Global Journal of Computer Science and Technology: C
Software & Data Engineering
Volume 23 Issue 1 Version 1.0 Year 2023
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Towards Optimized K means Clustering using Nature-Inspired Algorithms for Software Bug Prediction

By Kajal Tameswar, Geerish Suddul & Kumar Dookhitram

University of Technology (UTM)

Abstract- In today's software development environment, the necessity for providing quality software products has undoubtedly remained the largest difficulty. As a result, early software bug prediction in the development phase is critical for lowering maintenance costs and improving overall software performance. Clustering is a well-known unsupervised method for data classification and finding related patterns hidden in datasets. However, the k-means algorithm has the tendency to converge to local optima due to its sensitivity to its initial partition and random initialization of clusters centers. On the other hand, Nature-inspired algorithms (NIAs) are known for their general ability to establish global optima while searching around the whole search place. When these algorithms are combined with the K-means clustering mechanism, the novel hybrids are projected to yield outstanding results in terms of enhancing clustering quality by avoiding local optima and uncovering global optima. This study shows that the hybrid clustering of the Coral reefs algorithm outperforms the typical K-means specification in terms of prediction accuracy.

Keywords: *data clustering, K-means algorithm, Nature-inspired algorithms, software bug detection, coral reefs.*

GJCST-C Classification: *DDC Code: 005.1 LCC Code: QA76.76.D47*

Strictly as per the compliance and regulations of:

Towards Optimized K means Clustering using Nature-Inspired Algorithms for Software Bug Prediction

Kajal Tameswar [α], Geerish Suddul [σ] & Kumar Dookhitram [ρ]

Abstract- In today's software development environment, the necessity for providing quality software products has undoubtedly remained the largest difficulty. As a result, early software bug prediction in the development phase is critical for lowering maintenance costs and improving overall software performance. Clustering is a well-known unsupervised method for data classification and finding related patterns hidden in datasets. However, the k-means algorithm has the tendency to converge to local optima due to its sensitivity to its initial partition and random initialization of clusters centers. On the other hand, Nature-inspired algorithms (NIAs) are known for their general ability to establish global optima while searching around the whole search place. When these algorithms are combined with the K-means clustering mechanism, the novel hybrids are projected to yield outstanding results in terms of enhancing clustering quality by avoiding local optima and uncovering global optima. This study shows that the hybrid clustering of the Coral reefs algorithm outperforms the typical K-means specification in terms of prediction accuracy.

Keywords: data clustering, K-means algorithm, Nature-inspired algorithms, software bug detection, coral reefs.

I. Introduction

In an era of technological disruption, the demand for software adoption has accelerated. They are a part of our society and play an important role in shaping it. Our modern society is becoming increasingly reliant on complex software systems. Thus, it is critical to build reliable and trustworthy systems in a cost-effective and timely manner. The presence of defective modules in a software drives up development and maintenance expenses, leading to customer dissatisfaction. The need for quality assurance has inevitably remained the biggest challenge in today's software development environment Hence, software bug prediction is an important task to help developers locate bugs more efficiently.

Software bug prediction is an imperative task in Software Development Life cycle (SDLC) as it pertains to the overall success of software. One method in this direction is to use machine learning (ML) methods to predict defects in software. In addition, implementing this method earlier in the SDLC process enhances quality of the product and lowers the cost of software maintenance. Many researchers have applied different theories and methodologies in the field of software bug prediction. Two things are clear from the literature when it comes to defect prediction. Initially, no single prediction approach dominates (Lessmann et al.,2008), and next, the employment of various set of data, data pre-processing, validation systems, and performance statistics makes it challenging to make sense of the multiple prediction outcomes (Myrtveit et al.,2005). There are two common ML model used for prediction based on dataset availability. The first, known as supervised approach, in which a software defect prediction model is built from training set of data and then tested on a testing dataset. Secondly, unsupervised approach, in which the defect prediction model for software is built from scratch using the present testing dataset without training the dataset.

Clustering algorithms have been commonly used to evade the lack of training datasets available being a constraint. Cluster analysis groups things into clusters based on their similarity to create a visual representation of data (Jain and Dubes, 1998). As pointed out by Kaur, 2010, one of the better instances of unsupervised learning is K-means clustering. Clustering is beneficial because it makes it easier to obtain or locate relevant information at a faster rate. Among the different clustering approaches that already exist, the K-means methodology is obviously fairly popular. (Gayathri et al., 2015). The preliminary values of the initial centroids, which are generated randomly each time the algorithm is run, have a significant impact on the performance of k-means. K-means frequently fall into local optima that produce poor clustering results. Obtaining a globally optimal clustering result involves a time-consuming, exhaustive approach that tests all partitioning choices. A heuristic approach to the problem is to use an optimization algorithm to search for global optima in each computer iteration.

Our unsupervised approach uses the k-means approach to divide the unlabeled dataset into defective and non-defective non-overlapped clusters for bug prediction. The goal of this research is to verify the hybrids' efficacy as well as to quantify the quality of results produced by each clustering hybrid model. In

Author α σ ρ: University of Technology (UTM), Pointe-aux-Sables, Mauritius. e-mails: ktameswar@umail.utm.ac.mu, g.suddul@umail.utm.ac.mu, kdookhitram@umail.utm.ac.mu

this study, we have applied the k-means clustering algorithm, an unsupervised algorithm with different NIAs including Genetic algorithm (GA), Bat algorithm (BA), Particle Swarm Optimization (PSO), Coral Reefs Optimization (CRO), Cuckoo Search optimization (CSO) algorithm, Ant colony optimization (ACO), Firefly algorithm (FA) and Grey Wolf Optimizer (GWO) for software bug prediction. The rest of this paper is organized as follows. Section 2 presents a discussion of the related work in software bug prediction. An overview of the methodology, consisting of the algorithms used are presented in Section 3. Section 4 describes the proposed method. Section 5 describes the Dataset and Data Processing method. The evaluation methodology is discussed in section 6. The results and discussion part is discussed in Section 7. Section 8 discusses the practical implications followed by conclusions and future works in section 9.

II. Related Works

K-means clustering is a well-known partitioned clustering algorithm that has been used in a variety of applications. In the literature, several variations of K-means have been proposed to improve its performance for the broad clustering problem. Fong et al. (2012) studied the integration of bio-inspired optimization methods into K-means clustering for software bug prediction in order to assess clustering performance. The main optimization algorithms tested include the Firefly algorithm, Cuckoo search algorithm, Bat algorithm, Wolf and Ant Colony Optimization (ACO) algorithms. Results show that the combination of these algorithms acquired improved performance accuracy compared with ordinary k-means, at the same time accelerating the search process and avoid local optima. Zhong et al.,2004 compared the k-means algorithm to natural-gas algorithms. The natural gas algorithm outperformed the k-means algorithm in terms of mean square error values. However, this method necessitates the use of a software expert to determine whether the software is appropriate.

Annisa et al., 2020, came up with an improved version of k-means algorithm for software bug prediction, that locate the initial centroid of the k-means algorithm and determine the number of clusters present. Because it produces better accuracy than the simple K-Means method, this proposed method could be useful for clustering other data types. Seliya and Khoshgoftaar, 2007 proposed K-means for software failure prediction. Their method iteratively labels clusters as fault-prone or not using expert domain knowledge as a restriction.

The k-means algorithm based on quad tree was proposed by Bishnu and Bhattacherjee, 2012 and it was compared to some clustering algorithms. Their proposed algorithm has error rates that are comparable to k-means, Linear Discriminant Analysis and Naive Bayes. Catal et al. 2009 used the x-means clustering algorithm to create faulty and non-faulty clusters based on software metrics. Lines of code, cyclomatic complexity, operand and operator are the metrics. If the metric values are complex than the threshold, the software entity is predicted to be defective, and vice versa. Almayyan, 2021 used dataset from the NASA repository and used three clustering algorithms, Farthest First, X-means and Self-organizing map. This article presents a comparison of software defect prediction algorithms based on Bat, Cuckoo, Grey Wolf Optimizer (GWO), and Particle Swarm Optimization (PSO) in order to evaluate different feature selection algorithms. The Farthest First clustering algorithm was found to be effective in predicting software faultiness, and Bat and Cuckoo were found to be useful in comparison to all other metaheuristic algorithms.

Though several academics have sought to merge K-means clustering with nature-inspired algorithms (NIAs), their efforts have been restricted to almost identical group movements, such as the Firefly, Artificial Bee Colony (ACO), and Particle Swarm Optimization (PSO) algorithms (Jensi and Jiji, 2015). In addition, only a few bio-inspired optimization methods that are integrated with K-means are provided in the previous studies. Only 7 of the 28 NIAs hybridized with K-means (Genetic Algorithm, Particle Swarm Optimization, Bat Algorithm, Artificial Bee Colony, Differential Evolution, Harmony Search, and Symbiotic Organism Search) dedicated their hybridization to solving automatic clustering problems, accounting for 20.6 percent of the total (Ikotun et al., 2021). In general, it can be seen that the rate of publishing on K-means hybridization with specific NIAAs is minimal. More research is needed in this area to see if there are any other ways to improve the performance of the existing hybridization algorithm. This suggests that combining K-means with these other NIAs to solve automatic clustering problems should be investigated.

The purpose of this research is to look into the mechanics of incorporating certain NIAs into the K-means clustering algorithm. The optimization function adds to the existing best solution by progressively improving it with a new solution from an unknown fragment of the search space. When a new solution is identified to be better than the present one, the searching agents replace the solutions and continue searching until some stopping criteria are fulfilled.

III. Methodology

a) K means Clustering Algorithm

The K-means clustering algorithm is a partitioned clustering technique that divides a dataset into k number of clusters using a certain fitness measure. Due to the large amount of data objects in real-world datasets, distributing data items into

appropriate clusters to obtain an ideal cluster outcome is computationally expensive and time-consuming (Ikotun et al.2021).

Given a dataset X = {x_i}, where i = 1, 2, . . . n of d-dimension data points of size n, X is partitioned into 'k' clusters such that

$$J(c_k) = \sum_{xieck} ||x_i - \mu_k||^2 \quad (1)$$

With the objective function: minimize the sum of the square error over all the k clusters. That is, minimize

$$J(C) = {}^{K}\sum_{k=1} \sum_{xieck} ||x_i - \mu_k||^2 \quad (2)$$

When assigning N objects to k clusters, the purpose of the clustering algorithm is to limit the number of potential possibilities. This can be expressed numerically as:

$$S(N, K) = 1/K! \; {}^{K}\sum_{i=0} (-1)^{K-i} \binom{K}{t} t^N \quad (3)$$

b) Nature-inspired algorithms (NIAs)

Nature-inspired computation has gained popularity in the previous two decades and has been used in practically every field of research and engineering (Yang et al.2013). NIAs are global optimization strategies for solving difficult real-world issues (Okwu et al. 2020). NIAs have successfully provided suboptimal solutions to automatic clustering problems in a reasonable amount of time (Hruschka et al. 2009). The population is used for the exploration of search space in the nature-inspired metaheuristic, ensuring a higher possibility of finding optimal cluster partitions (Nanda and Panda, 2014). It has been discovered that combining K-means with NIAs for automatic clustering improves the performance of algorithms when dealing with cluster analysis. In most circumstances, the automatic cluster number determination aids in the selection of near-optimal starting cluster centroids for the clustering process rather than the normal random selection (Zhou et al. 2017).

c) Combination of k-means with Nature-Inspired Algorithms (NIAs)

Clustering using NIAs is now as simple as assigning combinations of centroids to the searching agents, allowing them to heuristically find the best answer. Though the specifics of conducting a heuristic search vary depending on which nature-inspired optimization algorithm technique is used, the initialization stage and the finishing step, where the quality of the discovered solution is evaluated as a stopping condition, are both comparable.

S is defined as the solution space that contains a finite number of x_i, where i is the solution's index, in the initialization construct. The search agents represent the solutions x, each of which holds a set of centroids, regardless of the types of bio-inspired optimization methods used.

Typically, a large population of searching agents, N, is utilized to collaboratively search for the best feasible cluster configurations (as expressed by the locations of the optimal centroids). K is the number of clusters that must be formed, which is generally a user-defined figure. D is the dimension of the search space, which is the number of attributes a data point possesses.

To find the optimal configuration of centroids we let $cen_{j,v}$ be the centroids at the j^{th} cluster and the v^{th} attribute. To obtain the centroid location, the following formula is used:

$$cen_{j,v} = \Sigma^{S}_{i=1} w_{i,} {}^{jX}i,v / \Sigma^{S}_{i=1} w_{i,} {}^{jX}i \;, \; Where\ j = 1\ldots K,\ v = 1\ldots K * D \quad (4)$$

In our concept, the matrix cenj,v contains all of the cluster centers and is a two-dimensional matrix with K * D characteristics.

$$F(cen) = \Sigma^{k}_{j=1} \Sigma^{S}_{i=1} W_{i,j} \Sigma^{K*D}_{v=1} (X_{iv,} - cen_{j,v})^2 \quad (5)$$

The calculation method loops K * D times to analyze the values of all the attributes of x in each cluster v to calculate the distance between each x and the centroid.

Cluster centers can be designated by data points. For example, in a two-cluster clustering task, the objective function requires three variables. As a result, there are three dimensions.

Three variables, and hence three-dimensional spaces, are required, and the i^{th} data point may be written as x_i= (i, [$x_{i,1}$, $x_{i,2}$, $x_{i,3}$, $x_{i,4}$, $x_{i,5}$, $x_{i,6}$]).

The clustering strategy can be formulated as follows:

$$clmat_{i,j} = \min_{k\varepsilon k} \{||X_i - cen_k||\} \quad (6)$$

Where i=1...N, j=1...S, k=1…K. Equation (3) tells us that the i^{th} data point belongs to the kth cluster. The equation is an objective function with a lower value indicating better performance.

Sets of functional parameters must be defined in order to execute the bio-inspired optimization algorithms. Despite the fact that some of their parameters are shared, each set of parameters for the hybrid bio-inspired clustering algorithms is designed independently. The six models investigated are K means with Genetic Algorithm, K means with Bat algorithm, K means with Ant colony algorithm, K means with Cuckoo Search Algorithm, K means with Firefly Algorithm and K means with Coral reefs algorithm. The most significant variations are in how the global optimal exploration is carried out for all these algorithms. The evaluation stage comes right after the exploration construct, and it compares if the new solution is better than the current best one.

d) Genetic Algorithm

Genetic Algorithm (Ga) are randomized heuristic search algorithms that are based on natural

selection and genetic principles (Goldberg, 1989). The genetic operators used in the combination of K-means and GA are selection, distance-based mutation, and the K-means operator. The parameters have been set according to the study of Bouhmala et al. 2015.

P (0) is chosen at random as the starting population. Each allele in the population can be given a cluster number from the uniform distribution over the set {1,…. K) at random.

According to the distribution given by, the selection operator selects a chromosome from the preceding population at random as follows:

$$P(s_i) = F(s_i) / {}^{N}\sum_{j=1} F(s_j) \quad (7)$$

Where $F(s_i)$ represents fitness value of the string s_i in the population.

The possibility of solutions surviving in the future population is ranked in the current population. Each solution in the population must be assigned a figure of merit or a fitness value.

$$F(s_W) = \{ {}^{g(sW)}{}_0 \text{ ; if } g(s_W) \geq 0 \text{ ; otherwise.} \quad (8)$$

e) Bat Algorithm (BA)

Bat echolocation is used in the bat algorithm (BA), which is a heuristic optimization tool (Yang, 2010). The four basic parameters of a BA are pulse frequency, pulse rate, velocity, and a constant. The parameters have been set according to the study (Huang and Ma, 2020).

The frequency, velocity, and position for each bat are initialized. The virtual bats' movement is described by updating their velocity and position using the equations below for each time step t, where T is the iteration limit.

$$f_i = f_{min} + (f_{min} - f_{max})^{\beta} \quad (9)$$

$$V_i^{t+1} = v_i^{t} + [X_i^{t} + X^*]f_i \quad (10)$$

$$X_i^{t+1} = X_i^{t} + v_i^{t} \quad (11)$$

Where V_i^t and X_i^t are the velocity and position at time t, V_i^{t+1} and X_i^{t+1} are the velocity and position at time t+1, and is a random number between 0 and 1.

A random number is generated when the bat positions are updated; if the random number is greater than the pulse emission rate, a new location is formed around the current best solutions, as shown in the equation below.

$$X_{new} = x_{old} + EA^t \quad (12)$$

Where E is a random number A^t represents the average loudness of all bats at time t.

f) Ant Colony Optimization (ACO)

The ACO heuristic was inspired by investigations of ant foraging behavior in real colonies, which indicated that ants can often figure out the shortest path between food source and nest (Zheng et al. 2003). The parameters have been set according to the study (Tang et al. 2012).

When the ant moves from i to j, the path node at the start can set as A, A= {0, 1,…… n-1}. This reflects the role of pheromones accumulated by ants during exercise during ant migration and reveals the relative relevance of the trajectory. The larger α is, it indicates the high probability for subsequent ants to choose this path.

The probability of the ant moving from I to j is computed using the following formula:

$$P_{ij}^{k}(t) = r_{ij}^{k}(t) n_{ij}^{\beta}(t) / \sum r_{ij}^{\alpha}(t)\, n_{ij}^{\beta}(t) \quad (13)$$

Where pheromone is ρ, which is a constant that represents weight. The time of iteration is Nc and the initial setting is ϕ. The predicted heuristic factor isβ, which demonstrates the relevance of visibility relative to other factors. It also represents the significance of the heuristic component in the entire path of the ant's movement.

g) Firefly Algorithm (FA)

Firefly algorithm is a very strong technique for solving restricted optimization and NP-hard problems (Apostolopoulos and Vlachos, 2011). The parameters have been set according to the study (Tang et al.2012).

The attractiveness of a firefly I on a firefly j is determined by the degree of the firefly i's brightness and the distance rij between the firefly I and the firefly j, as shown below:

$$I(r) = I_s / r^2 \quad (14)$$

Consider the case when there are n fireflies and the solution for firefly I is xi. The brightness of the firefly I is linked to the objective function f (xi).

$$I = f(x_i) \quad (15)$$

Each firefly has an attraction value, and the less dazzling (attractive) one is drawn to the brighter one and transferred there. The attractiveness value β is relative based on the distance between fireflies.

$$\beta(r) = \beta_0 e^{-\gamma r^2} \quad (16)$$

Where β_0 is the firefly attraction value at r = 0 and γ is the media light absorption coefficient.

h) Cuckoo Search (CS) Algorithm

Yang and Deb, 2009, developed the Cuckoo Search algorithm which is based on some cuckoo species' brood parasitism. The parameters have been set according to the study (Fong et al. 2014).

An initial population of n nests is randomly generated at the positions, $X = \{x^0_1, x^0_2, \ldots, x^0_n\}$, to evaluate the objective values to find the current global best g_t^0.

The new position is updated accordingly by performing a Lévy flight:

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus \text{Lévy}(\lambda), \quad (17)$$

Where $\alpha > 0$ denotes the step size, which should be connected to the problem's scales. In most circumstances, we can use $\alpha = 1$.

i) Coral Reefs Optimization Algorithm (CRO)

CRO is another nature-inspired algorithm, based on an artificial simulation of the process of coral reef formation and reproduction (Sanz et al.2014). The CRO algorithm has never been utilized in the realm of software bug detection to our knowledge. Corals reproduce at each iteration step in the CRO algorithm, producing new individuals. The parameters have been set according to the study (Medeiros et al., 2015).

By allocating a coral to each square (i j), the CRO algorithm generates a N x M square grid in which each square (i,j) may represent an alternate solution to a problem (or colony of corals). The formation of coral is the second phase. After three phases, the entire collection of existing corals in the reef is graded according to their level of healthiness (broadcast spawning, brooding, and larvae setting).

j) Particle Swarm Optimization (PSO)

The behavior of particles in a swarm is the central concept of the PSO. Each particle has its own location in a multidimensional space and communicates with the others. To move about in space, the particles employ social and cognitive information. When the algorithm comes to a halt, the best solution has been discovered (Koohi and Groza, 2014). The parameters have been set according to the study (Rana et al., 2010).

The inertia weight balances the algorithm's local and global search abilities. The proportional contribution of the prior velocity to the current velocity is defined by the inertia weight.

$$V_i^{k+1} = wv_i^k + c1\ \text{rand}\ (p_{besti} - x_i^k) + c2\ \text{rand}\ (g_{best} - X_i^k) \quad (18)$$

$$X_i^{k+1} = X_i^k + v_i^{k+1} \quad (19)$$

k) Grey Wolf Optimizer (GWO)

The Grey Wolf Optimizer (GWO) is a simple, population-based, flexible, and derivative-free meta-heuristic optimization method that intelligently avoids stagnation in local optima spots of the search space. It simulates the social behaviors of grey wolves in the aspects of their hierarchical leadership and hunting movement (Mirjalili et al., 2013). Grey wolves' leadership and haunting mechanism help to design a new metaheuristic algorithm with three steps: searching prey, encircling prey, and attacking prey.

During the GWO operation, the position of the wolves is continuously updated, with appropriate mathematical formulas (Hou et al., 2022). The parameters have been set according to the study (Wang et al., 2019).

IV. Proposed Method

To address the curbs of the K-means clustering approach in generating globally optimum clusters, the suggested method uses the k-means algorithm together with a range of NIAs for software bug prediction. By adding an exploration function to the k-means algorithm, the combination of these strategies may improve the model. The exploration function improves the existing solution by examining regions outside of its immediate vicinity, and if a new, better solution than the current best one is discovered, the search agents will move toward it. The exploring procedure will continue until certain stopping criteria are met. Nature-inspired algorithms are metaheuristic algorithms, which means they have the ability to explore the combinatorial search space heuristically rather than exhaustively. The integration methods are based on representing the search agents as a combination of centroid locations, then the search agents explore the search space for the best solution.

The purpose of clustering is to discover a proper set of centroids using the metaheuristic of the nature-inspired method as a guide. The metaheuristic will always insist on centroids being moved in a progressive manner in each phase, with the goal of finding the best grouping. The ideal group's ultimate result should be that the data points inside each cluster are closest to their centroid. During the search, the centroids move around in the search space, following the swarming pattern of the nature-inspired optimization method, until no further progress is seen. It comes to a halt when there is no other possible relocation that will yield a better result. Along with the success of employing nature-inspired metaheuristic algorithms to solve automatic clustering problems, it has been discovered that combining two or more metaheuristics for the same objective improves clustering performance. The performance of hybrid algorithms, according to Nanda and Panda 2014, is superior to that of separate algorithms in terms of robustness, effectiveness, and accuracy.

V. Dataset and Data Processing

The dataset was collected from the online PROMISE repository. AR1, AR3, AR4, AR5, AR6, KC1, KC2, JM1, CM1, PC1 and PC5 were used respectively. With reference to the paper, by Shepperd et al. 2013, data cleaning is mandatory before using any datasets available. Indeed, we noted a huge class imbalance issue with the available datasets (faulty, non-faulty),and all data inconsistencies, missing and null values were removed. Each dataset selected represents a NASA software system that includes various metrics. Each dataset is made up of a number of software modules and attributes. Modules with defects are classified as prone to faults, whereas those without defects are

classified as non-fault prone. For the training purpose, the entire dataset is used except for the last column (output column), only columns consisting of numerical values were considered.

Table 1: Summary of dataset

Dataset	Modules	Defective modules	Software metrics (Attributes)
AR1	121	9	29
AR3	63	8	29
AR4	107	20	29
AR5	36	8	29
AR6	101	15	29
KC1	2109	1783	22
KC2	522	107	21
JM1	7782	1672	21
CM1	327	42	37
PC1	705	61	37
PC5	1711	471	38

VI. Evaluation

a) Experimental Setup

The main goal of this research is to demonstrate the utility of the k-means algorithm with different NIAs, which we accomplished using Tensorflow to train the model. TensorFlow is an open-source machine learning platform to build and deploy prediction models. Google Colab was also used to run the results, which allowed the code to run with no configuration and free GPU access. Each dataset is performed 10 times in the trials to find the average CPU time and objective function values/best fitness value. The clustering results of the new hybrid clustering algorithms are compared to the K-means, which serve as a benchmarking reference. The full dataset is used for training, and cluster formation is referred to until perfection is attained using the entire set of data. The ultimate clustering result's quality is determined by each cluster's integrity, which is represented by the objective function's final fitness value.

The hardware configuration used for all experiments in this study is as follows: Corei7-6500U CPU @2.50 GHz 2.60 GHz, Windows 10, 64-bit operating system, x64 based processor, RAM: 8 GB DDR4, and Hard Disk: SSD.

b) Performance Evaluation Measures

In order to assess the effectiveness of combining the k-means algorithm and optimization algorithms in the prediction of software bugs, the evaluation metrics, accuracy and F-measure have been calculated accordingly as shown in the Equation (1):

$$Accuracy = (TP+TN) / (TP+TN+FP+FN), \quad (20)$$

Where TP = true positive, TN = true negative, FN = false negative and FP = false positive.

On the other hand, the external metric used to determine the accuracy of the clustering findings, known as the F-measure, is also computed.

The F-measure, which is the average of precision and sensitivity performance, is calculated as follows:

$$F = 2 * P * Sensitivity / P + Sensitivity, \quad (21)$$

Where P refers to precision and sensitivity is calculated by finding the non-defective modules that were accurately categorized.

VII. Results and Discussions

Table 2: Accuracy of algorithms

Datasets	AR1	AR3	AR4	AR5	AR6	KC1	KC2	JM1	CM1	PC1	PC5
k-Means	88.90	88.00	89.01	88.85	88.43	89.10	89.00	88.80	89.00	89.19	89.99
K-Means +GA	90.50	90.58	91.28	91.55	90.11	90.00	90.54	90.53	91.25	90.00	90.05
K-Means +BAT	90.00	91.59	91.00	92.34	92.00	92.98	91.34	90.00	91.25	92.56	92.00
K-Means +PSO	92.50	92.65	92.87	93.01	93.00	92.99	94.10	92.67	92.89	93.10	93.58
K-Means +Coral Reefs	94.00	94.54	94.56	94.87	94.00	95.96	95.66	96.88	95.01	95.04	95.54
K-Means +Cuckoo	94.50	94.58	94.58	94.00	94.56	95.45	95.88	95.67	95.44	94.56	94.78
K-Means + ACO	94.00	93.56	93.50	94.10	93.78	93.03	93.56	93.44	93.89	94.01	94.52
K-Means +Firefly	92.56	92.67	93.00	93.44	93.02	93.56	94.78	93.67	94.88	94.34	94.54
K-Means + GWO	90.09	92.47	94.65	93.22	92.00	92.60	93.00	92.50	94.50	94.12	94.13

From the table above, K-means clustering is optimized using the various NIAs. We can see that all of the proposed algorithms perform better than the traditional standalone k-means algorithm. K-means appears to take the shortest computation time in any of the tests, maybe because it stops early in local optima (Table 3). This is evident from the accuracy obtained from the table above. NIAs speed up the process of clustering centroids and illustrate that all partitioning clustering methods can be linked with the natural search

process to prevent local optima. Secondly, simple K-means were applied to the robust nature of GA, which shows adequate prediction accuracy for all datasets. Even though GA may converge to the global optimum due to mutation, GA faces the issue in terms of computational challenges. The application of k means with the Bat algorithm apparently yields the same accuracy. This hybrid algorithm improves the convergence speed of BA and helps the k means algorithm independent of the initial centers. Next, K means is combined with PSO. The PSO method is used to start the process because of its fast convergence, and then the K-Means algorithm is used to refine the PSO algorithm's outcome to near-optimal solutions. The hybridization of these two methods yields effective results in terms of efficiency and precision. The PSO algorithm can be used to generate good initial cluster centroids for the K-Means.

Furthermore, K means and Coral reefs algorithm are combined. The results for this combined method are quite promising since they show that using the CRO method for a clustering application can produce better results to using hybrid genetic algorithms, which is the most often used clustering optimization technique. To best of our knowledge, CRO has not been used with clustering for software bug detection. The hybrid model of k means with Cuckoo Search algorithm shows significant accuracy, likewise CRO algorithm. Cuckoo search is used to provide a robust initialization, whereas K-means is utilized to construct solutions faster. K means is also combined with Ant Colony Optimization algorithm. The suggested method's learning mechanism is based on the use of a defined parameter termed pheromone, which eliminates undesirable K-means algorithm solutions. The suggested method improves the K-means algorithm by making it less reliant on starting parameters such as randomly picked beginning cluster centers, resulting in a more stable algorithm. K means with firefly also produce near accuracy with CRO and Cuckoo search algorithm. This is because fireflies with high similarity are dispersed, resulting in a more diverse distribution of the entire swarm in search space. K means with GWO has also shown rapid convergence. This improvement is caused by the fact that K-means significantly affects the GWO population and separates it into two clusters. Because GWO often operates as three clusters and has three wolves in the search space, K-means is advantageous for GWO.As a result, it can be concluded that K- means combined with GWO increased GWO's effectiveness to some extent.

High clustering accuracy and efficiency were obtained from the hybrid clustering of Coral reefs and Cuckoo Search Algorithm. Hybrid clustering of Coral reefs algorithm has never been applied in the field of software bug detection and has indeed shown promising results. Hybrid clustering of Coral reefs algorithm locate cluster centroids without causing premature convergence. The findings of the evaluation results add evidence that NIAs can indeed speed up the process and avoid local optima.Because fewer iterations are required to achieve the best cluster outcome, selecting the number of clusters enhances the hybridized clustering method's convergence speed. The computational time for each algorithm is computed as shown in Table 3. Less computational time was noted when K means was integrated with Coral reefs and Cuckoo Search algorithm respectively.

Table 3: Computational Time for all Algorithms

Datasets	AR1	AR3	AR4	AR5	AR6	KC1	KC2	JM1	CM1	PC1	PC5
k-Means	79.88	80.01	79.23	80.10	80.15	79.99	79.98	80.65	80.00	79.80	80.34
K-Means +GA	159.99	157.85	162.89	161.00	162.00	172.45	170.55	169.87	172.99	169.00	170.03
K-Means +BAT	162.40	161.00	160.88	165.54	166.87	164.34	157.88	169.90	161.45	162.34	165.10
K-Means +PSO	165.78	155.00	168.98	172.99	170.00	169.99	159.00	159.90	172.78	172.00	169.56
K-Means +Coral Reefs	148.89	152.77	148.54	150.00	149.90	155.42	150.09	151.23	147.77	148.99	149.00
K-Means +Cuckoo	146.67	146.90	149.45	148.00	145.78	148.00	149.99	148.45	150.45	146.88	149.00
K-Means+ ACO	162.67	166.34	159.90	155.67	161.88	160.10	160.00	168.89	159.45	158.45	158.00
K-Means +Firefly	150.23	152,90	150.00	151.90	155.67	150.45	152.56	154.78	152.89	155.90	154.78
K-Means+ GWO	151.45	151.00	156.40	156.34	156.12	154.98	153.10	154.88	154.00	151.17	154.97

For statistical performance, the F1 score has been calculated for all the algorithms as shown in Table 4. Again, the F1 Score shows that K-means with Coral reefs resulted in dependable and significant performance that can be used to predict software defects. When a good validity measure is applied, most metaheuristic algorithms can automatically divide datasets into an appropriate number of clusters, according to Gbaje et al.2019.

Table 4: Statistical Performance Analysis of Algorithms- F1 Score

Datasets	AR1	AR3	AR4	AR5	AR6	KC1	KC2	JM1	CM1	PC1	PC5
k-Means	0.66	0.79	0.82	0.80	0.75	0.81	0.80	0.81	0.82	0.82	0.80
K-Means+GA	0.84	0.83	0.83	0.80	0.83	0.84	0.84	0.85	0.82	0.81	0.85
K-Means +BAT	0.83	0.81	0.83	0.86	0.86	0.86	0.85	0.85	0.85	0.87	0.85
K-Means +PSO	0.85	0.85	0.87	0.87	0.86	0.85	0.87	0.85	0.87	0.87	0.87
K-Means +Coral Reefs	0.86	0.86	0.86	0.85	0.86	0.86	0.87	0.88	0.86	0.87	0.88
K-Means +Cuckoo	0.89	0.85	0.88	0.89	0.86	0.89	0.86	0.89	0.89	0.87	0.88
K-Means+ ACO	0.84	0.83	0.86	0.85	0.84	0.86	0.85	0.85	0.86	0.85	0.86
K-Means +Firefly	0.86	0.85	0.83	0.87	0.87	0.85	0.83	0.85	0.86	0.88	0.85
K-Means+ GWO	0.82	0.82	0.81	0.86	0.84	0.83	0.79	0.85	0.84	0.84	0.85

VIII. Practical Implications

Metaheuristics algorithms have shown to be effective optimizers. This research found that each of the hybrid K means based-nature-inspired optimization algorithm models outperformed the standalone K means algorithm in terms of accuracy and F1 score. Following the intrinsic limitations of K-means design and the virtues of Nature-inspired optimization techniques, it seems feasible to integrate them, allowing them to complement and function together. The algorithms' successful integration gives reason to believe that more advanced optimization mining techniques can be developed. This study can be used as a roadmap for researchers who want to incorporate other new emerging NIAs into improved clustering methods in the field of software bug detection.

IX. Conclusion and Future Works

Prediction of defect-prone software modules is an important goal in software engineering. The traditional clustering algorithm usually gets trapped in the problem of local optima. As a result, the nature-inspired method provides an alternative technique for solving clustering problems using its searching capabilities. This study's main contribution is combining the clustering algorithm with the different NIAs for software bug detection. To the authors' knowledge, only PSO, Cuckoo, Bat, and GWO (Grey Wolf Optimizer) algorithms were applied with clustering algorithms for software bug detection (Almayyan, 2021). The results are improved significantly when clustering algorithms are combined with bio-inspired optimization methods, apparently for the hybrid model of k means clustering withCoral reefs algorithm, achieving an accuracy of 96%.For future work, this work can be replicated with other related datasets for the analysis of bug prediction in software.

Acknowledgments

This study received no formal support from public, private, or not-for-profit funding organizations.

References Références Referencias

1. J.B.Macqueen, "Some methods for classification and Analysis of Multivariate Observations". Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, 1967, pp 281-297
2. Rui Tang, Simon Fong, Xin-she Yang and Suash Deb, Integrating Nature inspired Optimization algorithms to k-means clustering, University of Macau, Aug 2012
3. RiskiAnnisa, DidiRosiyadi, Dwiza Riana, Improved point center algorithm for k-means clustering to increase software defect prediction, International Journal of Advances in Intelligent Informatics, Vol 6, No.3, November 2020, pp.328-339
4. S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings," IEEE Transactions on Software Engineering, vol. 34, no. 4, pp. 485–496, 2008.
5. I. Myrtveit, E. Stensrud, and M. Shepperd, "Reliability and validity in comparative studies of software prediction models," IEEE Transactions on Software Engineering, vol. 31, no. 5, pp. 380–391, 2005.
6. Jain, A.K. and R.C. Dubes, Algorithms for clustering data. 1988: Prentice-Hall, Inc.
7. Gayathri, R., Cauveri, A., Kanagapriya, R., Nivetha, V., Tamizhselvi, P., & Kumar, K. P. (2015, March). A Novel Approach for Clustering Based On Bayesian Network. In Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering & Technology (ICARCSET 2015) (p. 60). ACM.
8. Zhong S, Khoshgoftaar TM, Seliya N. Unsupervised learning for expert-based software quality estimation. In: Proceedings of the eighth IEEE international conference on high assurance systems engineering HASE 2004; 2004. p. 149–55. https://doi.org/10.1109/HASE.2004.1281739.

9. Bishnu PS, Bhattacherjee V. Software fault prediction using quad tree-based kk-means clustering algorithm. IEEE Trans Knowl Data Eng. 2012; 24(6):1146–50. https://doi.org/10.1109/TKDE.2011.163.
10. Catal C, Sevim U, Diri B. Software fault prediction of unlabeled program modules. In: Proceedings of the world congress on engineering WCE 2009; 2009. p. 1–6.
11. YunlongZhu ,Xiaohui Yan, Wenping Zou and Liang Wang, (2012), "A new approach for data clustering using hybrid artificial bee colony algorithm", Neuro computing, Vol. 97 , pp.241–250
12. Yi-Tung Kao, ErwieZahara and I-Wei Kao, (2008), "A hybridized approach to data clustering", Expert Systems with Applications, Vol.34, No. 3, pp.1754–1762
13. Waheeda Almayyan, Towards Predicting software defects with clustering techniques, International Journal of Artificial Intelligence and Application (IJAIA), Vol 12, No 1, January 2021
14. M. Shepperd, Q.Song, Z.Sun, and C.Mair, —Data quality: Some Eng., vol. 39, no. 9, pp. 1208–1215, 2013.
15. Z. Tóth, P. Gyimesi, and R. Ferenc, "A Public Bug Database of GitHub Projects and Its Application in Bug Prediction," in Computational Science and Its Applications -- ICCSA 2016, Cham, 2016, pp. 625-638: Springer International Publishing.
16. Simon Fong, Suash Deb, Xin-She Yang,and Yan Zhuang, Towards Enhancement of Performance of K-Means Clustering Using Nature-Inspired Optimization Algorithms, Computational Intelligence and Metaheuristic Algorithms with Applications, 2014, https://doi.org/10.1155/2014/564829
17. Deepinder Kaur, Arashdeep Kaur,"Fault Prediction using K-Canberra Means Clustering", CNC 2010[in Press]
18. Abiodun M. Ikotun, Mubarak S. Almutari and Absalom E. Ezugwu, K-Means-Based Nature-Inspired Metaheuristic Algorithms for Automatic Data Clustering Problems: Recent Advances and Future Directions, Appl. Sci. 2021, 11, 11246. https:// doi.org/10.3390/app112311246
19. Okwu, M.O.; Tartibu, L.K. Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications; Springer Nature: Berlin/Heidelberg, Germany, 2020; Volume 927.
20. Hruschka, E.; Campello, R.J.G.B.; Freitas, A.A.; de Carvalho, A. A Survey of Evolutionary Algorithms for Clustering. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 2009, 39, 133–155.
21. Nanda, S.J.; Panda, G. A survey on nature inspired metaheuristic algorithms for partitional clustering. Swarm Evol. Comput. 2014, 16, 1–18. [CrossRef]
22. Zhou, X.; Gu, J.; Shen, S.; Ma, H.; Miao, F.; Zhang, H.; Gong, H. An Automatic K-Means Clustering Algorithm of GPS Data Combining a Novel Niche Genetic Algorithm with Noise and Density. ISPRS Int. J. Geo-Inf. 2017, 6, 392
23. Mousa, A.A., El-Shorbagy, M.A. and Abd El-Wahed, W.F. (2012) Local Search Based Hybrid Particle Swarm Optimization for Multiobjective Optimization. Swarm and Evolutionary Computation, 3, 1-14.
24. N. Bouhmala, A. Viken, and J. B. Lønnum, Enhanced Genetic Algorithm with K-Means for the Clustering Problem, International Journal of Modeling and Optimization, Vol. 5, No. 2, April 2015
25. D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, New York, 1989 X.-S. Yang, "A new metaheuristic bat-inspired algorithm," *Nature Inspired Cooperative Strategies for Optimization*, vol. 284, pp. 65–74, 2010
26. Jianqiang Huang and Yan Ma, Bat Algorithm Based on an Integration Strategy and Gaussian Distribution, Volume 2020 | Article ID 9495281 | https://doi.org/10.1155/2020/9495281
27. Zheng, H., Zheng, Z., Xiang, Y., The application of ant colony system to image texture classification [textute read texture], In: Proceedings of the 2nd International Conference on Machine Learning and Cybernetics, Vol. 3, Xi'an, China, (2003) 1491-1495
28. R. Tang, S. Fong, X. Yang and S. Deb, "Integrating nature-inspired optimization algorithms to K-means clustering," *Seventh International Conference on Digital Information Management (ICDIM 2012)*, 2012, pp. 116-123, doi: 10.1109/ICDIM.2012.6360145.
29. Apostolopoulos, T. and Vlachos, A. (2011). Application of the Firefly Algorithm for Solving the Economic Emissions Load Dispatch Problem. International journal of Combinatorics. doi:10.1155/2011/523806
30. X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature & Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, india, December 2009.
31. Inacio G. Medeiros and Joao C. Xavier- Junior, Anne M. P. Canuto, Applying the Coral Reefs Optimization Algorithm to Clustering Problems, Conference Paper · July 2015 DOI: 10.1109/IJCNN.2015.7280845
32. Salcedo-Sanz, S. and Del Ser, J. and Gil-Lpez, S. and Landa-Torres, I. and Portilla-Figueras, J. A.," The Coral Reefs Optimization Algorithm: A Novel Metaheuristic for Efficiently Solving Optimization Problems," The Scientific World Journal, Volume 2014, Hindawi Publishing Corporation, 2014.
33. I. Koohi and V. Z. Groza, "Optimizing Particle Swarm Optimization algorithm," *2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering*

(CCECE), 2014, pp. 1-5, doi: 10.1109/CCECE.2014.6901057.
34. R.Jensi and G.WiselinJiji, HYBRID DATA CLUSTERING APPROACH USING K-MEANS AND FLOWER POLLINATION ALGORITHM, Advanced Computational Intelligence: An International Journal (ACII), Vol.2, No.2, April 2015
35. Agbaje, M.B.; Ezugwu, A.E.; Els, R. Automatic Data Clustering Using Hybrid Firefly Particle Swarm Optimization Algorithm.IEEE Access 2019, 7, 184963–184984.
36. Yang, X. S., and et al." Swarm Intelligence and Bio-Inspired Computation: Theory and Applications, Elsevier Science Publishers B. V. Amsterdam, The Netherlands,(2013)
37. S. Mirjalili, S. M. Mirjalili and A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw.* 69 (2014), 46–61.10.1016/j.advengsoft.2013.12.007
38. Hou, Y.; Gao, H.; Wang, Z.; Du, C. Improved Grey Wolf Optimization Algorithm and Application. Sensors 2022, *22*, 3810. https://doi.org/10.3390/s22103810
39. Jie-Sheng Wang, Shu-Xia Li, An Improved Grey Wolf Optimizer Based on Differential Evolution and Elimination Mechanism, *Sci*Rep 9, 7181 (2019). https://doi.org/10.1038/s41598-019-43546-3
40. Sandeep Rana, Sanjay Jasola , Rajesh Kumar, A hybrid sequential approach for data clustering using K-Means and particle swarm optimization algorithm, International Journal of Engineering, Science and Technology Vol. 2, No. 6, 2010, pp. 167-176