicrucial_Feature*** to generate the ***Offspring***
10: Replace the worst bat with the ***Offspring***
11: Train and evaluate the model with the ***Offspring*** to get its train and test acc.
12: Add the ***Offspring*** on the ***Gbest_dataframe***
13: *end if*

2.5 BBA with Greedy Crossover

his research introduces an innovative algorithm named the Binary Bat Algorithm (BBA) with greedy crossover. The main goal of developing this algorithm is to improve the efficiency and diversity of the feature selection solution search process. The algorithm iteratively applies the greedy crossover technique to achieve this goal. The pseudocode for the proposed algorithm is presented below:

Algorithm 2. BBA with Greedy Crossover

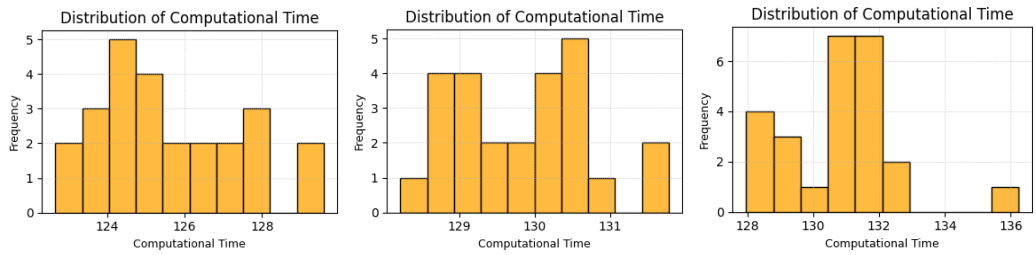
1: Initialize *Population_Size (n)*, *Alpha* ($\boldsymbol{\alpha}$), *Gamma* ($\boldsymbol{\gamma}$), and *Max_Iteration* ($\boldsymbol{t_{max}}$)
2: Set *Min_Freq* = 0 and *Max_Freq* = 1
3: *for i in range* (*n*) *do*
4: $\boldsymbol{A_i^0}$ = *Random*[1, 2]
5: $\boldsymbol{r_{max_i}}$ = *Random*[0, 1]
6: *for j in range*(*m*) *do*
7: $\boldsymbol{x_i^j}$ = *Random*$\{\mathbf{0}, \mathbf{1}\}$
8: *end for*
9: *if* $\boldsymbol{x_i = 0}$, *then*

Modify some entry on $\boldsymbol{x_i^j}$ from zero to one to ensure $\boldsymbol{x_i^j \neq 0}$ for all *j*
end if
end for
Set *Global_Fitness* = 0
Set $\boldsymbol{fit_i} = 0$ for all *i*
Create an empty ***Gbest_dataframe*** with feature, train acc., and test acc. as columns
for t in range($\boldsymbol{t_{max}}$) *do*
  for i in range(*n*) *do*
    Train model with feature on $\boldsymbol{x_i^j}$ and store its test acc. in *new_fit*
    if Random[0, 1] < $\boldsymbol{A_i}$ and *new_fit* > $\boldsymbol{fit_i}$, *then*
      $\boldsymbol{fit_i} = new\_fit$
      $\boldsymbol{A_i^t = \alpha A_i^{t-1}}$
      $\boldsymbol{r_i^t = r_{max_i}[1 - \exp(-\gamma t)]}$
    end if
  end for
  max_fit = $\boldsymbol{\max(fit)}$
  if max_fit > *Global_Fitness, then*
    Global_Fitness = *max_fit*
    Determine best bat position from *Global_Fitness*
  end if
  Perform **Greedy Crossover Technique in BBA**
  for i in range(*n*) *do*
    if Random[0, 1] > $\boldsymbol{r_i^t}$, *then*
      for j in range(*m*) *do*
        $\boldsymbol{x_{new}^j = x_{old}^j + \varepsilon A^t}$
        if $\boldsymbol{\sigma < S\left(x_i^j\right)}$, *then*
          $\boldsymbol{x_i^j = 1}$
        else
          $\boldsymbol{x_i^j = 0}$
        end if
      end for
    end if
    if Random[0, 1] < $\boldsymbol{A_i^t}$ and $\boldsymbol{fit_i < Global\_Fitness}$, *then*
      for j in range(*m*) *do*
        $\boldsymbol{f_i = f_{min} + (f_{max} - f_{min})\beta}$
        $\boldsymbol{v_i^j = v_i^{j-1} + \left(x_i^j - x_{best}^j\right)f_i}$
        $\boldsymbol{x_i^j = x_i^{j-1} + v_i^j}$
        if $\boldsymbol{\sigma < S\left(x_i^j\right)}$, *then*
          $\boldsymbol{x_i^j = 1}$
        else
          $\boldsymbol{x_i^j = 0}$
        end if
      end for
    end if
  end for
end for

RESULTS AND DISCUSSION

3.1 Influence of Parameters on Computational Time

The BBA with greedy crossover was evaluated using a Random Forest model. This research employs a Random Forest model with a maximum tree depth of five to prevent overfitting. Other parameters use the default settings provided by the sci-kit-learn library, including 100 trees, a minimum sample leaf size of one, an auto selection for maximum features, and the Gini impurity criterion. The BBA with greedy crossover, was evaluated with variations in the $\boldsymbol{\alpha}$ (loudness' depreciation factor) and $\boldsymbol{\gamma}$ (pulse rates' increasing speed) parameters. The results suggest that the distribution of computation time is not notably affected by the values assigned to $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$. Upon experimentation, it is observed that the computation time exhibits variations, as depicted in **Figure 4**. The distribution of computation time does not consistently adhere to a normal distribution. Notably, there are instances where computation time behaves as an outlier, as illustrated in **Figure 4** (c). Additionally, skewed distributions with two peaks are observed, as exemplified in **Figure 4** (b).



(a) **(b)** **(c)**

Figure 4. Distribution of computational time with different values α and γ specifically; (a) $\alpha = 0.8$ and $\gamma = 0.1$, (b) $\alpha = 0.8$ and $\gamma = 0.3$, (c) $\alpha = 0.7$ and $\gamma = 0.1$

In the proposed algorithm, the computation time is notably affected by two key parameters: the maximum number of iterations (denoted as t_{max}) and the bat population size (denoted as n). The larger the values assigned to these parameters, the longer the algorithm takes to compute its results. This relationship is visually represented in **Figure 5**. The correlation between computation time and the maximum number of iterations, with a constant bat population (n_{20}), follows a regression line expressed by **Equation (4)**.

$$C(t_{max}, 20) = a_1 t_{max} + b_1 = 4.477 t_{max} + 5.630 \quad (4)$$

Similarly, the relationship between computation time and the bat population, with a constant maximum number of iterations ($t_{max} = 20$) can be described by a regression line given in **Equation (5)**.

$$C(20, n) = a_2 n + b_2 = 4.008 n + 9.080 \quad (5)$$

Analyzing these equations, it is evident that the maximum number of iterations (t_{max}) has a more significant impact on computation time compared to the bat population size (n). This conclusion is drawn from the coefficients a_1 and a_2, where $|a_1| > |a_2|$, indicating a greater influence of the maximum number of iterations on the computational time.

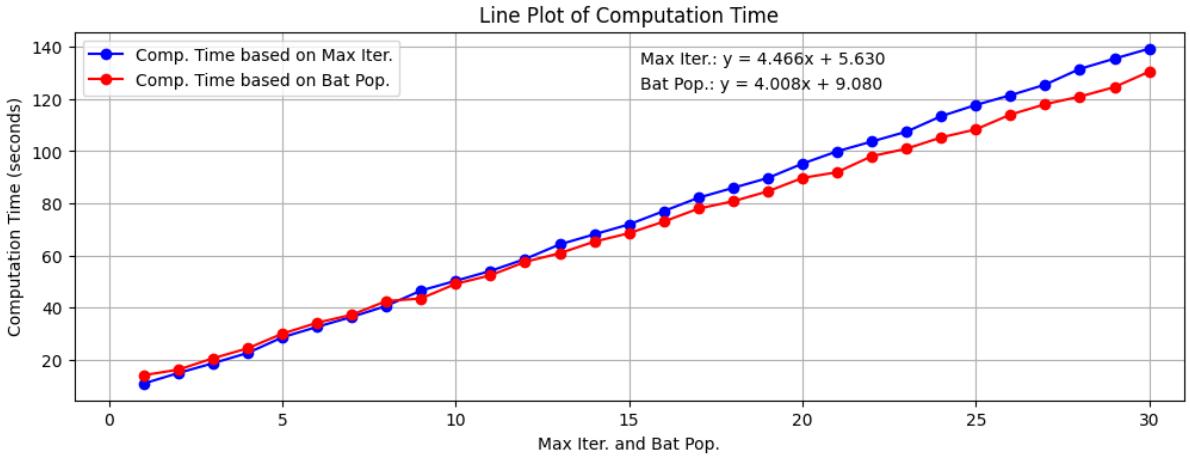


Figure 5. Plot of computational time based on maximum iteration and bat population

Table 2. Performance of the Proposed Algorithm with Various Parameter α and γ

Max Iteration	Alpha	Gamma	Bat Pop.	Avg. Accuracy		Std. Accuracy		Average Comp. Time	Std. Comp. Time
				Train	Test	Train	Test		
25	0.9	0.1	25	0.9095	0.9205	0.0141	**0.0072**	139.1388s	2.4324s
25	0.9	0.2	25	0.9085	**0.9213**	0.0159	0.0095	139.9221s	3.7042s
25	0.9	0.3	25	0.9059	0.9205	**0.0125**	0.0086	**124.9108s**	1.3360s
25	0.8	0.1	25	0.9076	0.9206	0.0183	0.0091	125.5980s	1.8942s
25	0.8	0.2	25	0.9046	0.9205	0.0253	0.0098	130.7010s	3.2170s
25	0.8	0.3	25	0.9037	0.9206	0.0198	0.0082	129.8627s	**0.9243s**
25	0.7	0.1	25	0.9051	0.9205	0.0209	0.0086	130.7874s	1.7485s
25	0.7	0.2	25	**0.9114**	0.9187	0.0130	0.0100	130.0732s	1.5676s
25	0.7	0.3	25	0.9084	0.9205	0.0134	0.0086	135.4855s	1.6164s

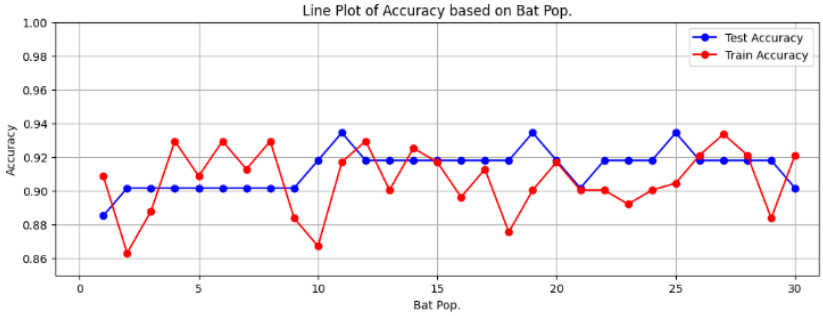
3.2 Best Parameter of the Proposed Algorithm

In this section, the performance of the proposed algorithm using various α and γ parameters to identify the optimal parameters for diagnosing heart attacks will be examined. The selection of optimal parameters is based on achieving a fast average computing time and high accuracy with minimal standard deviation. The selection of minimal standard deviation aims to reduce uncertainty in computing time. Test results, conducted with a fixed population size of 25 and a maximum iteration, revealed a commendable average accuracy on test data, approximately 92%, with a standard deviation ranging around 0.0085%. Concurrently, the computing time averaged around 130 seconds, accompanied by a standard deviation that exhibited variability ranging from 0.9 to 3.7 seconds, as detailed in **Table 2**.

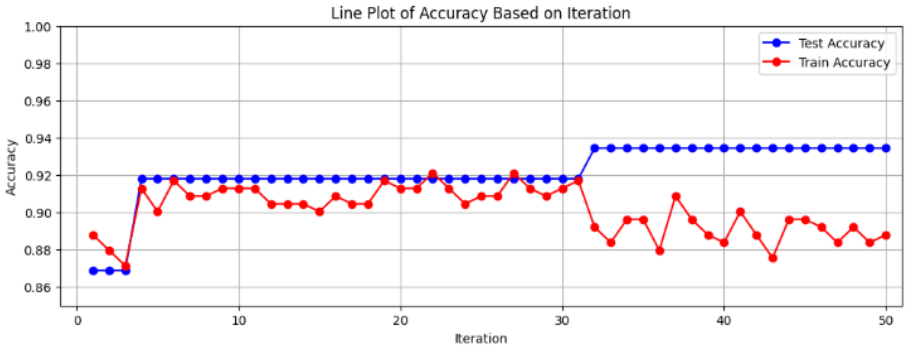
In **Table 2**, the results indicate that the optimal parameters are $\alpha = 0.9$ and $\gamma = 0.2$ as they yield the highest average accuracy on the test data, specifically 92.13%. However, it's noteworthy that this parameter combination resulted in the longest computation time compared to other parameter sets during program testing. In the context of heart attack diagnosis, where accuracy is prioritized over computational time due to the critical nature of the task, the decision was made to prioritize higher accuracy even if it meant a longer runtime. The marginal difference in computing time, ranging from 5 to 10 seconds, was deemed acceptable given the higher stakes involved in accurate heart attack diagnosis.

Testing the program with parameters $\alpha = 0.9$ and $\gamma = 0.2$ revealed that the bat population significantly influenced the accuracy. In **Figure 6a**, it is evident that a low bat population leads to a higher likelihood of the model achieving low test accuracy. Conversely, an increased bat population enhances the chances of achieving higher accuracy. However, it is crucial to note that a higher bat population does not guarantee higher accuracy. Based on **Figure 6a**, a bat population of 25 is identified as the optimal parameter, striking a balance between not being too low and offering the highest achievable accuracy.

During program execution with parameters $\alpha = 0.9$, $\gamma = 0.2$, and a population size of 25 bats, the test accuracy graph exhibited a monotonically increasing trend as iterations progressed. The peak accuracy of 93.44% was attained at the 32nd iteration, as depicted in **Figure 6b**. Based on these findings, it can be deduced that the optimal range for the maximum iteration parameter lies between 40 and 50. It was found that excessively small values for the maximum iteration parameter hinder bats from having ample opportunities to execute greedy crossovers. Consequently, this limitation may result in bats becoming trapped in local optima.



(a)



(b)

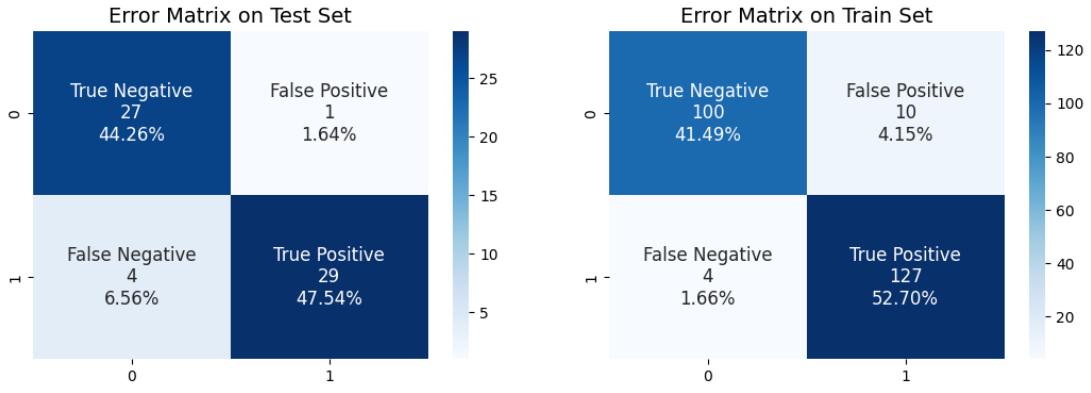
Figure 6. Plot of accuracy on test set based on parameter (a) bat population, (b) maximum iteration

3.3 Feature Selection

During program execution, several combinations of feature subsets have the potential to produce high accuracy. One essential feature subset, consisting of age, gender, cp, chol, thalach, oldpeak, slope, and ca, shows the high accuracy of 92.12% on train data and 93.44% on test data. This particular subset of features is identified by carefully examining the output, which shows high accuracy through the BBA with a greedy crossover algorithm. The inherent randomness in the construction of random forest structures plays an important role in achieving high accuracy. This randomness arises from the bootstrap sampling process used during model training, which aims to reduce overfitting [40]. This inherent uncertainty stemming from randomness can be addressed by iteratively building a Random Forest model and stopping the process when accuracy exceeds a predetermined threshold.

3.4 Model Evaluation

In this research, several metrics used to characterize the performance of the resulting Random Forest model. The program was executed ten times to get its average of evaluation metrics, which are accuracy, precision, recall, f_1-score, and confusion matrix as shown in Figure 7. Random Forest model worked very well on the test data because it had true negatives of 44.26% and true positives of 47.54%, which was quite high when evaluated. The model only predicted incorrectly on only 5 data which was only 8.2% of the total test data as seen in Figure 7 (a). The Random Forest model also does not overfitting because the difference in accuracy between train and test data is insignificant in size, reaching only 2.39%. This indicates that the model is able to work well with test data that has never been seen before.



(a) **(b)**

Figure 7. Confusion matrix of the Random Forest on (a) test set, (b) train set

During the precision, recall, and f_1-score evaluations, the Random Forest model demonstrated excellent performance, achieving precision and recall values exceeding 0.87 to 0.96 on both classes. Additionally, it yielded remarkably high f_1-scores, specifically 0.91 for class 1 and 0.92 for class 0, as shown

in Table 3. These results suggest that the Random Forest model is proficient in accurately predicting instances belonging to both classes.

Table 3. Performance of the Proposed Algorithm on Precision, Recall, and F_1-score

Label	Meaning	Precision	Recall	F_1-score
0	Patients are not at risk of having a heart attack	**0.96**	**0.96**	0.91
1	Patients are at risk of having a heart attack	0.87	0.87	**0.92**
	Macro average	0.92	0.92	0.92
	Weighted average	0.92	0.92	0.92

3.5 Comparative analysis of Random Forest with and without BBA with Greedy Crossover

During the precision, recall, and f_1-score evaluations, the Random Forest model with BBA and greedy crossover performed better than the Random Forest without the optimization. The results in Table 4 show improvement in precision, with a 0.05 difference in macro average recall, a 0.05 difference in macro average, and a -score, with a 0.05 difference in macro average. These results suggest that the proposed model performs better than the regular Random Forest model.

Table 4. Performance of the Proposed Algorithm on Precision, Recall, and F_1-score

Label	Meaning	RF with Optimization			RF without Optimization		
		Precision	Recall	F_1-score	Precision	Recall	F_1-score
0	Patients are not at risk of having heart attack	0.87	**0.96**	0.91	0.83	**0.89**	0.86
1	Patients are at risk having heart attack	**0.96**	0.87	**0.92**	**0.90**	0.85	**0.88**
	Macro average	0.92	0.92	0.92	0.87	0.87	0.87
	Weighted average	0.92	0.92	0.92	0.87	0.87	0.87

4. CONCLUSIONS

In this research, a variant of the BBA by incorporating a greedy crossover technique was introduced. This modification aims to mitigate rapid convergence to local optima during each iteration. The computational time of the proposed algorithm displays a linear relationship with the bat population and maximum iteration parameters, forming linear regression equations $C(t_{max}, 20) = 4.477t_{max} + 5.630$ when the bat population is constant (n_{20}), and $C(20, n) = 4.008n + 9.080$ when the maximum iteration is constant $(t_{max} = 20)$. Notably, larger bat populations and maximum iteration values lead to extended computational times required for feature selection within this algorithm. The proposed algorithm demonstrates a commendable average accuracy of 92.13% on the test set, accompanied by a standard deviation of 0.95%, under the parameter settings $\alpha = 0.9$ and $\gamma = 0.2$.

It is important to note that the computational time is marginally higher under these specific parameter configurations. Furthermore, the bat population and maximum iteration parameters significantly influence the algorithm's performance. Higher bat populations and maximum iterations correlate with an increased likelihood of achieving higher accuracy. Optimal parameters for the proposed algorithm in the context of heart attack diagnosis are identified as $\alpha = 0.9, \gamma = 0.2, n = 25$, and $t_{max} = 40$. Nevertheless, this does not guarantee heightened accuracy due to the Random Forest model's bootstrap sampling process during training.

The feature selection results with the proposed algorithm reveal that the subset comprising age, gender, cp, chol, thalach, oldpeak, slope, and ca yields the highest test accuracy. This particular feature subset empowers the Random Forest model to achieve accuracies of 94.19% on the training data and 91.8% on the test data. Evaluation metrics further indicate precision and recall values ranging from 0.87 to 0.96 in both classes, with an approximate f_1-score of 0.92. The proposed method model accuracy has increased by around 0.05 if compared with the regular Random Forest model.

This research significantly contributes to the medical field by assisting healthcare professionals in making better and timelier decisions regarding the diagnosis and treatment of heart attacks. It also plays a crucial role in planning more effective public health programs to prevent heart attacks. The findings of this research provide valuable insights that can enhance the quality of care for patients, improve treatment outcomes, and ultimately save lives.

REFERENCES

[1] P. Ghadge, V. Girme, K. Kokane, and P. Deshmukh, "Intelligent heart attack prediction system using big data," *International Journal of Recent Research in Mathematics Computer Science and Information Technology*, vol. 2, pp. 73–77, 2015, [Online]. Available: https://api.semanticscholar.org/CorpusID:252692313

[2] WHO, "Cardiovascular diseases." Accessed: Jul. 18, 2023. [Online]. Available: https://www.who.int/health-topics/cardiovascular-diseases#tab=tab_1

[3] Y. Kumar, A. Koul, R. Singla, and M. F. Ijaz, "Artificial intelligence in disease diagnosis: a systematic literature review, synthesizing framework and future research agenda.," *Journal of ambient intelligence and humanized computing*, vol. 14, no. 7, pp. 8459–8486, 2023, doi: 10.1007/s12652-021-03612-z.

[4] C. Nguyen, Y. Wang, and H.-N. Nguyen, "Random forest classifier combined with feature selection for breast cancer diagnosis and prognostic," *Journal of Biomedical Science and Engineering*, vol. 06, pp. 551–560, 2013, doi: 10.4236/jbise.2013.65070.

[5] S. M. S. Shah, F. A. Shah, S. A. Hussain, and S. Batool, "Support vector machines-based heart disease diagnosis using feature subset, wrapping selection and extraction methods," *Computers & Electrical Engineering*, vol. 84, p. 106628, 2020, doi: https://doi.org/10.1016/j.compeleceng.2020.106628.

[6] J. Zeniarja, A. Ukhifahdhina, and A. Salam, "Diagnosis of heart disease using k-nearest neighbor method based on forward selection," *Journal of Applied Intelligent System*, vol. 4, pp. 39–47, 2020, doi: 10.33633/jais.v4i2.2749.

[7] K. Vembandasamy, R. Sasipriya, and E. Deepa, "Heart diseases detection using naive bayes algorithm," *International Journal of Innovative Science, Engineering & Technology*, vol. 2, no. 9, pp. 441–444, 2015, [Online]. Available: https://ijiset.com/vol2/v2s9/IJISET_V2_I9_54.pdf

[8] M. Pal and S. Parija, "Prediction of heart diseases using random forest," *Journal of Physics: Conference Series*, vol. 1817, no. 1, p. 12009, Mar. 2021, doi: 10.1088/1742-6596/1817/1/012009.

[9] K. Kathirvelu, A. V. P. Yesudhas, and S. Ramanathan, "Spectral unmixing based random forest classifier for detecting surface water changes in multitemporal pansharpened Landsat image," *Expert Systems with Applications*, vol. 224, p. 120072, 2023, doi: https://doi.org/10.1016/j.eswa.2023.120072.

[10] P. S. Rao, P. Parida, G. Sahu, and S. Dash, "A multi-view human gait recognition using hybrid whale and gray wolf optimization algorithm with a random forest classifier," *Image and Vision Computing*, vol. 136, p. 104721, 2023, doi: https://doi.org/10.1016/j.imavis.2023.104721.

[11] M. R. Ali, S. M. A. Nipu, and S. A. Khan, "A decision support system for classifying supplier selection criteria using machine learning and random forest approach," *Decision Analytics Journal*, vol. 7, p. 100238, Jun. 2023, doi: 10.1016/J.DAJOUR.2023.100238.

[12] A. K, D. N, D. T, B. B B, B. D. N, and N. V, "Effect of multi filters in glucoma detection using random forest classifier," *Measurement: Sensors*, vol. 25, p. 100566, Feb. 2023, doi: 10.1016/j.measen.2022.100566.

[13] A. T. Azar, H. I. Elshazly, A. E. Hassanien, and A. M. Elkorany, "A random forest classifier for lymph diseases," *Computer Methods and Programs in Biomedicine*, vol. 113, no. 2, pp. 465–473, 2014, doi: https://doi.org/10.1016/j.cmpb.2013.11.004.

[14] M. Zhu, B. Su, and G. Ning, "Research of medical high-dimensional imbalanced data classification ensemble feature selection algorithm with random forest," in *2017 International Conference on Smart Grid and Electrical Automation (ICSGEA)*, 2017, pp. 273–277. doi: 10.1109/ICSGEA.2017.158.

[15] A. Chaudhary, S. Kolhe, and R. Kamal, "An improved random forest classifier for multi-class classification," *Information Processing in Agriculture*, vol. 3, no. 4, pp. 215–222, 2016, doi: https://doi.org/10.1016/j.inpa.2016.08.002.

[16] S. Han, H. Kim, and Y.-S. Lee, "Double random forest," *Machine Learning*, vol. 109, no. 8, pp. 1569–1586, 2020, doi: 10.1007/s10994-020-05889-1.

[17] Y. Ao, H. Li, L. Zhu, S. Ali, and Z. Yang, "The linear random forest algorithm and its advantages in machine learning assisted logging regression modeling," *Journal of Petroleum Science and Engineering*, vol. 174, pp. 776–789, 2019, doi: https://doi.org/10.1016/j.petrol.2018.11.067.

[18] T. M. Oshiro, P. S. Perez, and J. A. Baranauskas, "How many trees in a random forest?," in *Machine Learning and Data Mining in Pattern Recognition*, P. Perner, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 154–168.

[19] B. Lakshminarayanan, D. M. Roy, and Y. W. Teh, "Mondrian forests: Efficient online random forests." 2015.

[20] B. Wundervald, A. C. Parnell, and K. Domijan, "Generalizing gain penalization for feature selection in tree-based models," *IEEE Access*, vol. 8, pp. 190231–190239, 2020, doi: 10.1109/ACCESS.2020.3032095.

[21] I. M. B. Adnyana, "Penerapan feature selection untuk prediksi lama studi mahasiswa," *Jurnal Sistem dan Informatika*, vol. 13, no. 2, pp. 72–76, 2019.

[22] B. Pes, "Ensemble feature selection for high-dimensional data: a stability analysis across multiple domains," *Neural Computing and Applications*, vol. 32, no. 10, pp. 5951–5973, 2020, doi: 10.1007/s00521-019-04082-3.

[23] H. M. Farghaly and T. A. El-Hafeez, "A high-quality feature selection method based on frequent and correlated items for text classification," *Soft Computing*, vol. 27, no. 16, pp. 11259–11274, 2023, doi: 10.1007/s00500-023-08587-x.

[24] R. Ge *et al.*, "McTwo: a two-step feature selection algorithm based on maximal information coefficient," *BMC*

Bioinformatics, vol. 17, no. 1, p. 142, 2016, doi: 10.1186/s12859-016-0990-0.
[25] T. Verdonck, B. Baesens, M. Óskarsdóttir, and S. vanden Broucke, "Special issue on feature engineering editorial," *Machine Learning*, 2021, doi: 10.1007/s10994-021-06042-2.
[26] T. Islam, M. Islam, and M. R. Ruhin, "An analysis of foraging and echolocation behavior of swarm intelligence algorithms in optimization: ACO, BCO and BA," *International Journal of Intelligence Science*, vol. 08, pp. 1–27, 2018, doi: 10.4236/ijis.2018.81001.
[27] A. Chakri, R. Khelif, M. Benouaret, and X.-S. Yang, "New directional bat algorithm for continuous optimization problems," *Expert Systems with Applications*, vol. 69, pp. 159–175, 2017, doi: https://doi.org/10.1016/j.eswa.2016.10.050.
[28] A. Kaveh and P. Zakian, "Enhanced bat algorithm for optimal design of skeletal structures," *ASIAN JOURNAL OF CIVIL ENGINEERING (BHRC)*, vol. 15, pp. 179–212, Jan. 2014.
[29] H. Zhu, Y. Wang, and Y. Zhang, "Improved bat algorithm with novel search mechanism and one-dimensional perturbation local search strategy," 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:229347482
[30] M. A. Salam, "Comparative study between FPA, BA, MCS, ABC, and PSO algorithms in training and optimizing of LS-SVM for stock market prediction," *International Journal of Advanced Computer Research*, vol. 5, pp. 35–45, Mar. 2015.
[31] A. Kaur and Y. Kumar, "Recent developments in bat algorithm: a mini review," *Journal of Physics: Conference Series*, vol. 1950, p. 12055, Aug. 2021, doi: 10.1088/1742-6596/1950/1/012055.
[32] S. Akila and S. Allin Christe, "A wrapper based binary bat algorithm with greedy crossover for attribute selection," *Expert Systems with Applications*, vol. 187, p. 115828, 2022, doi: https://doi.org/10.1016/j.eswa.2021.115828.
[33] M. A. Farsi, "Chapter 3 - Genetic algorithms: Principles and application in RAMS," in *Nature-Inspired Computing Paradigms in Systems*, M. A. Mellal and M. G. Pecht, Eds., in Intelligent Data-Centric Systems. , Academic Press, 2021, pp. 25–46. doi: https://doi.org/10.1016/B978-0-12-823749-6.00001-5.
[34] S. Anam, M. R. A. Putra, Z. Fitriah, I. Yanti, N. Hidayat, and D. M. Mahanani, "Health claim insurance prediction using support vector machine with particle swarm optimization," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 17, no. 2, pp. 0797–0806, Jun. 2023, doi: 10.30598/barekengvol17iss2pp0797-0806.
[35] X. Ma and J. Wang, "Optimized parameter settings of Binary Bat Algorithm for solving function optimization problems," *Journal of Electrical and Computer Engineering*, 2018.
[36] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Computing and Applications*, vol. 25, no. 3, pp. 663–681, 2014, doi: 10.1007/s00521-013-1525-5.
[37] A. Hassanat and E. Alkafaween, "On enhancing genetic algorithms using new crossovers," *International Journal of Computer Applications in Technology*, vol. 55, Jun. 2017, doi: 10.1504/IJCAT.2017.10005868.
[38] R. Marappan and G. Sethumadhavan, "Complexity analysis and stochastic convergence of some well-known evolutionary operators for solving graph coloring problem," *Mathematics*, vol. 8, no. 3, p. 303, Feb. 2020, doi: 10.3390/math8030303.
[39] A. M. Aladdin and T. A. Rashid, "A new Lagrangian problem crossover—a systematic review and meta-analysis of crossover standards," *Systems*, vol. 11, no. 3, p. 144, Mar. 2023, doi: 10.3390/systems11030144.
[40] M. Schonlau and R. Y. Zou, "The random forest algorithm for statistical learning," *The Stata Journal: Promoting communications on statistics and Stata*, vol. 20, no. 1, pp. 3–29, Mar. 2020, doi: 10.1177/1536867X20909688.