

International Journal of Business Intelligence and Data Mining

ISSN online: 1743-8195 - ISSN print: 1743-8187
<https://www.inderscience.com/ijbidm>

Building a hybridised meta-heuristic optimisation algorithm for efficient cluster analysis

D. Pradeep Kumar, B.J. Sowmya, Anita Kanavalli, Varun Cornelio, Jaison Pravith Dsouza, Wasim Memon, P. Prashanth

DOI: [10.1504/IJBIDM.2023.10050464](https://doi.org/10.1504/IJBIDM.2023.10050464)

Article History:

Received:	11 September 2021
Accepted:	23 November 2021
Published online:	30 November 2022

Building a hybridised meta-heuristic optimisation algorithm for efficient cluster analysis

D. Pradeep Kumar, B.J. Sowmya*,
Anita Kanavalli, Varun Cornelio,
Jaison Pravith Dsouza, Wasim Memon and
P. Prashanth

Department of Computer Science and Engineering,
M.S. Ramaiah Institute of Technology,
Bangalore, India

Email: pradeepkumard@msrit.edu

Email: sowmyabj@msrit.edu

Email: anithak@msrit.edu

Email: varuncornelio2000@gmail.com

Email: jaisonp.dsouza@gmail.com

Email: wmemon100@gmail.com

Email: prashanthp0703@gmail.com

*Corresponding author

Abstract: Nature-inspired algorithms are a relatively recent field of meta-heuristics introduced to optimise the process of clustering unlabelled data. In recent years, hybridisation of these algorithms has been pursued to combine the best of multiple algorithms for more efficient clustering and overcoming their drawbacks. In this paper, we discuss a novel hybridisation concept where we combine the exploration and exploitation processes of the vanilla bat and vanilla whale algorithm to develop a hybrid meta-heuristic algorithm. We test this algorithm against the existing vanilla meta-heuristic algorithms, including the vanilla bat and whale algorithm. These tests are performed on several single objective CEC functions to compare convergence speed to the minima coordinates. Additional tests are performed on several real-life and artificial clustering datasets to compare convergence speeds and clustering quality. Finally, we test the hybrid on real-world cases with unlabelled clustering data, namely a credit card fraud detection dataset, and a COVID-19 diagnosis dataset, and end with a discussion on the significance of the work, its limitations and future scope.

Keywords: nature-inspired algorithms; cluster analysis; vanilla whale; vanilla bat; optimisation algorithms.

Reference to this paper should be made as follows: Kumar, D.P., Sowmya, B.J., Kanavalli, A., Cornelio, V., Dsouza, J.P., Memon, W. and Prashanth, P. (2023) 'Building a hybridised meta-heuristic optimisation algorithm for efficient cluster analysis', *Int. J. Business Intelligence and Data Mining*, Vol. 22, Nos. 1/2, pp.170–222.

Biographical notes: D. Pradeep Kumar is an Assistant Professor in the Department of Computer Science and Engineering at M.S. Ramaiah Institute of Technology, Bangalore. His area of interest is data analytics, machine learning, and big data. He is the co-author of several papers published in international journals, conferences and book chapters.

B.J. Sowmya is an Assistant Professor in the Department of Computer Science and Engineering at M.S. Ramaiah Institute of Technology, Bangalore. Her area of interest is data analytics, machine learning, and internet of things. She is the co-author of several papers published in international journals, conferences, book chapters and a book.

Anita Kanavalli is working as the Head of Computer Science Department of Ramaiah Institute of Technology. Her areas of interest include ad hoc networks, high performance computing and microprocessors.

Varun Cornelio is studying in Computer Science and Engineering at M.S. Ramaiah Institute of Technology, Bangalore. His areas of interest are artificial intelligence, deep learning, machine learning and data science.

Jaison Pravith Dsouza is studying in Computer Science and Engineering at M.S. Ramaiah Institute of Technology, Bangalore. His areas of interest are internet of things and data science.

Wasim Memon is studying in Computer Science and Engineering at M.S. Ramaiah Institute of Technology, Bangalore. His areas of interest are web application security, OS security, blockchain and data science.

P. Prashanth is studying in Computer Science and Engineering at M.S. Ramaiah Institute of Technology, Bangalore. His areas of interest are web development and blockchain.

1 Introduction

In the present-day world, there is a lot of raw data, most of which is unlabelled. Data is being produced at a much faster pace than it can be analysed. One of the best to process such unlabelled data can be done by grouping the data based on the common features. Unsupervised learning is one of the best ways in ML, which helps us to analyse the unlabelled data. Clustering is one of the best ML algorithms which allow us to group unlabelled data. Even though the modern-day clustering algorithms are an efficient way to group the data, they need not be the best in class since they were defined a few years back. A recent trend is taking inspiration from nature and the environment to solve the optimisation problem of the clustering algorithms. A more advanced and optimal way of solving these clustering issues is to combine the existing nature-based algorithms (Agarwal and Mehta, 2014) to create a hybrid algorithm (Ezugwu, 2020) that better optimises the model. In this project, we look into implementing a hybridised algorithm of two or more nature-based algorithms, which can be used as an optimiser for the clustering model to improve the accuracy.

Clustering techniques are generally classified into three main classes: partitional, overlapping and hierarchical. The latter two are linked in which hierarchical clustering is a nested classification of partition clustering. Therefore, they present poor performance when the separation of overlapping clusters is conducted. Optimisation is required to have better accuracy in clustering. Hence in this paper, we propose and design a hybrid

nature-based heuristic algorithm inspired by the bat optimisation algorithm (Yang and He, 2013) and whale optimisation algorithm (WOA) (Mirjalili and Lewis, 2016). We infer this novel algorithm on several CEC objective functions, a set of artificial and real-world labelled clustering datasets, and also on a few unlabelled real-world datasets, to compare its performance against its corresponding vanilla versions (Yang and He, 2013; Mirjalili and Lewis, 2016), and also against other nature-inspired, meta-heuristic algorithms (Oduntan and Thulasiraman, 2018).

Modern-day problems require quick and accurate solutions, and in certain scenarios, are also constrained with respect to resource utilisation. Hence, providing an accurate solution in lesser time and budget constraints become a necessity. The hybrid proposed in this work aims to achieve the same by converging the result faster, fewer iterations, and more accurately than normal meta-heuristic algorithms.

2 Literature survey

Cluster analysis is a crucial tool in the mining of data. Clustering is the process of organising objects into groups whose constituents have similar features and are unique to the data points in other groups. Several clustering algorithms introduced have been pivotal to the success of this method. However, traditional clustering algorithms depend on preliminary data such as the number of clusters and may struggle to deal with obstacles where the number of clusters is unknown. This lack of information results in expanding computational burdens. However, these simulate various types of real-world problems where the number of clusters in data cannot be easily distinguished. The most commonly used clustering algorithm is the k-means, used to solve many clustering problems. Heuristic techniques are applied to compute global optimal solutions by identifying all possible centroid positions. Nature-inspired algorithms have been successfully applied to optimise a wide variety of numerical optimisation problems in the last few years. When joined with clustering algorithms such as K-means, the centroid for clusters is determined iteratively.

Nature-inspired algorithms have proved to be highly efficient in solving NP-hard and NP-complete problems. Agarwal and Mehta (2014) helped decide the type of algorithms that would aid in exploring the local area to defeat the issue of ‘curse of dimensionality (COD). An extensive audit of the 12 calculations, in particular, genetic algorithm (GA), ant colony optimisation (ACO), particle swarm optimisation (PSO), memetic algorithm (MA), bacterial foraging enhancement calculation (BFOA), shuffled frog leaping algorithm (SFLA), artificial bee colony (ABC) algorithm, firefly algorithm (FFA), biogeography-based optimisation (BBO), cuckoo search algorithm (CSA), bat algorithm (BA) and flower pollination algorithm (FPA). The principal focal point was to illuminate the exploration of the local area with the streamlining ability of current algorithms over multi-modular and unimodal functions for optimisation on a global scale.

In 2020, Ezugwu et al. presented an overview and analysis of the trends and development in nature-inspired meta-heuristic clustering methods. A brief review of clustering techniques such as partitional and hierarchical clustering is made, followed by ideas like proximity measures. A meticulous examination of the algorithms using different methods is undertaken.

Further, they present major meta-heuristic algorithms that have been used for the same in recent years. The authors also suggest three hybrid swarm intelligence and evolutionary algorithms: particle swarm differential evolution algorithm, firefly differential evolution algorithm, and invasive weed optimisation differential evolution (IWO) algorithm. These are suggested to deal with the job of automated data clustering. The principal reason for the success of such meta-heuristic algorithms is that there is no necessity for predetermined data or preliminary information of the dataset to be classified.

Results show that the FFA was more suitable for low and high-dimensional data clustering than any modern algorithm. Moreover, the paper shows the advantage of the three proposed hybrid algorithms over the standard state-of-the-art methods. Data clustering has a broad application because it uses relevant information that could be hidden within groups. This has been used in several fields like engineering, computer science, medical science, Earth science, life science, economics and bioinformatics.

Test results for the hybrid FFA display better performance results concerning computation cost, with an enhanced convergence speed, while the hybrid particle swarm optimisation differential evolution (PSODE) and IWO) defeated their single algorithms variants by producing better quality clustering solutions. The performance of eight meta-heuristic algorithms and FA proved to be superior.

Agarwal and Mehta (2015) examined a comparative study of three nature-inspired clustering algorithms, i.e., bat clustering, firefly clustering, and flower pollination clustering algorithms, to discuss the problem of obtaining the most optimal clusters in datasets. Clustering is the most effective unsupervised learning technique that deals with fixing a structure in a mysterious dataset. K-means clustering is a widely accepted clustering algorithm that gathers similar data objects instantly. The convergence speed of K-means clustering is quite palpable, but it can get stuck into local optima. The nature-inspired algorithm, when combined with clustering algorithms, provides a globally optimal solution. The comparison research on four real-life datasets collected from the UCI machine learning repository and two simulated datasets. Algorithms are tested based on several fitness functions and CPU time per run.

The selection of clusters is an integral part of the optimisation problem. José-García and Gómez-Flores (2016) discussed several meta-heuristics that perform automatic clustering to select the most optimal number of clusters for an optimisation problem. They also discuss the characteristics of the meta-heuristics regarding automatic clustering like encoding schemes, validity indices, and proximity measures. These approaches are reviewed on a single objective and multi-objective optimisation problems, finally discussed with research directions. The authors discuss the significant meta-heuristics in physics and evolution-based fields. Several swarm intelligence-based techniques like PSO, its derivatives such as DCPSO, MEPSO, and other algorithms such as ACO, invasive weed optimisation, bee colony, artificial immune systems, etc., are researched. For multi-objective optimisation problems, they discuss powerful algorithms such as PESA II, MOCK, MOKGA, etc. They also discuss differential evolution techniques for optimisation, such as MODE and DEMO, finally cover multi-objective swarm optimisation methods such as ADCMC, MONOCLONAL, AMOSA, and others. The authors discuss how swarm-based and artificial immune system-based algorithms

play a significant role in recent research, including hybrid approaches of existing meta-heuristics. They used distinct datasets in different scenarios to test the algorithms. They mentioned that if the clustering hassle is linearly separable, a clustering technique primarily-based totally on unmarried goal meta-heuristics can be enough to reap ultimate clustering solutions. On the other hand, if the clustering hassle is nonlinearly separable, a multi-goal clustering technique is suggested because, in general, they try to optimise the compactness and the connectedness of clusters simultaneously.

An up-to-date review of all major nature-inspired algorithms for partitional clustering is provided by Nanda and Panda (2014), also discussing critical issues in the field of meta-heuristics in clustering and application areas. Apart from physics-based and evolution-based algorithms, swarm intelligence is employed when the clustering problem is inspired by the collective intelligence of social animals, such as bees, ants, dolphins, whales, bats, etc. A few notable works in this field are the ant colony algorithm, which uses the foraging behaviour of the ants, particle swarm algorithm, cat swarm optimisation, cuckoo search, BA, and many more. The authors also discuss multi-objective algorithms, which closely resemble many real-world optimisation problems, where algorithms such as MOCK and NGSA-II have been proposed. Real-world application of the algorithms is made in several fields like character recognition, travelling salesman problem, blind channel equaliser design, human action classification, etc. The authors assert that meta-heuristic algorithms trump the traditional gradient descent approach by obtaining global minima.

A new aim in clustering optimisation is to connect existing meta-heuristic algorithms to subdue their shortcomings. Oduntan and Thulasiraman (2018) proposed the hybridisation of the ant brooding sorting and the tabu search algorithms to reveal an optimal solution. These algorithms are joined in a collaborative and integrative manner to build two variants of the algorithm. They mix the positive traits from both algorithms to produce a single hybrid clustering algorithm which includes checking the component tactics of ant brooding and the tabu search. The final purpose is to recognise similar techniques and choose the ones that contribute to the clustering goal. The hybrids are applied to a synthetic dataset to learn its underlying behaviours. It is noted that it is usually easier to build a complex version of the hybrid rather than a simple one. But another conclusion was that any major defect of the starting algorithms also exists in the hybrid version.

In another study, Ezugwu (2020) has presented an up-to-date review on powerful nature-inspired meta-heuristic algorithms, including PSO, FA, IWO, GA, and differential algorithm (DE), to solve automatic clustering problems. Comparative research is done on several modified well-known global meta-heuristic algorithms. A proposal is made for three hybrid meta-heuristic algorithms for resolving the automated data clustering task, namely, the PSODE, firefly algorithm differential evolution (FADE), and the IWODE. Forty-one benchmarked datasets that include 11 synthetic, and 30 real-world datasets are used to assess the performances of these nature-inspired clustering algorithms. Further, an empirical study proves the superiority of the three proposed hybrid algorithms over the standard state-of-the-art methods in obtaining essential clustering solutions to the problem at hand. In particular, the FA and its hybrid equivalent, the FADE algorithm, show better solution accuracy and attain higher levels of stabilities than the other compared algorithms.

Along with their previous study, Mehta and Agarwal (2018) proposed a novel hybrid algorithm (ABC_DE_FP) produced by integrating FPA and DE in the original ABC

algorithm. Notwithstanding having good efficiency and more straightforward implementation, ABC suffers from a few disadvantages, such as getting caught in a local minimum. Several altered and hybridised versions of the algorithm were proposed. Still, the ABC_DE_FP outperformed them in terms of minimum error value achieved and convergence speed by maintaining stabler synchronisation between exploration and exploitation. To test the novel hybrid algorithm's feasibility, it is principally compared with up-to-date ABC variants such as GABC, IABC, and AABC over simple benchmark problems. After that, it is evaluated concerning original ABC, FPA, hybrid ABC_FP, ABC_DE, and ABC_SN over CEC2014 optimisation problems up to 100 dimensions.

The issue of NP-hard and NP-complete is shown by six original articles by Soto et al. (2020), which concentrate on theoretical and pragmatic aspects of neural networks, neural models, brain-computer interface, machine learning and optimisation algorithms.

The initial article is named 'Double-criteria active learning for multiclass brain-computer interfaces' and concentrates on enhancing the data collection process for improving these systems. Usually, these systems are created by using electroencephalography (EEG) signal datasets. A proposal for connecting a two-query active learning algorithm with an extreme learning machine (ELM) to resolve this problem is made. The recommended approach even reveals that hybrid outperforms several state-of-the-art methods.

In 2020, Tzanetos and Dounias (2020) discussed how evolutionary-based techniques had been developed, which are inspired by flora and fauna in nature to solve optimisation problems. But many of these proposed strategies have no such inspiration from nature and show no real-world utility due to lack of real-world applications. Hence, the authors focus on surveying the existing algorithms, which are swarm-based and derived from these algorithms, and display their preliminary findings. Several swarm intelligence-based algorithms have been introduced to solve such problems, among others, like the ABC, bacterial foraging, whale optimisation, grey wolf optimisation, and tons more, which are modelled based on the social behaviour of these animals when it comes to movement and hunting.

Eesa et al. (2013), showcased a new meta-heuristic optimisation algorithm motivated by the mechanism of the colour-changing form of cuttlefish to find the optimal solution in numerical optimisation problems. The designs and colours seen in cuttlefish are created by reflected light from different layers of cells heaped together. The sequence of specific cells at once allows cuttlefish to possess an extensive collection of patterns and colours. The suggested algorithm simulates the light reflection process by combining these layers and the visibility of the matching pattern process used by cuttlefish to match its environment.

The algorithm divides the population (cells) into four groups; each group works individually, sharing only the most suitable solution. Two of them were used as a global search, while others were used as a local search.

In recent years, a lot of different meta-heuristics have been developed. Even the cuttlefish algorithm (CFA) is motivated by changing colour behaviour to find the optimal solution. The results achieved by the proposed CFA in all cases produce excellent outcomes when compared with GA, PSO, and BA. For future work, more study on CFA parameters is required.

In 2017, Singh and Salgotra (2017) depicted a new modification of the FPA, i.e., enhanced flower pollination algorithm (EFPA), which has been suggested for the pattern

integration of non-uniform linear antenna arrays (LAA). However, the proposed algorithm implementing; however utilises the theory of Cauchy mutation in global pollination and developed local search to improve the exploration and exploitation trends of FPA.

Antenna arrays find their purpose in several wireless uses such as radar, sonar, mobile, TV and satellite. Antenna arrays design is an elaborate and nonlinear problem. Hence, several optimisation methods such as GA, DE, PSO, BBO, and many others have been used to incorporate these. LAA consist of several antenna components arranged in a straight line. LAA has become very popular because of its simple geometry and applications. LAA design has been reviewed by many researchers using several optimisation algorithms in the past.

In the recent past, many researchers have concentrated on enhancing the basic abilities of FPA. The algorithm, due to its linear character, makes it fitting for deeper investigation. But it has been proved that the FPA algorithm has little feasibility to optimise problems at hand. Additionally, the performance of FPA has not been investigated to a more intricate level, and the algorithm still has to show, through results, its effectiveness for becoming a feasible algorithm. Considering the above review, a new version of FPA, namely EFPA, has been suggested. Three modifications in the basic FPA have been introduced in the enhanced version. The changes in EFPA help in providing more helpful searchability and allow it to escape local minima. The proposed algorithm has been examined on seven benchmark functions. The results determine that EFPA can find the global optima of most of the benchmark functions. Furthermore, the EFPA is employed for pattern construction of non-uniform LAA.

Dhiman and Kumar (2017) presented a novel swarm-based meta-heuristic algorithm named spotted hyena optimiser (SHO) motivated by the behaviour of spotted hyenas. The leading theory behind this algorithm is the social connection between spotted hyenas and their collaborative behaviour. The three fundamental steps of SHO are seeking prey, encircling, and attacking the target, and all three are mathematically formed and executed. The proposed algorithm is compared with eight recently developed meta-heuristic algorithms on 29 benchmark test functions. The results illustrate that the SHO provides remarkably competing results compared to other heuristics such as GWO, PSO, MFO, MVO, SCA, GSA, GA and HS. The analytical results, which are based on the comparisons of the proposed SHO against other optimisation techniques, show that the suggested method can handle different types of restrictions and offer better resolutions than other optimisers.

In 2020, Martínez-Álvarez et al. suggested a novel bio-inspired meta-heuristic mimicking how the coronavirus disperses and infects healthy people. The virus swiftly spreads itself from patient zero (The first infected patient). The infected community initially grows exponentially over time, but taking into factor social isolation measures, the mortality rate, and the number of recoveries, the infected population slowly declines.

The coronavirus optimisation algorithm has two significant benefits when compared with other comparable strategies. First, the input parameters are already set according to the disease statistics, blocking researchers from initialising them with arbitrary values. Second, the system can end after many iterations without fixing this value either. Moreover, a parallel multi-virus version is introduced, where several coronavirus strains develop over time and explore more comprehensive search space areas in fewer iterations. The meta-heuristic has been coupled with deep learning models to find optimal hyperparameters during the training stage.

The algorithm has shown to be exceptionally successful in applications such as time series forecasting. Names of the variables such as (P DIE), i.e., the probability of death from the virus, (P SUPER SPREADER), (SPREADING RATE) are relatively self-explanatory.

CVOA has three significant advantages. First, its high relation to the coronavirus spreading model restricts users from judging the input values. Second, it stops after a certain number of iterations due to the transfer of individuals among healthy and dead/recovered lists.

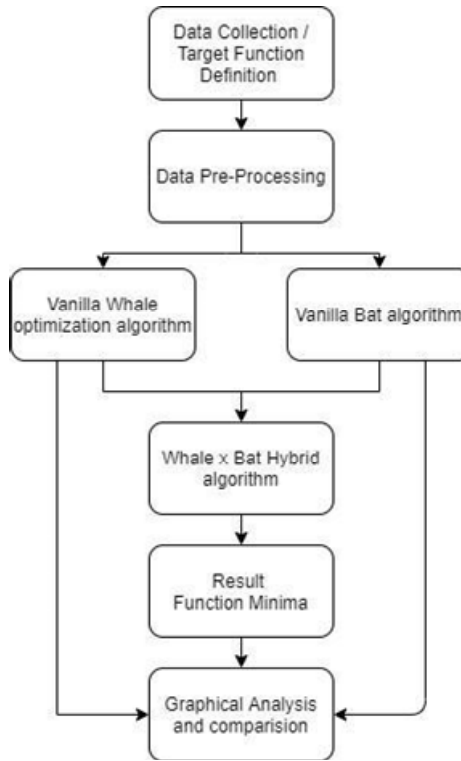
Nature-inspired algorithms have proved to be extremely efficient in solving NP-hard and NP-complete problems. But they have their limitations. It is seen that vanilla algorithms get stuck at local optima, have slower convergence, or both. Hybridisation of these algorithms can give much faster convergence rates, with a reduced probability of fixating on local minima.

3 Design and implementation

Initial collection of data and objective functions is done, followed by preprocessing the data to eliminate any missing values, outliers and corrupt data entries and non-informative features. The resulting dataset is normalised and given input to the vanilla algorithms with an evaluation function for minimising centroid distance. Alternatively, the target objective function is supplied to the algorithm to be minimised. A hybrid is developed from the vanilla algorithms, and the same dataset/objective functions are supplied to the hybrid. The results are collected, and graphical comparison is performed.

The workflow mentioned in Figure 2 explains the flow of the hybrid algorithm more descriptively and how the vanilla algorithms are integrated to create the hybrid. The input to the algorithm is either the unlabelled data to be clustered or an objective function to be minimised. The hyperparameters and other variables for the hybrid, such as population size, number of generations, loudness, frequency, rate, spiral parameter, etc., are initialised. In the case of clustering data, we preprocess the input to eliminate corrupted entries, missing features, and redundant columns, normalise the data and convert it into a NumPy array for easier processing. In the case of an objective function, we define the function to be minimised and the boundary limits, i.e., the search space of the hybrid algorithm.

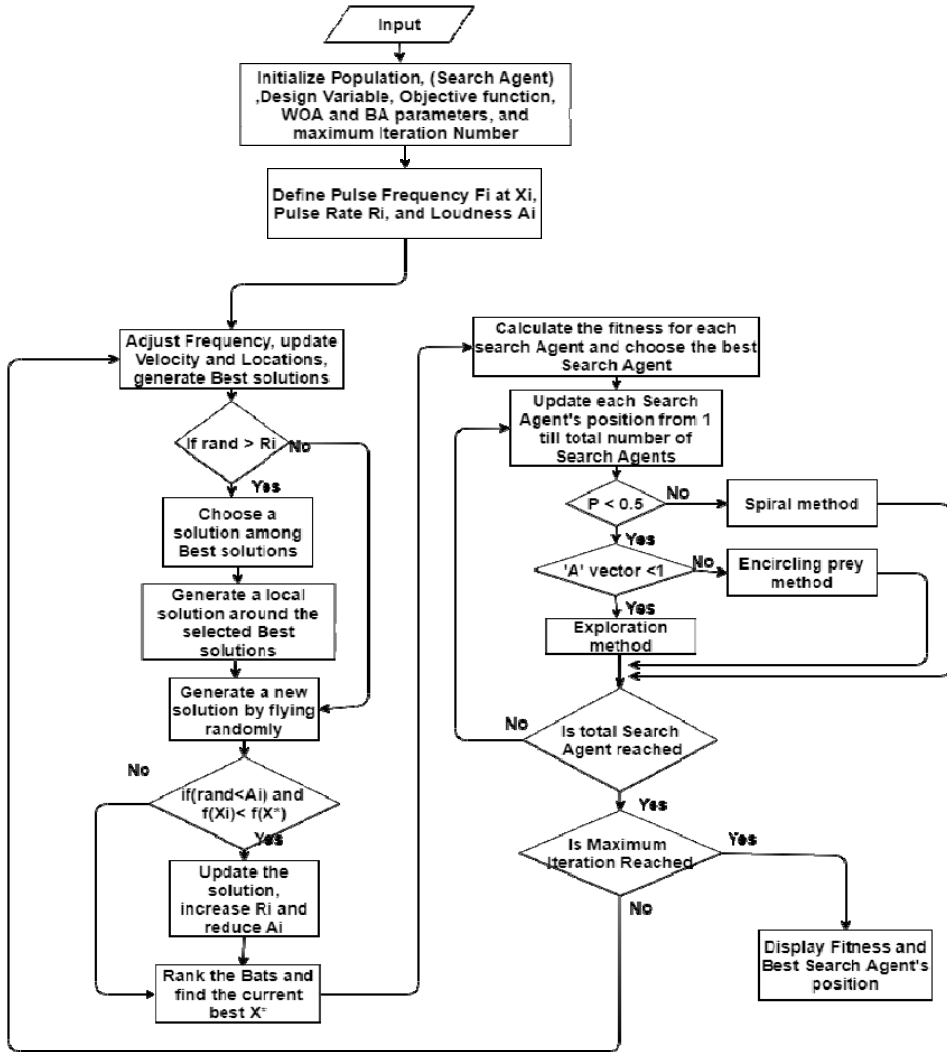
Figure 1 Proposed architecture



The initial section of the hybrid is the bat exploration/exploitation section. The bat conducts its exploration and exploitation, as mentioned before. The best indices are obtained from the local search using equation (4), and the frequency, velocity, and position of the rest of the swarm members can be updated using equations (1), (2) and (3). Similarly, the rate and loudness are also updated using equations (5) and (6).

The best swarm members, hence obtained from the bat exploration/exploitation, are used to update the leader positions of the swarm for the exploration and exploitation process of the whale optimisation section of the hybrid. With the best swarm members defined, the rest of the swarm now updates its position towards the best member encircling the prey using equation (7). A spiral equation is then calculated to mimic the helical structure adopted by whales, to exploit the location. This is described in equation (9). The exploration phase of the whale optimisation is conducted similarly, with the position vector being randomised to cover the entire search space, as described in equation (8). Hence, the leader positions of the swarm are obtained as the result of one generation of the hybrid. This process is repeated for a certain number of generations to obtain the best leader positions of the swarm for the hybrid algorithm. Suppose the input data is unlabelled clustering data. In that case, these leader positions specify the centroid locations for each feature of the unlabelled data. In the case of the input being an objective function to be minimised, these leader positions specify the location of the minima and the minima obtained for the objective function.

Figure 2 Workflow of the system



3.1 Vanilla BA

The idea of bat optimisation is based on the natural phenomenon of how bats use echolocation to sense distance from prey. The bats fly arbitrarily with velocity V_i at position X_i with a frequency F and loudness A to find its prey. They automatically adjust the pulse frequency and adjust the rate R , depending on the closeness of the target. There is a current best solution X among the whole swarm, to which the rest of the swarm updates its positions. Hence, the frequency of the bats is updated using the equation.

$$f_i = f_{min} + (f_{max} - f_{min}) * \beta \tag{1}$$

where f_{\max} and f_{\min} are the predefined maximum and minimum bounds of frequency, β is a random number uniformly generated from the interval $[0, 1]$. The velocity of the bats is updated using the equation.

$$v_{it} + 1 = v_{it} + (x_{it} - x^*) * f_i \quad (2)$$

and finally, the location of the bats is updated using the equation

$$x_{it} + 1 = x_{it} + v_{it} + 1 \quad (3)$$

where $x_{it} + 1$ and $v_{it} + 1$ are the position and velocity of the bat i in generation t . Here, x^* contains the best indices for the bats among the swarm. For local exploitation of best indices, the following equation is used:

$$x_{it} + 1 = x^* + \varepsilon * A' \quad (4)$$

where ε is a random number uniformly generated from the interval $[1, 1]$, and AO is the average loudness of all bats. The loudness A_i and the rate r_i of pulse emission are updated similarly as the iterations proceed. Loudness decreases once a bat has found the prey, and the rate increases. The equations for the same are given below.

$$r_{it} + 1 = r_{i0} * (1 - e^{-\gamma t}) \quad (5)$$

$$A_{it} + 1 = \sigma * A_{it} \quad (6)$$

where $A_{it} + 1$ is the loudness and emission rate is $r_{it} + 1$.

This process is repeated for a certain number of iterations, and at the end, the best indices among the bats are returned as a result.

3.2 *Vanilla whale algorithm*

Humpback whales have a unique hunting method. This method of foraging is called the bubble-net feeding method. There exist two methods associated with bubbles called ‘upward-spirals’ and ‘double-loops’. In the former manoeuvre, the whales create bubbles in a spiral shape around the prey and swim up toward the surface. The second manoeuvre includes three parts, coral loop, lobe tail and capture loop.

Encircling of prey is done using the following equation where t indicates the current iteration, C is the coefficient vector, X^* is the position vector of the best solution obtained so far, X is the position vector. Here, the WOA algorithm assumes that the current best candidate solution is the prey or is close to the same. After the best whale is found, the other whales will try to update their positions towards the best whale.

A similar search method is used for the exploration phase, where the whale randomly searches the space based on their position to each other. A is used for defining whether the whales explore or exploit. If $A > 1$, then it defines the exploration phase and is given by the equation.

$$X_{(t+1)} = X_{\text{rand}} - A * D \quad (7)$$

where X_{rand} is a random position vector.

In the spiral updating part, first find the distance between the whale located at (X, Y) and the prey. We then create a spiral equation between the position of whales and prey to mimic the helix-shaped movement of whales as given below.

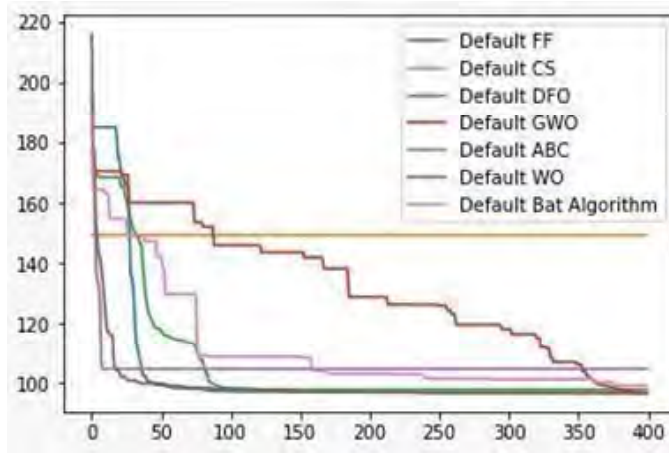
$$X_{(t+1)} = D' * e^{bl} * \cos(2\pi l) + X^*(t) \tag{8}$$

Where $D = |X(t) - X^*(t)|$ and indicates the distance of the t^{th} whale to the prey that is the best solution obtained till then, b is a constant for setting the logarithmic spiral's shape, l is a random number in $[-1, 1]$.

3.3 Selection of optimisers for hybrid

To find appropriate optimisers for our hybrid, we go through a few prominent nature-inspired algorithms and test them on some standard datasets to find the best fit for our hybrid. The results of the testing on the IRIS dataset are shown in Figure 3.

Figure 3 Convergence graph of seven optimisers on IRIS dataset (see online version for colours)



Comparing the convergence of all optimisers, we find that the best performing algorithms are the FFA and the dispersive fly algorithm. However, these algorithms take very long to converge and train, which can be an issue while evaluating complex target functions. Hence, we consider the next best in this group, namely the WOA and the BA.

Since the IRIS dataset has three clusters, we will get three centroids that represent the clusters. Thus, we get 12 variables that characterise four features for each centroid. The last row depicts the minimum of the cumulative sum of distances of the data points to its corresponding centroid obtained by the seven optimisers. After reviewing individual nature-based algorithms, we conclude that the whale and BA works better than other algorithms with faster and more accurate minima, according to Table 1.

Concerning implementation, the hybrid was developed as a process of trial and error, which included experimenting with different types of integration between the vanilla algorithms. Initial experimentation of hybridising was performed by parallelly integrating the default algorithms by comparing individual bat and whale members and retaining the better among the 2. Although the integration was successful, the results were below average. More research was conducted in this field, which led to the eventual development of the present hybrid, which integrates the two default algorithms serially by executing the exploration/exploitation of the BA initially. The best indices obtained as a result are used to update the leader positions of the whale swarm, hence effectively

making the process of whale exploration more optimised as it now searches for the minima in a better-defined search space. Finally, the leader positions from the whale algorithm are returned as the centroid positions of the detected clusters, or it returns the minima of the specified target function in the case of target objective functions.

Figure 4 Algorithmic description

Algorithm 1 BA_WOA Hybrid Algorithm

```

1:  Initialise Hyper parameters position, velocity, loudness  $At$ , rate  $Rt$  and other param- eters
2:   $max\_iter =$  Maximum number of iterations
3:   $rand =$  random number between 0 and 1
4:  Initialize the swarm population  $X_i$  ( $i = 1, 2, \dots, n$ )
5:  while  $t < max\_iter$  do
6:      for each search agent do
7:          Randomly generate Frequency for each swarm individual with Eq. 1
8:          Update Velocity using equation (2)
9:          Update Position using equation (3)
10:         if  $rand > Rt$  then
11:             Update position using equation (4)
12:             Calculate the fitness
13:             if ( $rand < At$ ) && ( $f(x_t) < f(x^*)$ ) then
14:                 Replace the position with the new one
15:                 Update  $Rt$  and  $At$  using equation (5) and equation (6)
16:                 Select the current global best position
17:             Calculate the fitness of each search agent using best position
18:              $X^* =$  the best search agent
19:         for each search agent do
20:             Update  $a$ ,  $A$ ,  $C$ ,  $l$ , and  $p$ 
21:             if  $p < 0.5$  then
22:                 if  $\|A\| < 1$  then
23:                     Update the position of the current search agent by equation (7)
24:                 else
25:                     if  $\|A\| > 1$  then
26:                         Select a random search agent ( $X_{rand}$ )
27:                         Update the position of the current search agent by the equation (8)
28:                     else
29:                         if  $p \geq 0.5$  then
30:                             Update the position of the current search by equation (9)
31:                 Check if any search agent goes beyond the search space and amend it
32:             Calculate the fitness of each search agent
33:             Update  $X^*$  if there is a better solution
34:              $t = t + 1$ 
35:  Return  $X^*$ 

```

Table 1 Variable and minima values for the algorithms

<i>Val/Algo</i>	<i>Whale optimisation</i>	<i>Firefly algorithm</i>	<i>Gray Wolf optimisation</i>	<i>Dispersive fly algorithm</i>	<i>Bat algorithm</i>	<i>Artificial bee colony</i>	<i>Cuckoo search</i>
Variable 0	5.937555	6.756563	6.679055	5.934326	4.970297	6.056295	5.557552
Variable 1	2.798314	3.141794	3.027999	2.797795	3.423471	2.920790	3.719123
Variable 2	4.419815	5.570584	5.519128	4.417893	1.479011	4.167382	1.699666
Variable 3	1.423832	2.100884	2.022472	1.417253	0.262172	1.383642	0.513712
Variable 4	6.732514	5.051060	5.014842	5.001879	6.894647	4.885129	6.001694
Variable 5	3.066595	3.400815	3.402012	3.391111	3.160739	3.319504	2.742119
Variable 6	5.630325	1.463851	1.480659	1.468645	5.769044	1.348613	5.330351
Variable 7	2.102004	0.234059	0.310977	0.100000	2.175160	0.100000	1.497156
Variable 8	5.012007	5.926498	5.827516	6.733348	5.822522	6.452000	4.586730
Variable 9	3.400124	2.791996	2.771119	3.067850	2.796140	2.932192	2.937747
Variable 10	1.471578	4.431637	4.307615	5.630075	4.299324	5.318797	3.830607
Variable 11	0.215677	1.401602	1.334323	2.106808	1.521637	1.783736	1.265619
Minimum	96.684575	97.025503	97.481687	97.900166	99.297205	104.760097	149.260500

The algorithmic description of the hybrid whale \times BA is described in Figure 4. This algorithm provides the descriptive working of the hybrid and explains the integration of the default algorithms into the hybrid and its working.

Tables 2–8 describe the 14 functions being used to test the hybrid algorithm against its default counterparts. These functions are single objective CEC benchmark functions taken across various years to test the algorithm's effectiveness in finding the location and value of the function's global minimas. Each table/column consists of the function formulae, the global minima and its position, and a visual representation of the function in a 3D plane.

Table 2 Beale function and Easom function (see online version for colours)

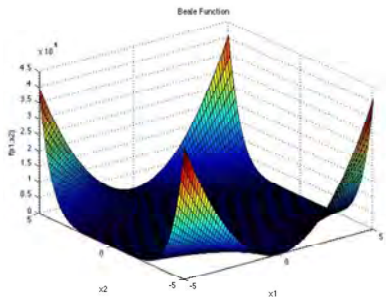
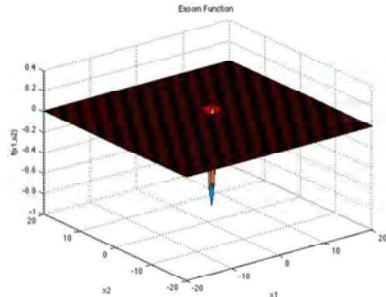
	<i>Beale function</i>	<i>Easom function</i>
Function	$f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$	$f(x) = -\cos(x_1)\cos(x_2)\exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$
Minimum	$f(x) = 0$ at $x \in [3, 0.5]$	$f(x) = -1$ at $x \in [\pi, \pi]$
Visualisation		

Table 3 Six-hump camel-back function, eggholder function and holder table function (see online version for colours)

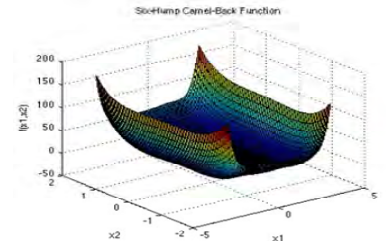
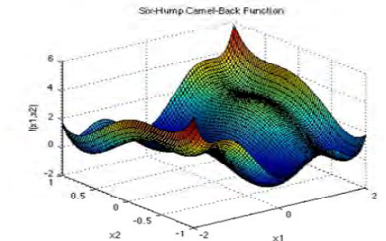
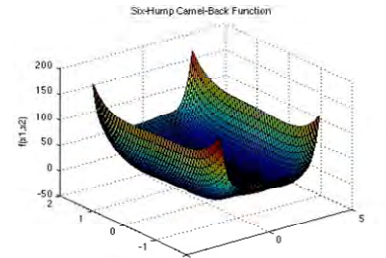
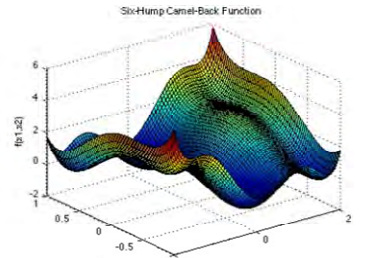
	<i>Six-hump camel-back function</i>	
Function	$f(x) = (4 - 2.1x_1^2 + x_1^4 / 3)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$	
Minimum	$f(x) = -1.0316$ at $x \in [0.0898, -0.7126], [-0.0898, 0.7126]$	
Visualisation		
		

Table 3 Six-hump camel-back function, eggholder function and holder table function (continued) (see online version for colours)

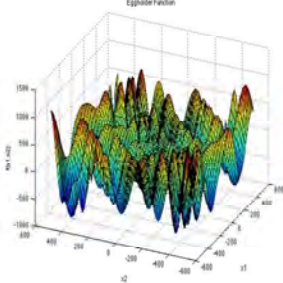
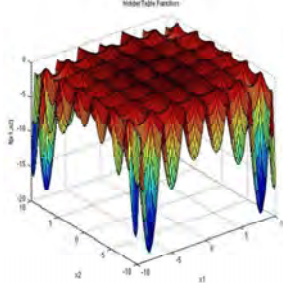
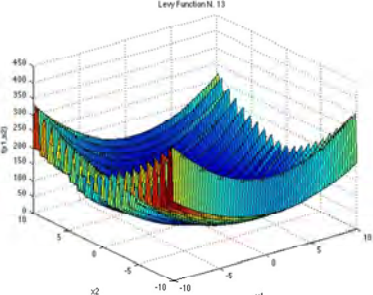
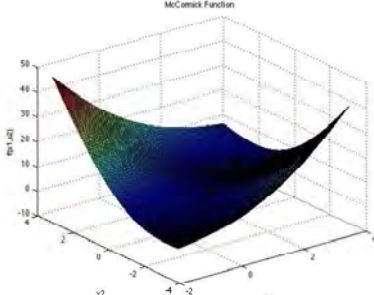
	<i>Eggholder function</i>	<i>Holder table function</i>
Function	$f(x) = -(x_2 + 47)\sin(x_2 + x_1 / 2 + 47) - x_1((x_1 - (x_2 + 47)))$	$f(x) = - \sin(x_1)\cos(x_2)\exp(1 - (x_1^2 + x_2^2) / \pi) $
Minimum	$f(x) = -959.64$ at $x \in [512, 404.2319]$	$f(x) = -19.2$ at $x \in [8.05, 9.66], [8.05, -9.66], [8.05, 9.66], [-8.05, -9.66]$
Visualisation		

Table 4 Levy N. 13 function and McCormick function (see online version for colours)

	<i>Levy N. 13 function</i>	<i>McCormick function</i>
Function	$f(x) = \sin^2(3\pi x_1) + (x_1 - 1)^2[1 + \sin^2(3\pi x_2)] + (x_2 - 1)^2[1 + \sin^2(2\pi x_2)]$	$f(x) = \sin(x_1 + x_2) + (x_1 - x_2)^2 - 1.5x_1 + 2.5x_2 + 1$
Minimum	$f(x) = 0$ at $x \in [1, 1]$	$f(x) = -1.913$ at $x \in [-0.55, -1.55]$
Visualisation		

Tables 2–8 help understand the functions to be minimised. It consists of both simple and complex functions, including a few with many local minimas in which a naive algorithm can get stuck in. They help us to better understand the advantage of the hybrid algorithm against a default meta-heuristic algorithm.

Table 5 Rosenbrock function and MICHALEWICZ function (see online version for colours)

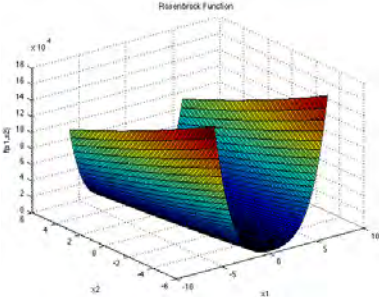
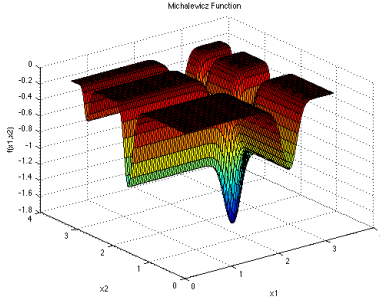
	<i>Rosenbrock function</i>	<i>Michalewicz function</i>
Function	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$f(x) = -\sum_{i=1}^n \sin(x_i) \sin^{2m}(ix_1^2 / \pi)$
Minimum	$f(x) = 0$ at $x \in [1, 1]$	$f(x) = -1.8$ at $x \in [2.2, 1.57]$
Visualisation	 <p>Rosenbrock Function</p>	 <p>Michalewicz Function</p>

Table 6 Schwefel function and Styblinski-tang function (see online version for colours)

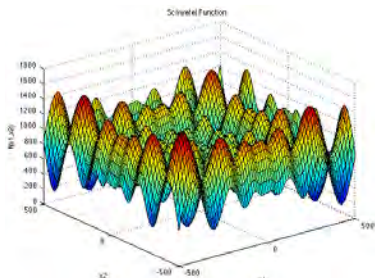
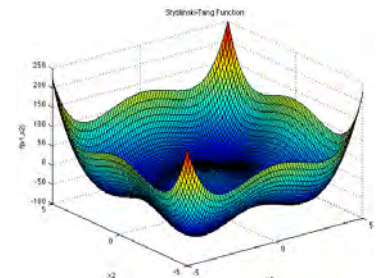
	<i>Schwefel function</i>	<i>Styblinski-tang function</i>
Function	$f(x) = 418.9829d - \sum_{i=1}^n x_i \sin(x_i)$	$f(x) = 12 \sum_{i=1}^n (x_i^4 - 16x_i^2 + 5x_i)$
Minimum	$f(x) = 0$ at $x \in [420.96, 420.96]$	$f(x) = -39.16$ at $x \in [-2.9, -2.9]$
Visualisation	 <p>Schwefel Function</p>	 <p>Styblinski-Tang Function</p>

Table 7 Cross-in-tray function and Bukin N. 6 function (see online version for colours)

	<i>Cross-in-tray function</i>	<i>Bukin N. 6 function</i>
Function	$f(x) = -0.0001 \sin(x_2) \exp(10 - x_{12} + x_{22}) + 10.1$	$f(x) = 100x_2 - 0.01x_{12} + 0.01x_1 + 10$
Minimum	$f(x) = -2.06$ at $x \in [1.35, 1.35], [1.35, -1.35], [-1.35, 1.35], [-1.35, -1.35]$	$f(x) = 0$ at $x \in [-10, 1]$

Table 7 Cross-in-tray function and Bukin N. 6 function (continued) (see online version for colours)

	Cross-in-tray function	Bukin N. 6 function
Visualisation		

Table 8 Shubert function

Function	$f(x) = \sum_{i=1}^{25} 15 \cos((i + 1)x_1 + i) + \sum_{i=1}^{25} 15 \cos((i + 1)x_2 + i)$	
Minimum	$f(x) = -186.7$ at $x \in [-10, 10]$	
Visualisation		

4 Result and performance analysis

The following results were collected after extensively testing the hybrid against other meta-heuristic algorithms, on 14 CEC functions and 20 real-life and artificial datasets, and finally on two real-world scenarios, namely a credit card fraud detection dataset and

a COVID-19 diagnosis dataset. The below snapshots are divided into three sections for the same.

4.1 CEC functions

The results for each objective CEC function have been recorded in a box graph and convergence graph. The blue, yellow, and green boxes/lines represent the WOA, BA and the hybrid algorithm.

4.1.1 Beale function

Figure 5 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over Beale function. For this function, the actual minima are the same as the expected value, which is depicted by lower whiskers of the box plot. It indicates that hybrid eventually performs better than WOA and BA in the Beale function since it has a minimum value in both box plot and convergence graph.

Figure 5 Beale function box and convergence plot (see online version for colours)

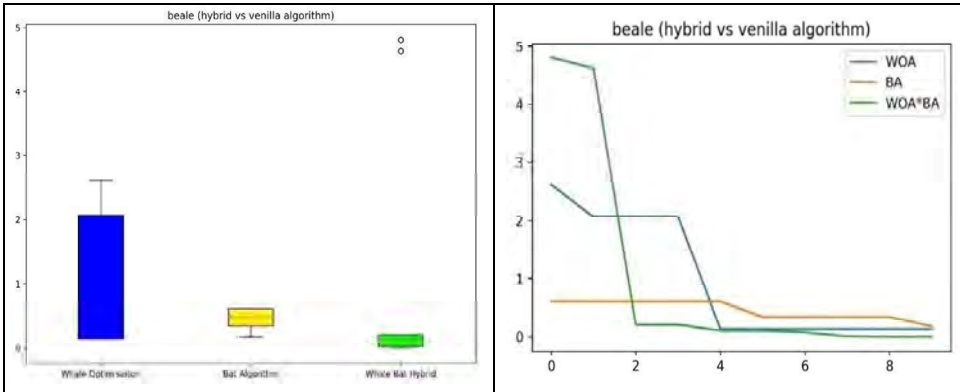
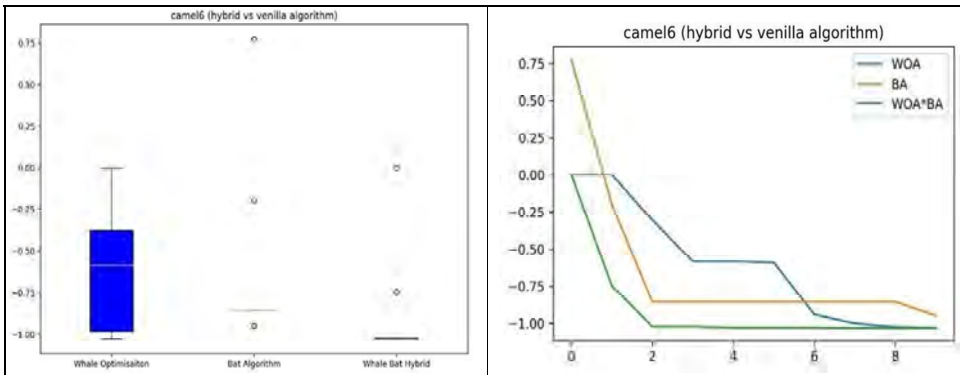


Figure 6 Hump camel back function box and convergence plots (see online version for colours)



4.1.2 Six-hump camelback function

Figure 6 shows box and convergence plots depict the convergence of hybrid and vanilla algorithms over the six hump camel-back function. It can be seen that the hybrid has a better convergence rate right from the start. The whale performs comparatively poorer but eventually reaches the global minima. The BA converges better but gets stuck in a local minimum towards the end.

4.1.3 Easom function

Figure 7 shows box and convergence plots depict the convergence of hybrid and vanilla algorithms over Easom function. Here, we notice a change as per how the BA has an overall better convergence than the hybrid and whale algorithm, but eventually, six the hybrid converges better. The initial variance is due to the poor performance by the whale algorithm, which is carried forward to the hybrid.

Figure 7 Easom function box and convergence plots (see online version for colours)

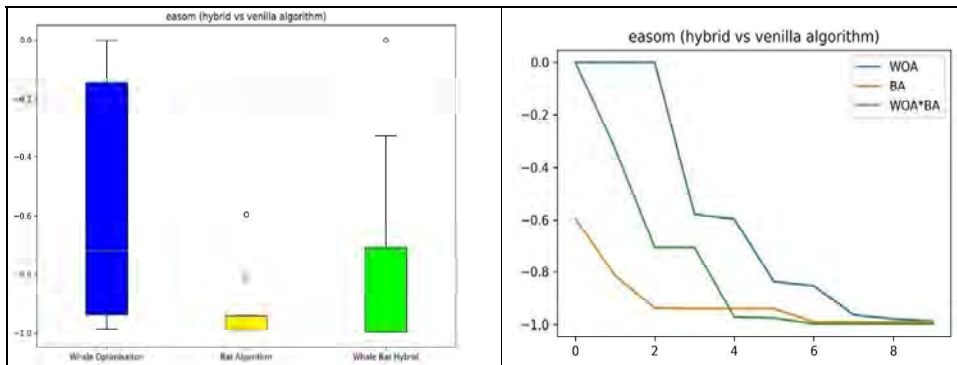
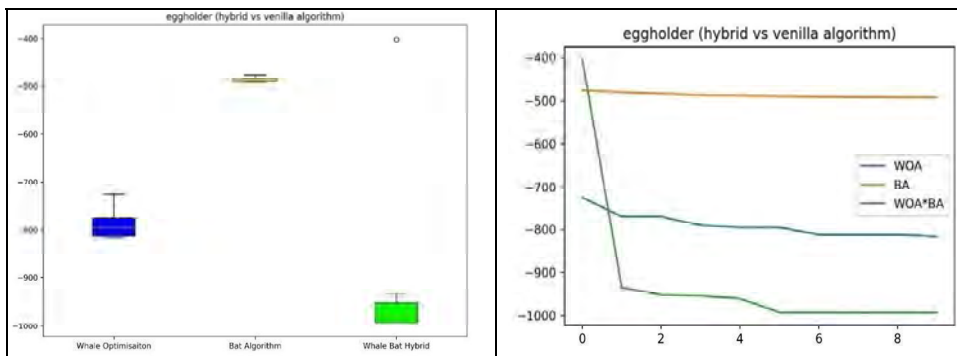


Figure 8 Egg holder function box and convergence plots (see online version for colours)



4.1.4 Eggholder function

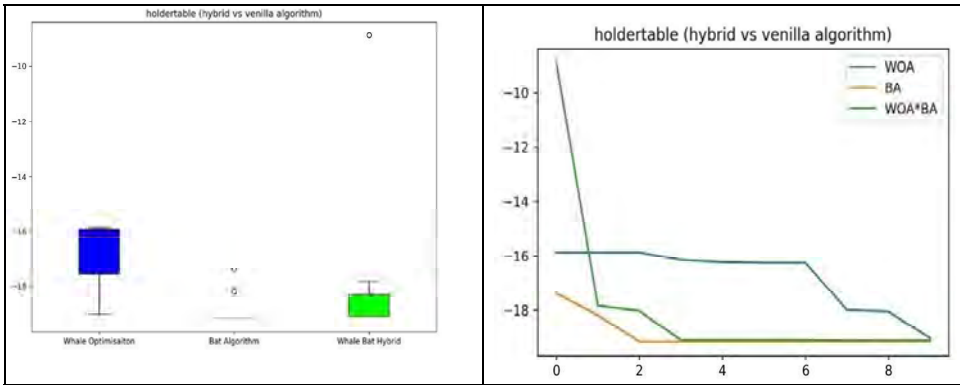
Figure 8 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over eggholder function. Here, we see an issue with the vanilla bat and

whale algorithms, as to getting stuck in local minima. However, the hybrid beats both by converging to the global minima quickly.

4.1.5 Holder table function

Figure 9 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over holder table function. This is another example of how the poorer performance of the default algorithm, in this case, the whale algorithm, affects the hybrid. The BA performs better at the beginning, but the hybrid eventually catches up.

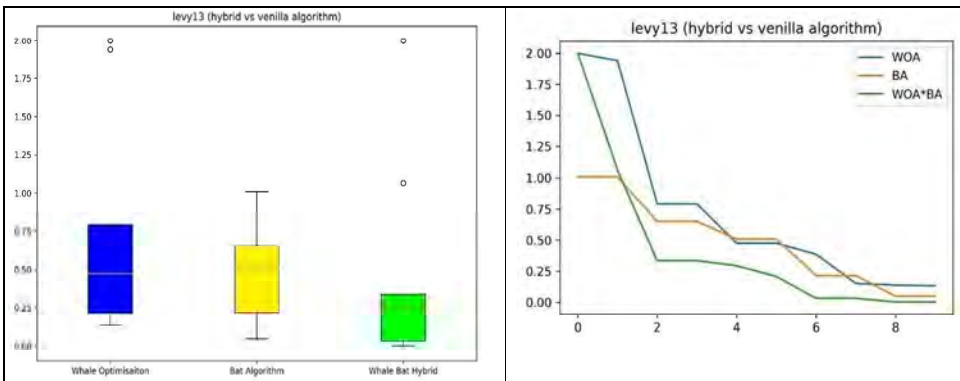
Figure 9 Holder-table function box and convergence plots (see online version for colours)



4.1.6 Levy N. 13 function

Figure 10 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over Levy N. 13 function. Here, we see that all algorithms show a similar trend but the hybrid performs the best, eventually reaching the actual global minima, which the default algorithms would take more iteration to reach.

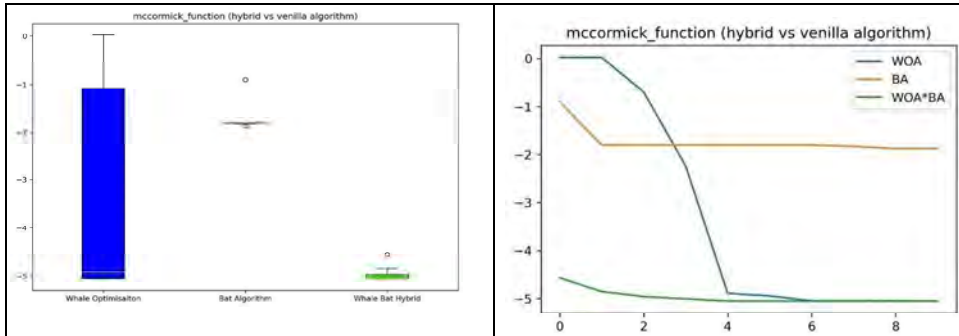
Figure 10 Levy N. 13 function box and convergence plots (see online version for colours)



4.1.7 McCormick function

Figure 11 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over McCormick function. Here, we see the hybrid performing better from the beginning. The whale algorithm eventually reaches the global minima, whereas the BA is stuck in a local minima.

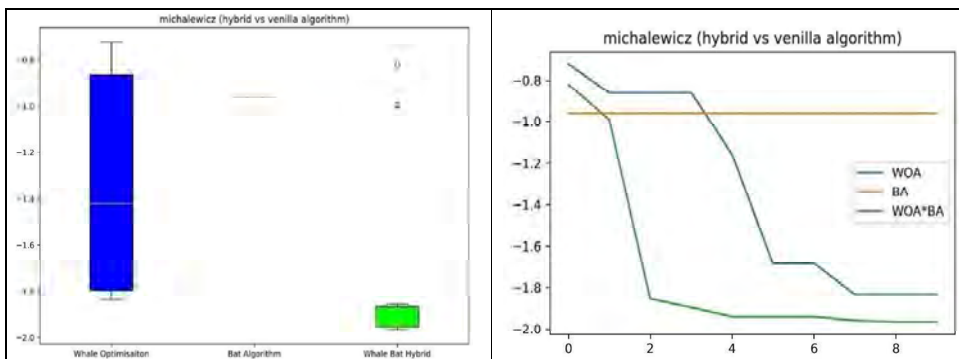
Figure 11 McCormick function box and convergence plots (see online version for colours)



4.1.8 Michalewicz function

Figure 12 box and convergence plots depict the convergence of hybrid and vanilla algorithms over Michalewicz function. These results are similar to the previous function, as the hybrid performs the best, followed by the whale algorithm, and the BA being stuck in a local minima.

Figure 12 Michalewicz function box and convergence plots (see online version for colours)



4.1.9 Rosenbrock function

Figure 13 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over Rosenbrock function. Here, we see a common trend among all algorithms, but again, the hybrid is the overall winner, followed by the bat and then the whale algorithm, which are apparently stuck in a local minima.

4.1.10 Schwefel function

Figure 14 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over Schwefel function. Here, we see the whale algorithm performing better initially, while the hybrid catches up later on. The bat shows a slow convergence for this function.

Figure 13 Rosenbrock function box and convergence plots (see online version for colours)

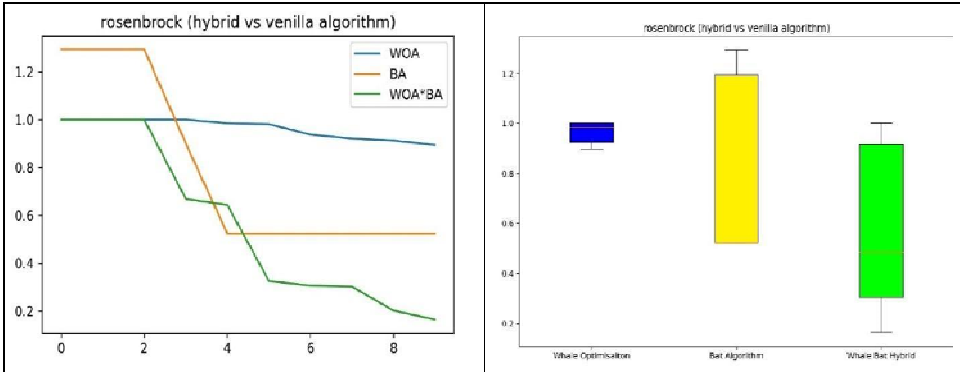
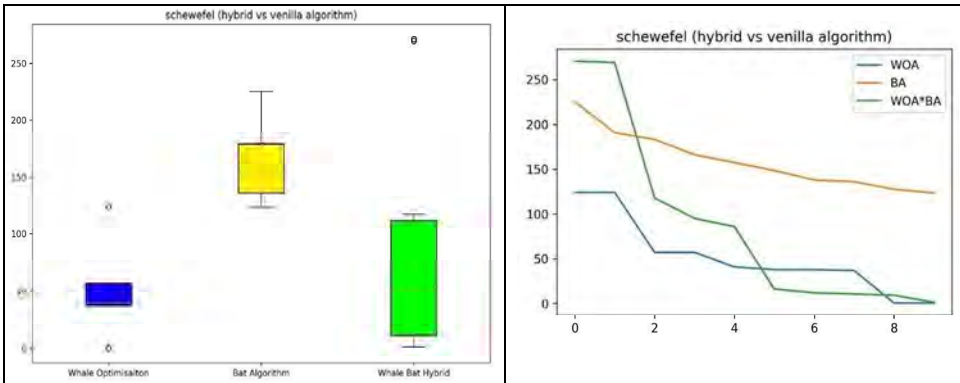


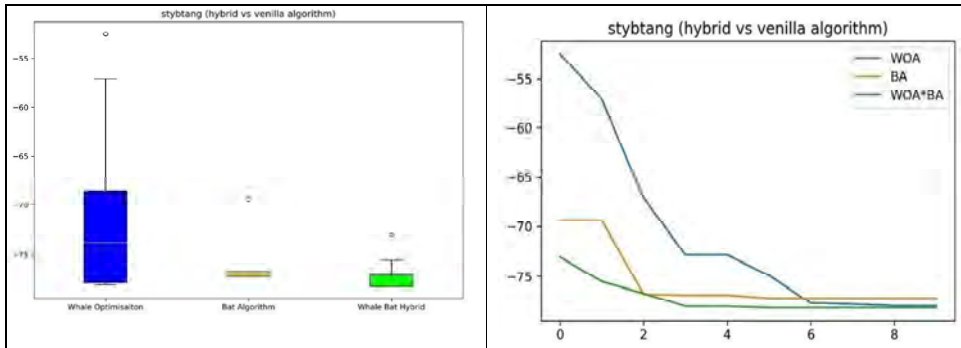
Figure 14 Schwefel function box and convergence plots (see online version for colours)



4.1.11 Styblinski-Tang function

Figure 15 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over Styblinski-Tang function. Here, we see a close common trend among all three algorithms since they eventually almost reach the same minima, but with the hybrid performing better overall.

Figure 15 Styblinski-Tang function box and convergence plots (see online version for colours)



4.1.12 Cross-in-tray function

Figure 16 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over cross-in-tray function. Here, we see a different outcome, showing the BA performing the best. The hybrid eventually converges too, but the whale algorithm occasionally gets stuck in some local minimas.

Figure 16 Cross-in-tray function box and convergence plots (see online version for colours)

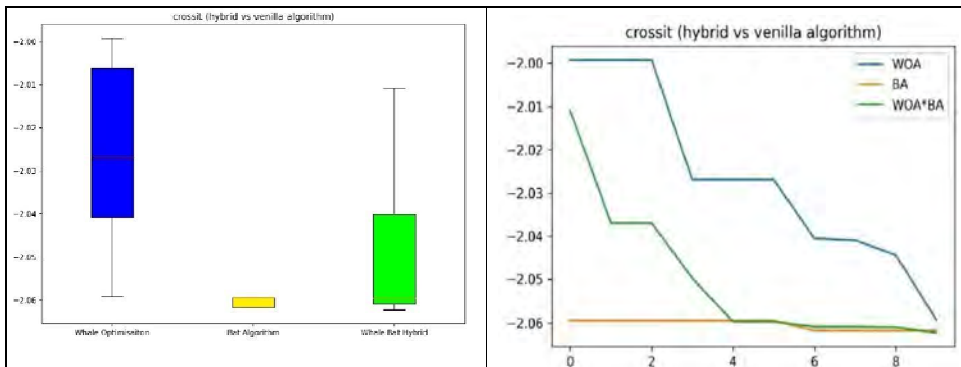
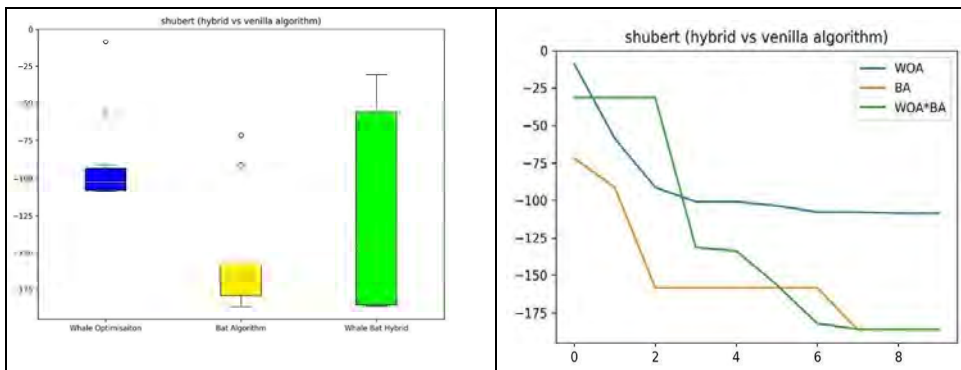


Figure 17 Shubert function box and convergence plots (see online version for colours)



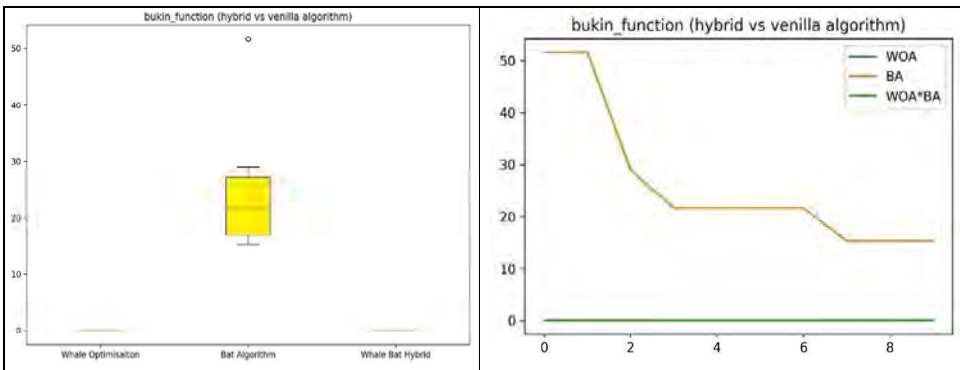
4.1.13 Shubert function

Figure 17 shows box and convergence plots depicting the convergence of hybrid and vanilla algorithms over Shubert function. Like the above function, this one also shows the BA performing the best, followed by the hybrid which eventually converges better. The whale algorithm gets stuck in a local minima after the second iteration.

4.1.14 Bukin N. 6 function

Figure 18 box and convergence plots depict the convergence of hybrid and vanilla algorithms over Buckin N. 6 function. These graphs show the whale and hybrid algorithm directly attaining global minima, whereas the BA shows a slow convergence.

Figure 18 Buckin N. 6 function box and convergence plots (see online version for colours)



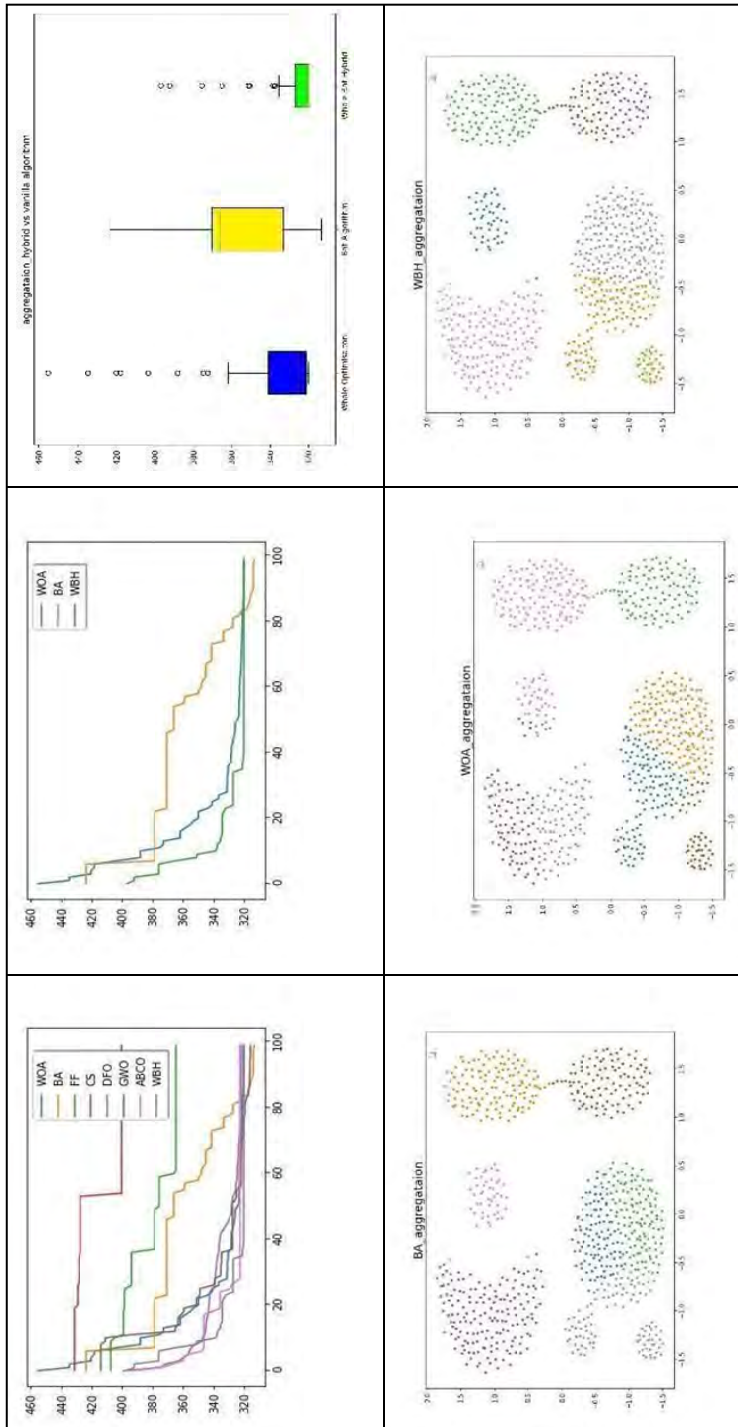
4.2 Inference on real-life/artificial datasets

The following results are concerning tests on a set of real-world and artificial datasets, with six graphs for each dataset. The first convergence graph compares the hybrid with seven vanilla meta-heuristic algorithms, and the second graph compares it with only the constituent whale optimisation and BAs. The third is a box graph which represents the same information as the second graph. The last three graphs are visualisations of the clustering performed by the BA, WOA and the hybrid, respectively.

4.2.1 Aggregation dataset

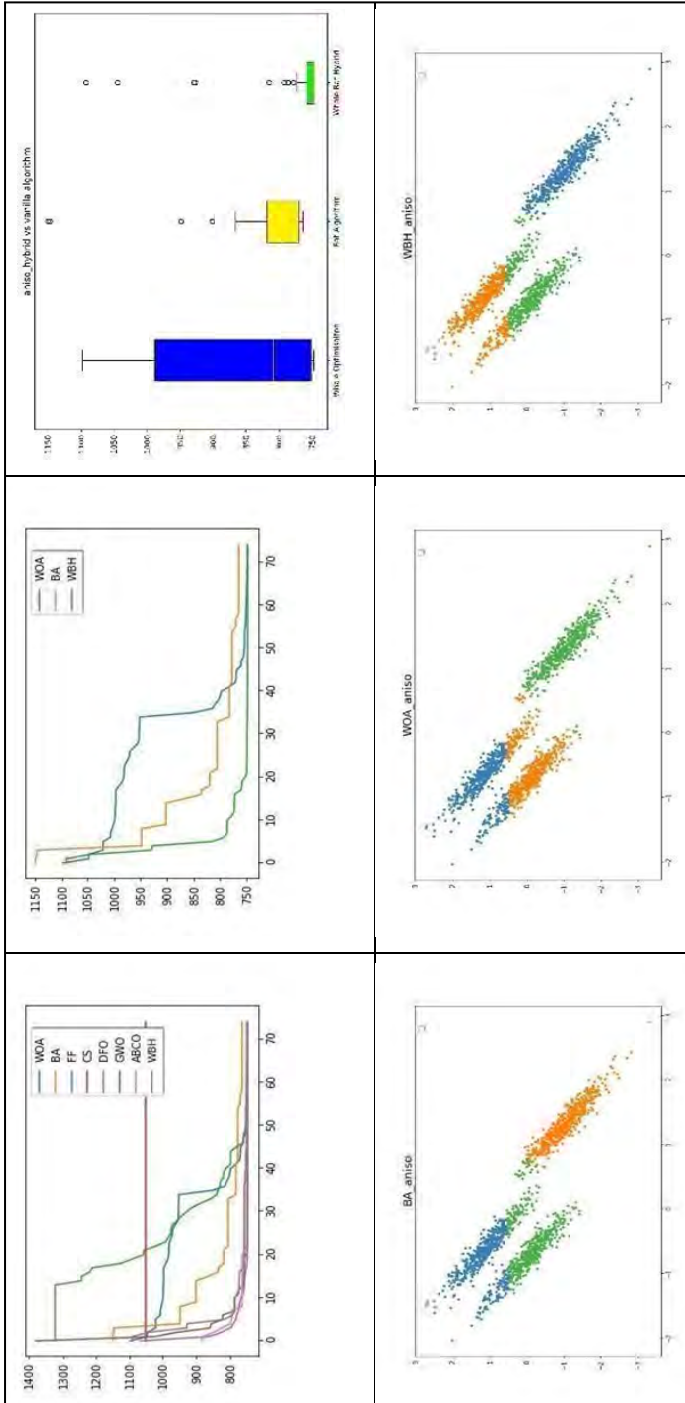
Figure 19 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for the aggregation dataset. As seen, the hybrid performs better than BA and WOA and also better than most of the other vanilla algorithms. The clustering data is better among the hybrid and WOA when compared to BA.

Figure 19 Aggregation dataset results (see online version for colours)



Note: Number of clusters: 7.

Figure 20 Aniso dataset results (see online version for colours)



Note: No. of clusters: 3.

4.2.2 *Aniso dataset*

Figure 20 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for Aniso dataset. Here, we see some of the other vanilla algorithms performing better, but nevertheless, the hybrid performs better than BA and WOA. The clustering looks identical due to all algorithms almost reaching the same minima. The clustering graphs are not perfect for any algorithm since the convergence is based on a naive Euclidean closest distance formula. A better fitness function would result in more accurate graphs.

4.2.3 *Appendicitis dataset*

Figure 21 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for appendicitis dataset, which is a real-life dataset. Here, we see the hybrid being the best performer among all the vanilla algorithms. Not much can be inferred from the clustering graphs as they are based on a real-life dataset without much structure.

4.2.4 *Balance dataset*

Figure 22 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for the balance dataset. We see that the hybrid is not the best performer for this dataset, but eventually reaches the same global minima. The clustering graphs look identical since all algorithms reach the same minima.

4.2.5 *Banknote dataset*

Figure 23 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for banknote dataset, which is a real-life dataset. The hybrid is one of the best performers as it converge the fastest. Not much can be inferred from the clustering data since all algorithms reach the same minima.

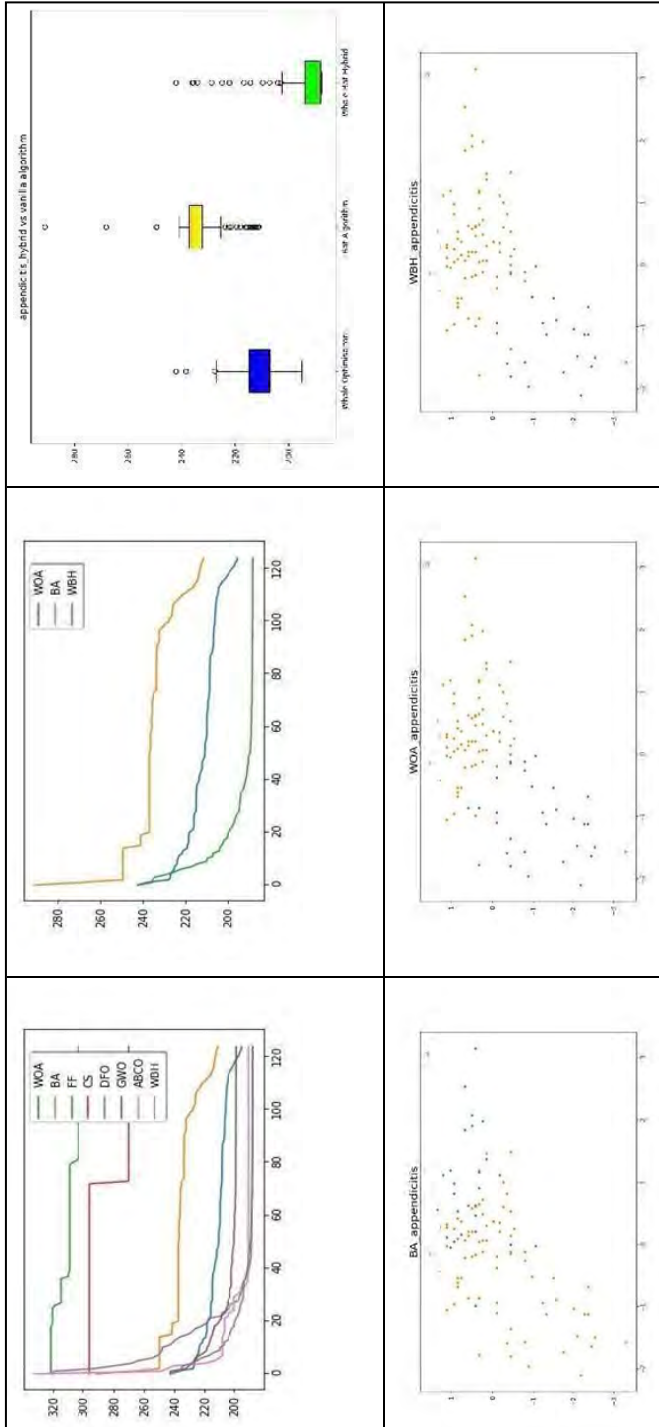
4.2.6 *Blood dataset*

Figure 24 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for blood dataset, which is a real-life dataset. Again, the hybrid is one of the best performers due to its faster convergence. The clustering graphs help visualise how the clustering occurs among the algorithms.

4.2.7 *Diagnosis dataset*

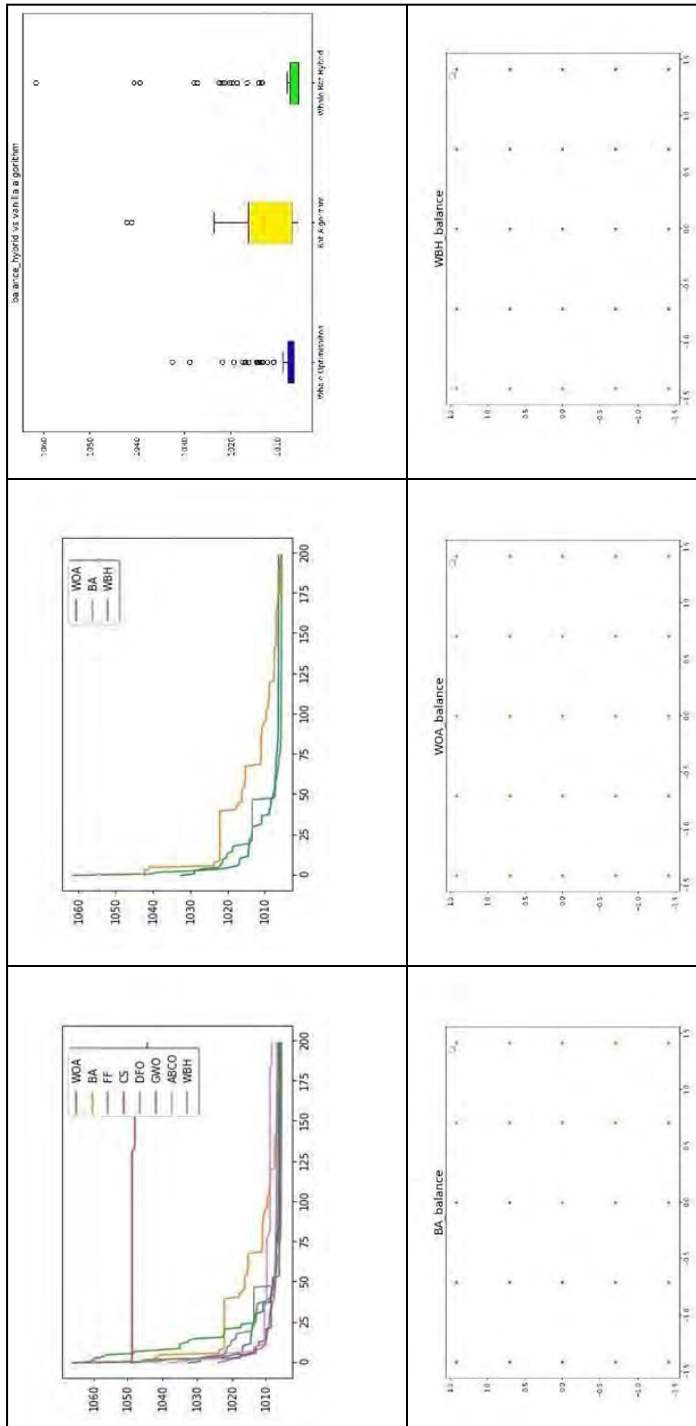
Figure 25 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for the diagnosis dataset, which is a real-life dataset. This dataset shows a clear distinction as to how the hybrid performs better among the other algorithms. The clustering graphs also provide valuable information, as it can be seen that the hybrid can isolate the two clusters accurately, but WOA and BA have a few mispredictions in both clusters.

Figure 21 Appendicitis dataset results (see online version for colours)



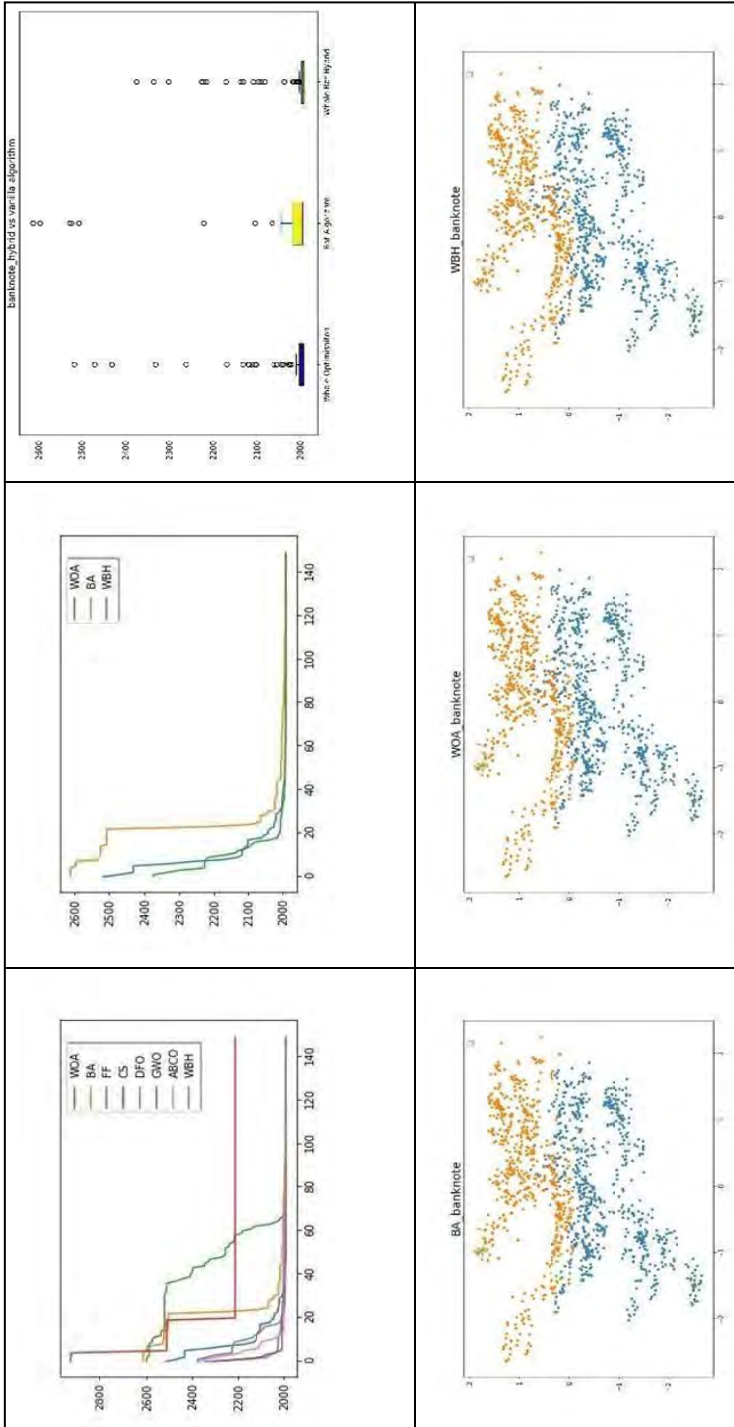
Note: No. of clusters: 2.

Figure 22 Balance dataset results (see online version for colours)



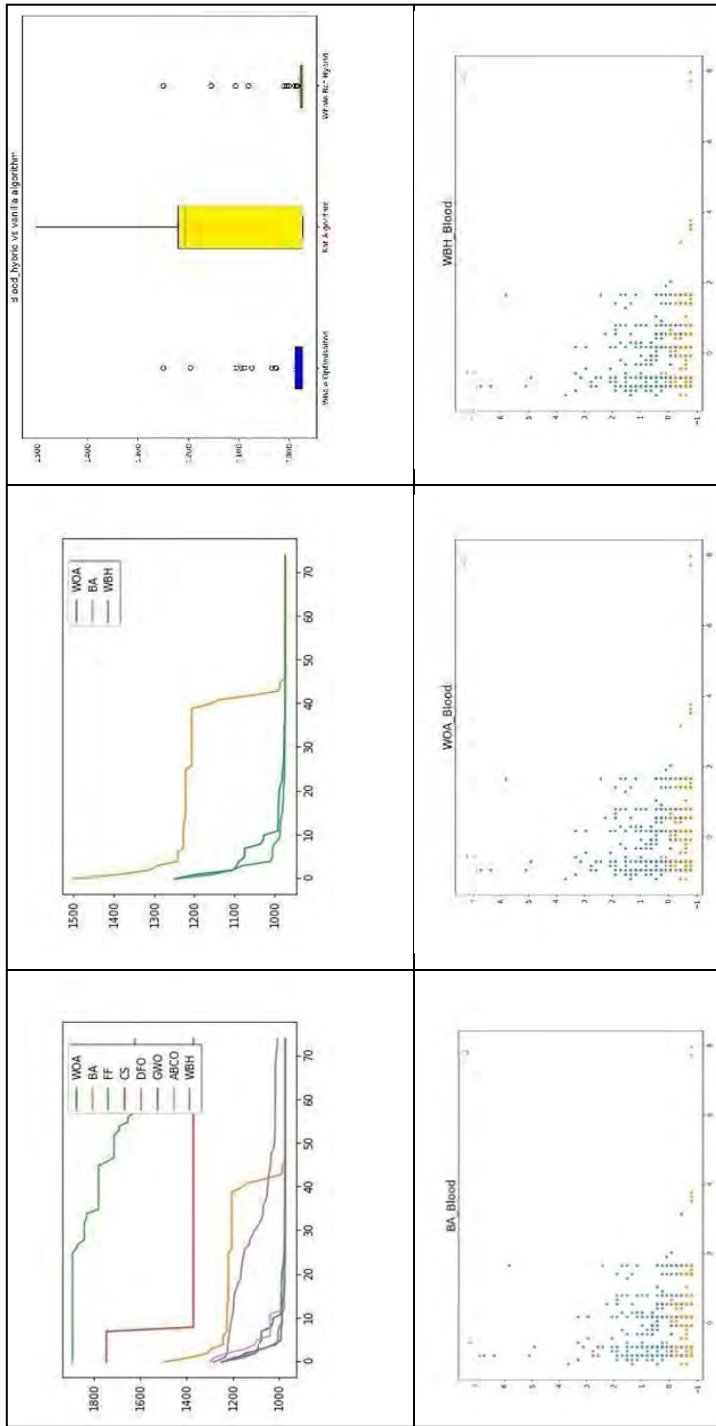
Note: No. of clusters: 3.

Figure 23 Banknote dataset results (see online version for colours)



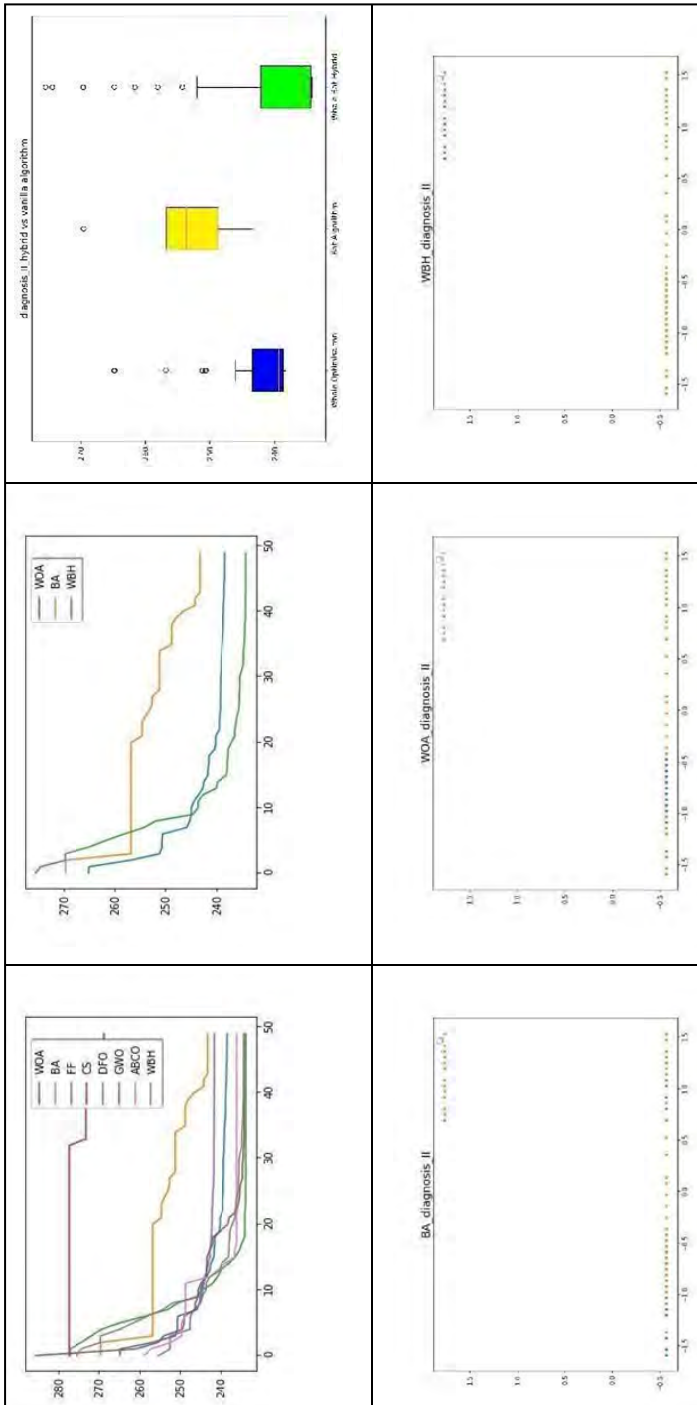
Note: No. of clusters: 2.

Figure 24 Blood dataset results (see online version for colours)



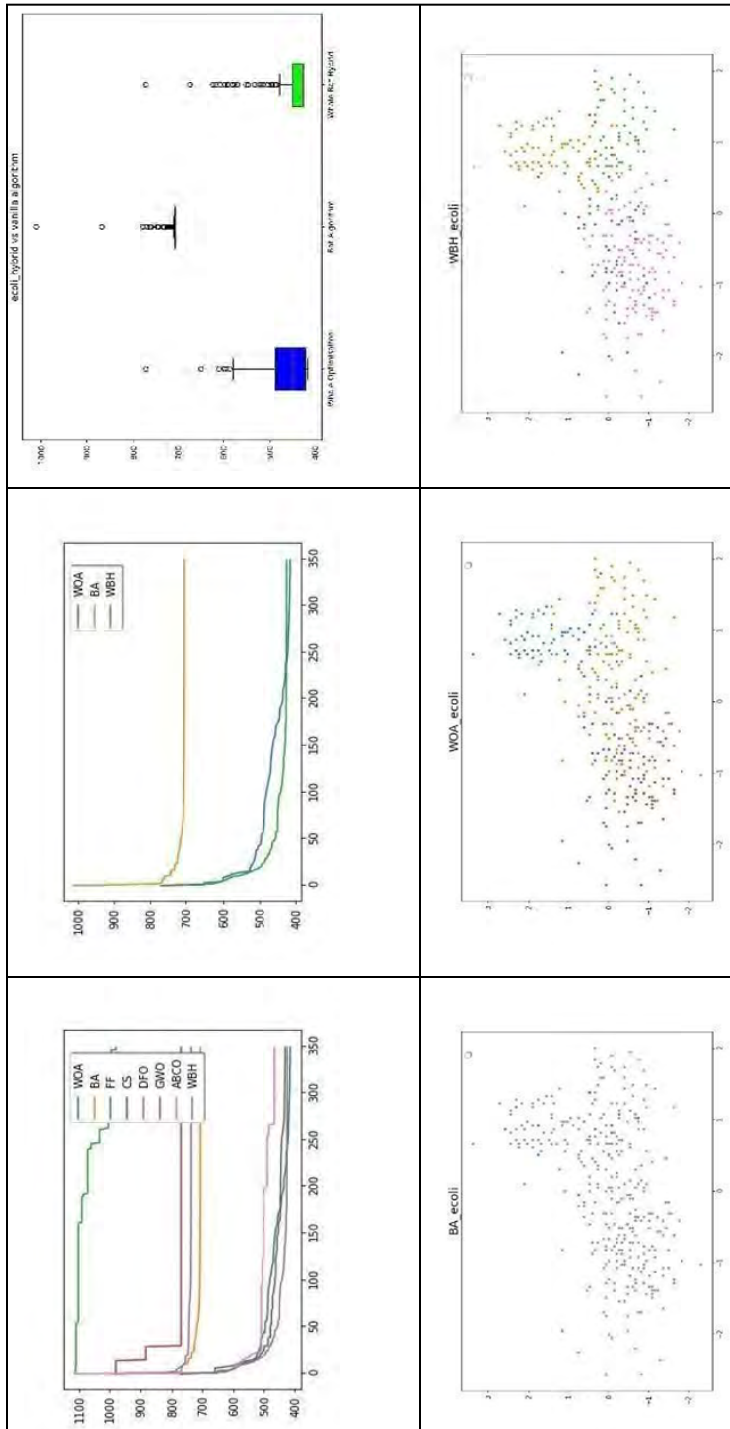
Note: No. of clusters: 2.

Figure 25 Diagnosis dataset results (see online version for colours)



Note: No. of clusters: 2.

Figure 26 *E. coli* dataset results (see online version for colours)



Note: No. of clusters: 7.

4.2.8 *E. coli* dataset

Figure 26 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for *E. coli* dataset, which is a real-life dataset. The hybrid is one of the best performers, and the WOA performs almost similarly. It can be seen for the clustering graphs, how the hybrid and WOA show variability in up to seven clusters, but the BA clusters the points in only 2 or 3 clusters.

4.2.9 *Flame* dataset

Figure 27 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for flame dataset. The hybrid is one of the best performers here, as can be seen in the convergence graphs. The clustering graphs are almost similar, with the BA with a few mispredictions. The clustering graphs are not perfect for any algorithm since the convergence is based on a naive Euclidean closest distance formula. A better fitness function would result in more accurate graphs.

4.2.10 *Glass* dataset

Figure 28 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for glass dataset, which is a real-life dataset. The hybrid is the clear winner among all algorithms among the convergence graphs. It can be seen that the hybrid and WOA display good variability of clustering with up to six clusters whereas the BA, which got stuck in a local minima, has only detected up to three major clusters.

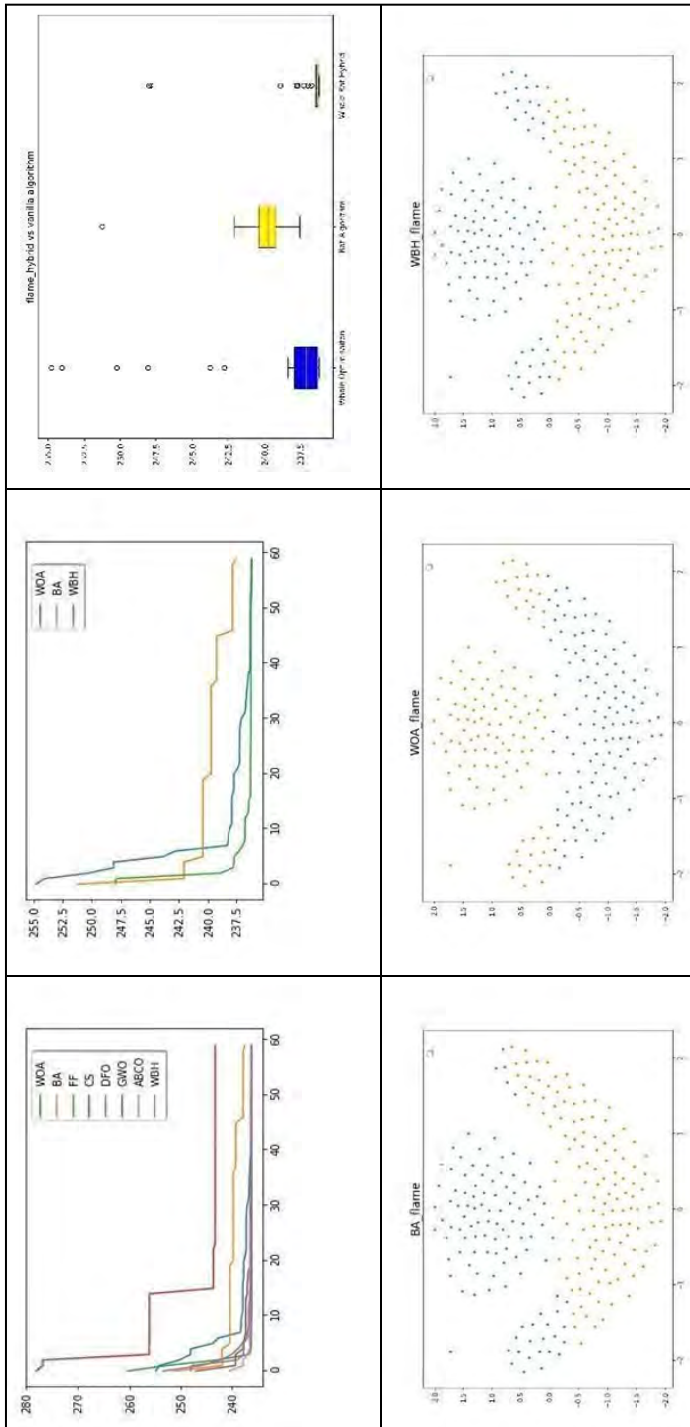
4.2.11 *Heart* dataset

Figure 29 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for heart dataset, which is a real-life dataset. The hybrid is one of the best performers, with WOA being in the same lines, but BA is stuck in a local minima. The cluster does not help much in deciding which is the best algorithm but does show the poor performance of BA, having detected only one major cluster.

4.2.12 *Ionosphere* dataset

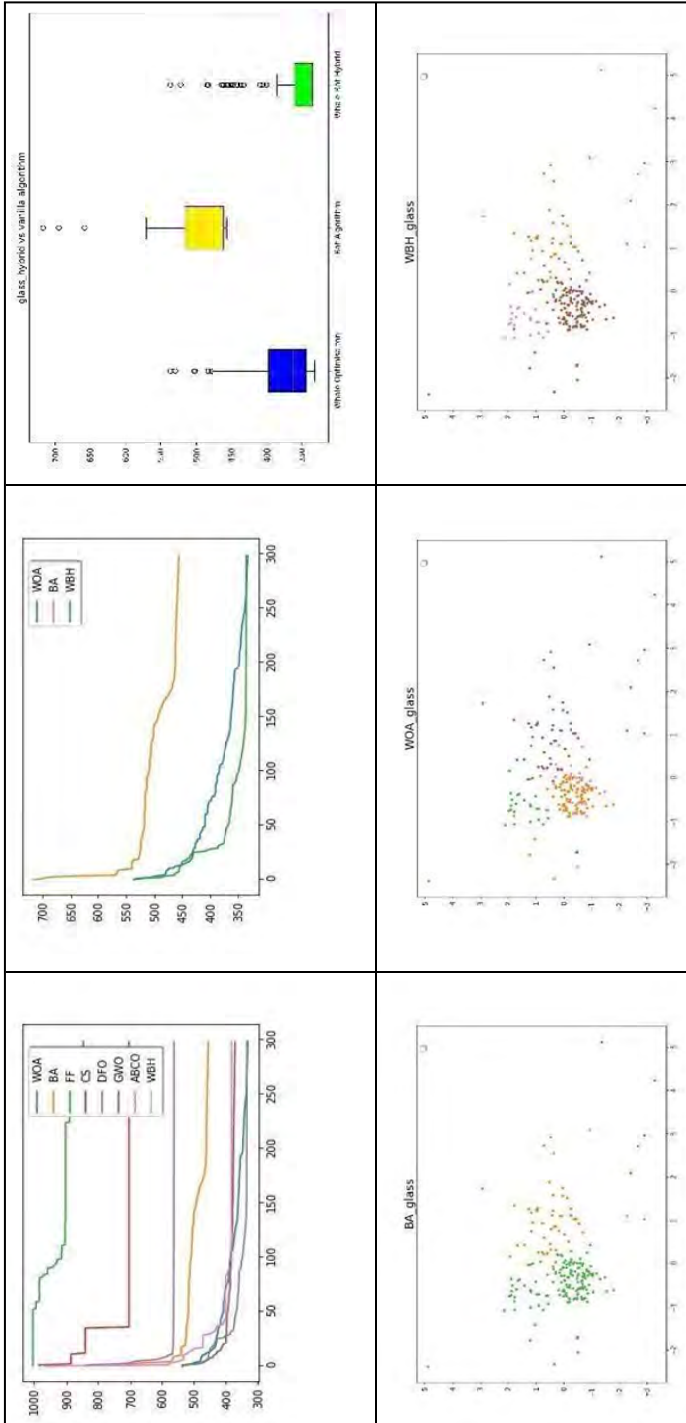
Figure 30 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for ionosphere dataset, which is a real-life dataset. Again, the hybrid is a clear winner among all algorithms. Not much can be inferred from the clustering graphs as the points coincide at two points. But this dataset is one with 34 features. This is a good example of how well the hybrid algorithm handles data with high dimensions, while others like BA gets stuck in local minima due to the 'COD'.

Figure 27 Flame dataset results (see online version for colours)



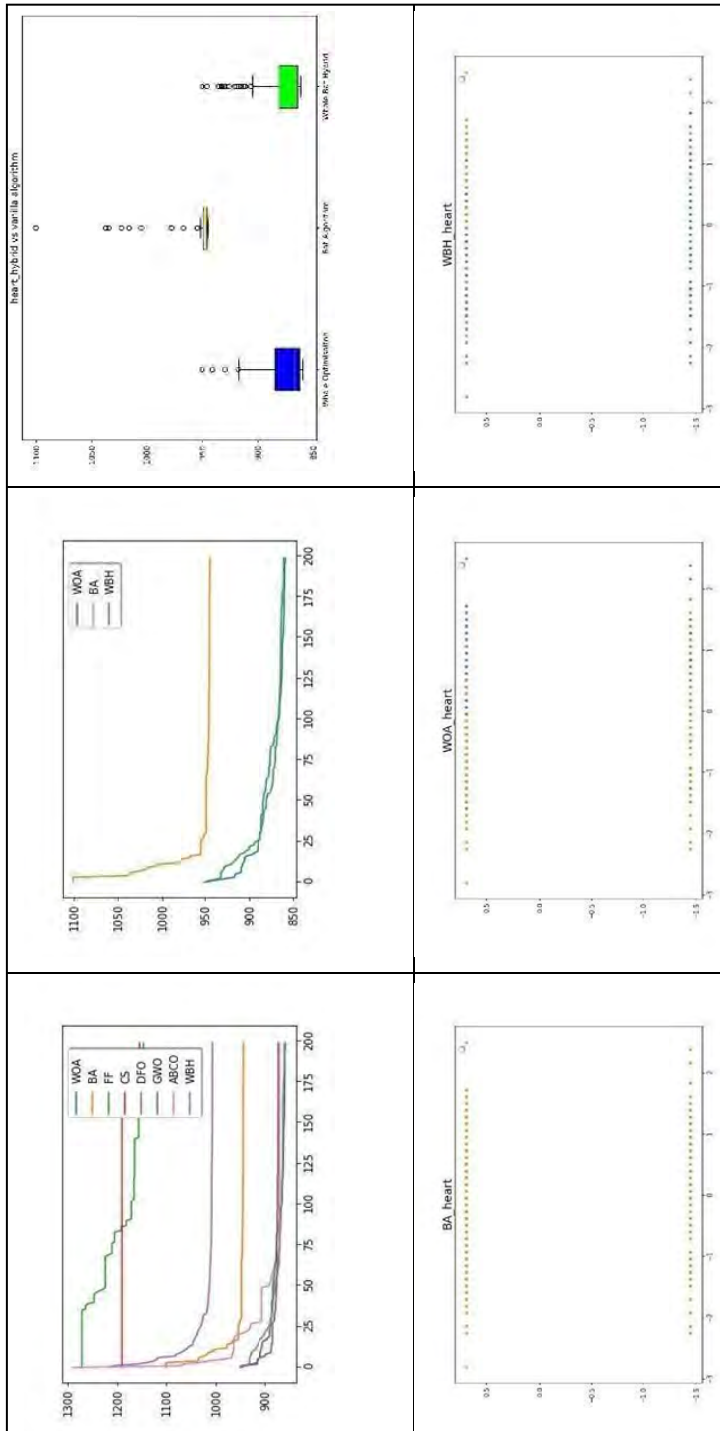
Note: No. of clusters: 2.

Figure 28 Glass dataset results (see online version for colours)



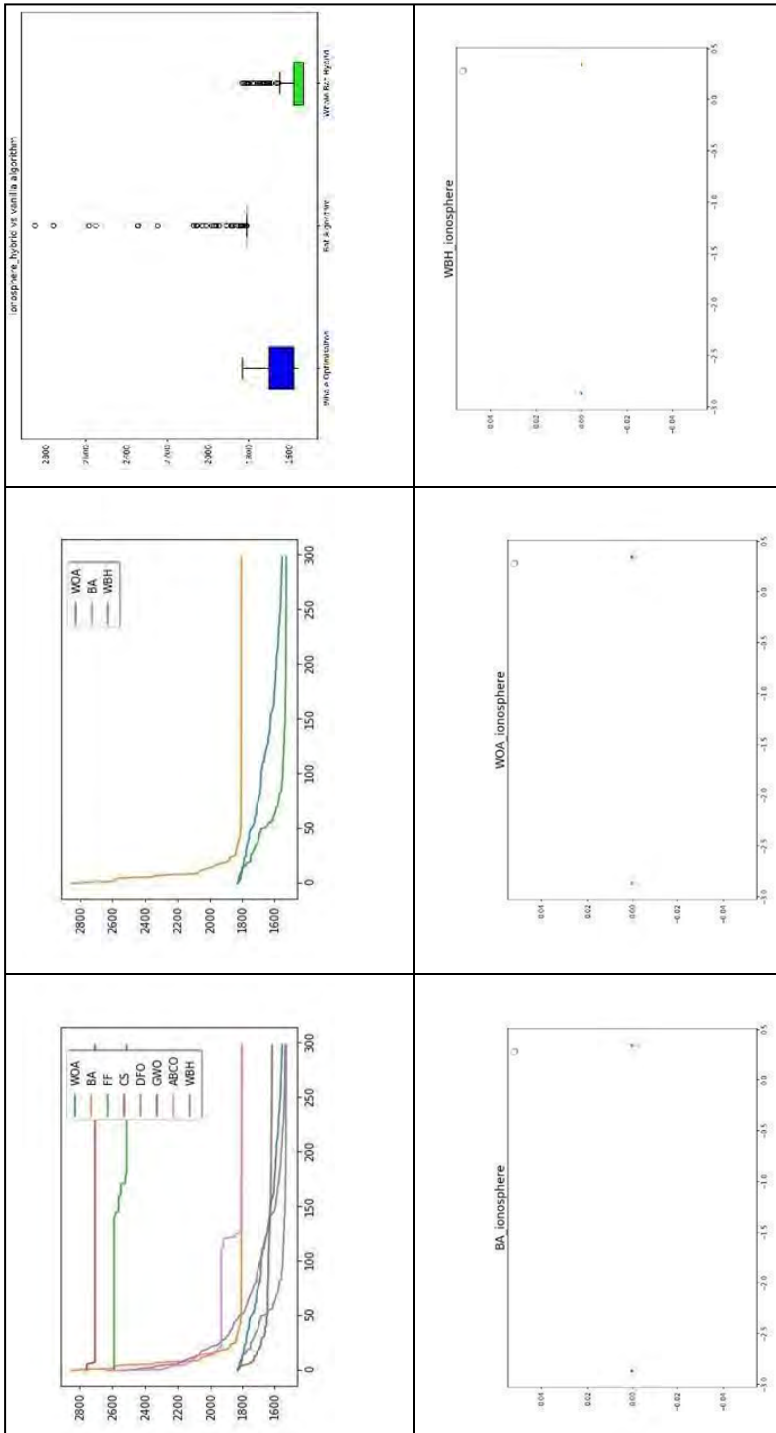
Note: No. of clusters: 6.

Figure 29 Heart dataset results (see online version for colours)



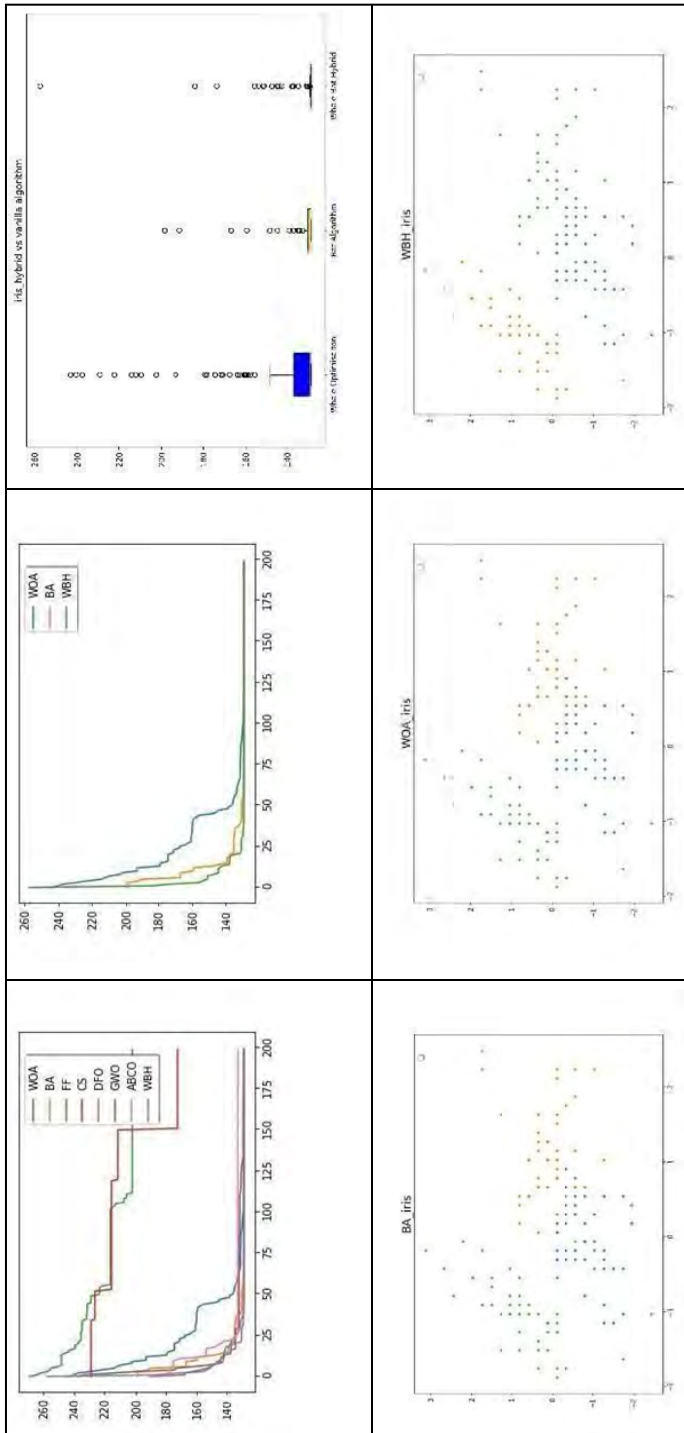
Note: No. of clusters: 2.

Figure 30 Ionosphere dataset results (see online version for colours)



Note: No. of clusters: 2.

Figure 31 IRIS dataset results (see online version for colours)



Note: No. of clusters: 3.

4.2.13 *IRIS dataset*

Figure 31 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for IRIS dataset, which is a real-life dataset. This is a relatively easier dataset, which again shows better convergence in the hybrid. The clustering graphs are the same since all the algorithms achieved the same minima.

4.2.14 *Moons dataset*

Figure 32 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for moons dataset. This is another example of the above scenario showing better convergence in hybrid, with no much difference in the clustering graphs. The clustering graphs are not perfect for any algorithm since the convergence is based on a naive Euclidean closest distance formula. A better fitness function would result in more accurate graphs.

4.2.15 *Path-based dataset*

Figure 33 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for path-based dataset. The hybrid shows faster convergence, and eventually all algorithms reach the global minima. The clustering graphs are not perfect for reasons mentioned before.

4.2.16 *Sonar dataset*

Figure 34 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for sonar dataset, which is a real-life dataset. The hybrid again is the clear winner with fastest convergence and reaching the global minima. The bat performs poorly as it detects only a single major cluster. This dataset is also a good example of a high dimension dataset with up to 60 features, which normally causes issues such as the 'COD', but the hybrid is able to handle it better than most algorithms.

4.2.17 *Varied dataset*

Figure 35 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for varied dataset. The hybrid shows the better convergence rate. Additionally, the cluster graphs show a more accurate level of distinction among the clusters for the hybrid algorithm, when compared to WOA and BA.

4.2.18 *Vary-density dataset*

Figure 36 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for the vary-density dataset. Like before the hybrid has better convergence compared to other algorithms. Not much can be inferred from the cluster graphs.

4.2.19 Vertebral II dataset

Figure 37 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for vertebral II dataset, which is a real-life dataset. This is a case where the hybrid may be better than the WOA and BA, but there are a few vanilla algorithms that perform better than the hybrid.

4.2.20 Vertebral III dataset

Figure 38 represents the graphical comparison of the hybrid with the WOA and BA, along with other vanilla meta-heuristic algorithms, for vertebral III dataset, which is a real-life dataset. The hybrid is one of the best algorithms with the best convergence rate. This proves to be a deciding factor in constrained environments.

4.3 Inference on unlabelled real-life clustering data

For more concrete inferences on real-world data, testing was conducted on two unlabelled real-world datasets.

4.3.1 Credit card data

This dataset requires the development of customer segmentation/clustering to define marketing strategy. The dataset summarises the usage behaviour of about 9,000 active credit card holders in a certain time period. Each entry has 18 behavioural variables.

Figure 39 shows the graphical comparison of hybrid with the vanilla bat and whale algorithm. Number of clusters was chosen to be 3 using the elbow method. The data was pre-processed, and after normalisation, standard scaling of data, and performing principal component analysis to condense the number of variables, the dataset was tested on the three algorithms. Here, the hybrid converges faster compared to the vanilla bat and whale algorithms and also performs better clustering overall. In contrast, the BA gets stuck in a local minima, and the WOA lags behind the hybrid.

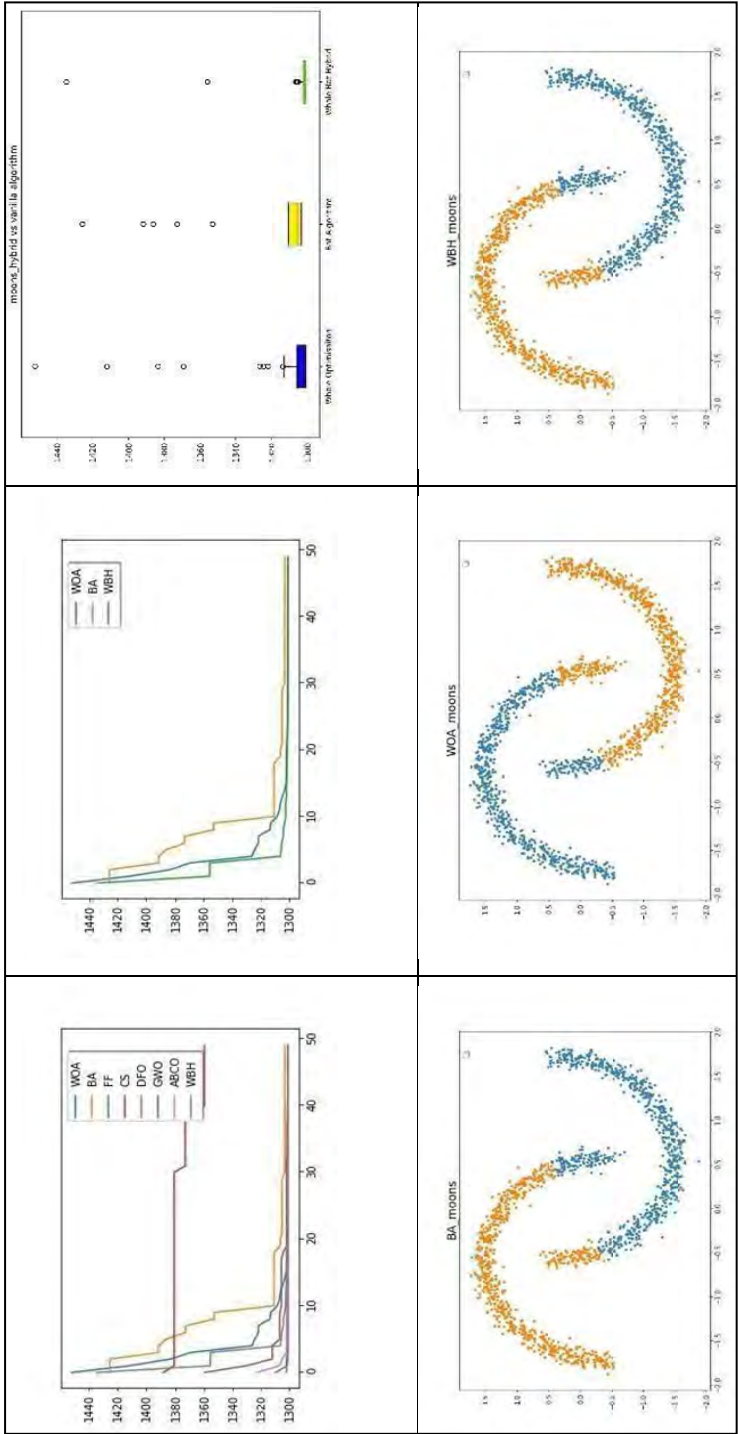
4.3.2 COVID-19 diagnosis data

This data will help to identify whether any person has the COVID-19 disease or not, based on some predefined standard symptoms. The dataset consists of several variables for each entry, such as fever, cough, tiredness, pains, etc. We use 10,000 entries from the dataset for our inference purposes.

Figure 40 shows the graphical comparison of hybrid with the vanilla bat and whale algorithm. The number of clusters was chosen to be two due the diagnosis result being either having COVID or not having COVID. The data was pre-processed, and after normalisation, standard scaling of data and performing principal component analysis to condense the number of variables, the dataset was tested on the three algorithms. Here, the hybrid converges much faster compared to the vanilla bat and whale algorithms and also performs better clustering overall. In contrast, the BA is stuck in a local minima, giving poor clustering results, and the whale algorithm, lagging behind the hybrid.

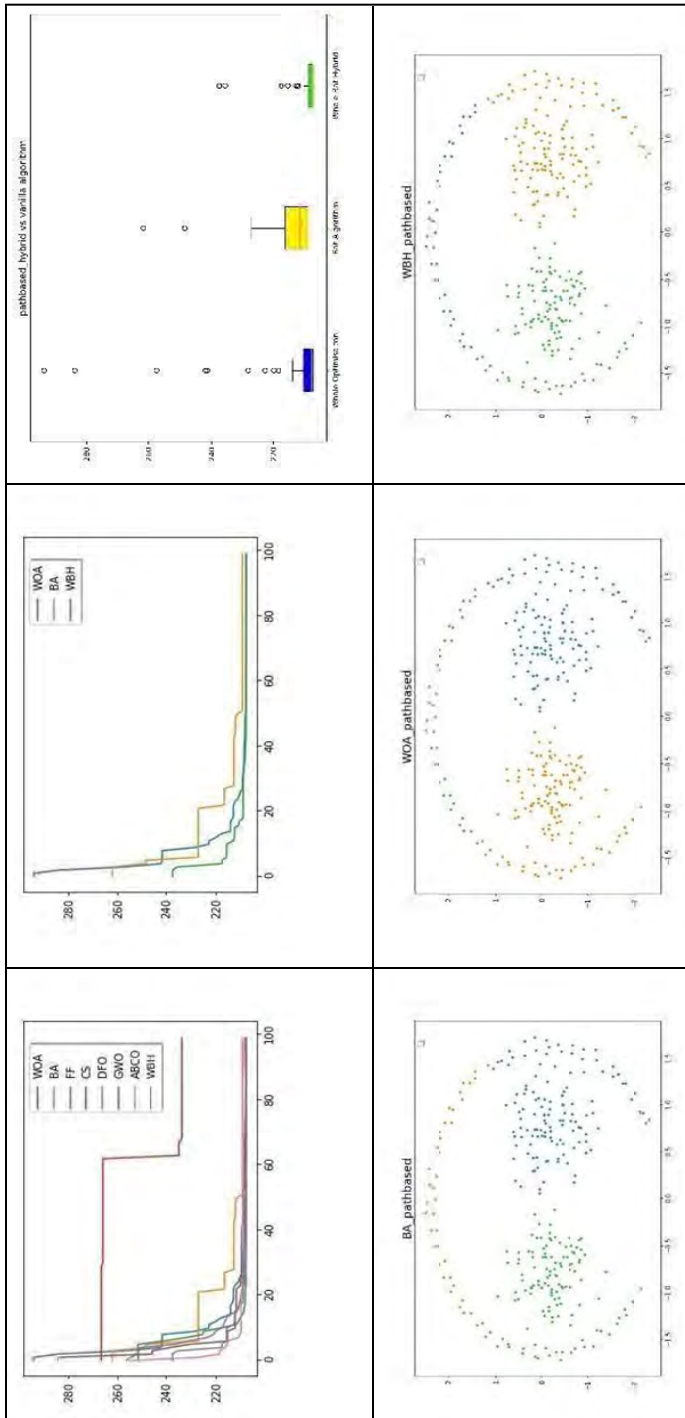
These two unlabelled datasets tested on support the inference derived from the testing performed on the CEC functions and labelled datasets, to prove that the hybrid performs a better task by converging faster and providing more accurate results.

Figure 32 Moons dataset results (see online version for colours)



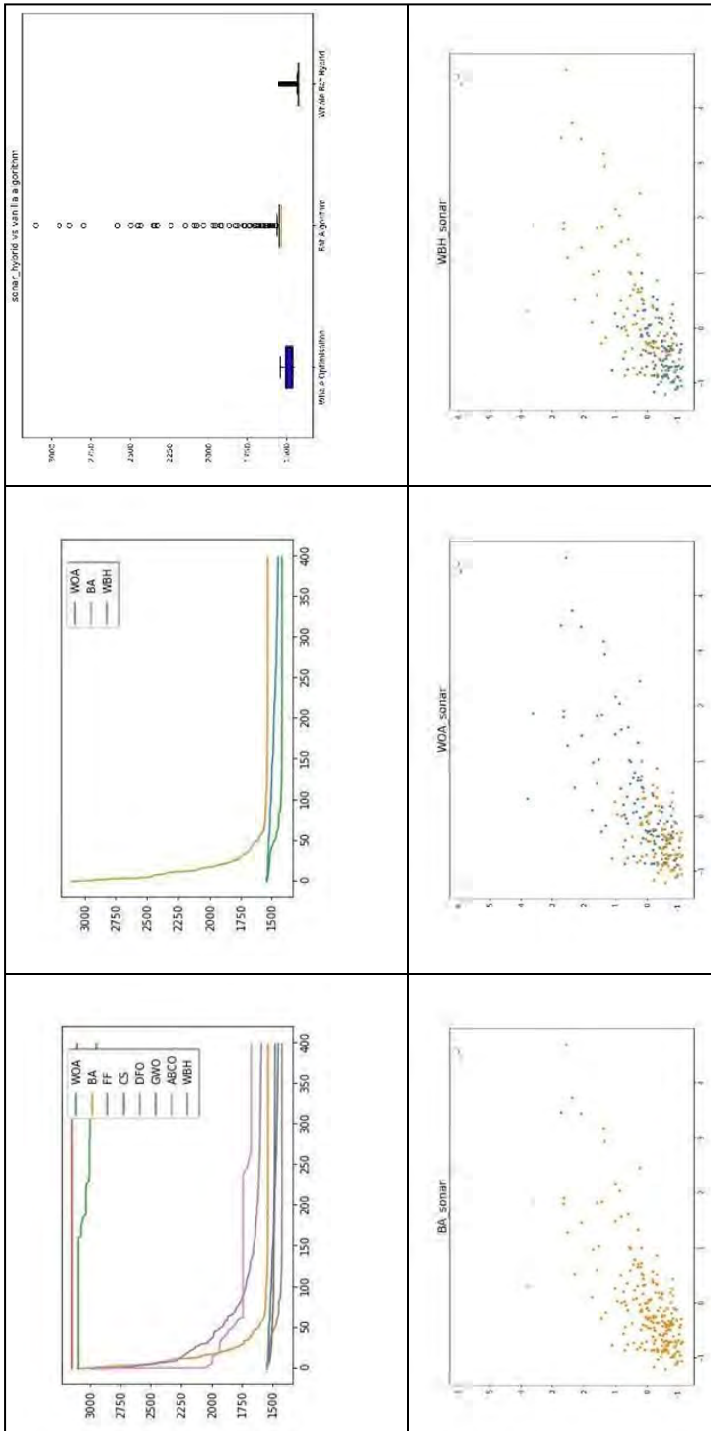
Note: No. of clusters: 2.

Figure 33 Path-based dataset results (see online version for colours)



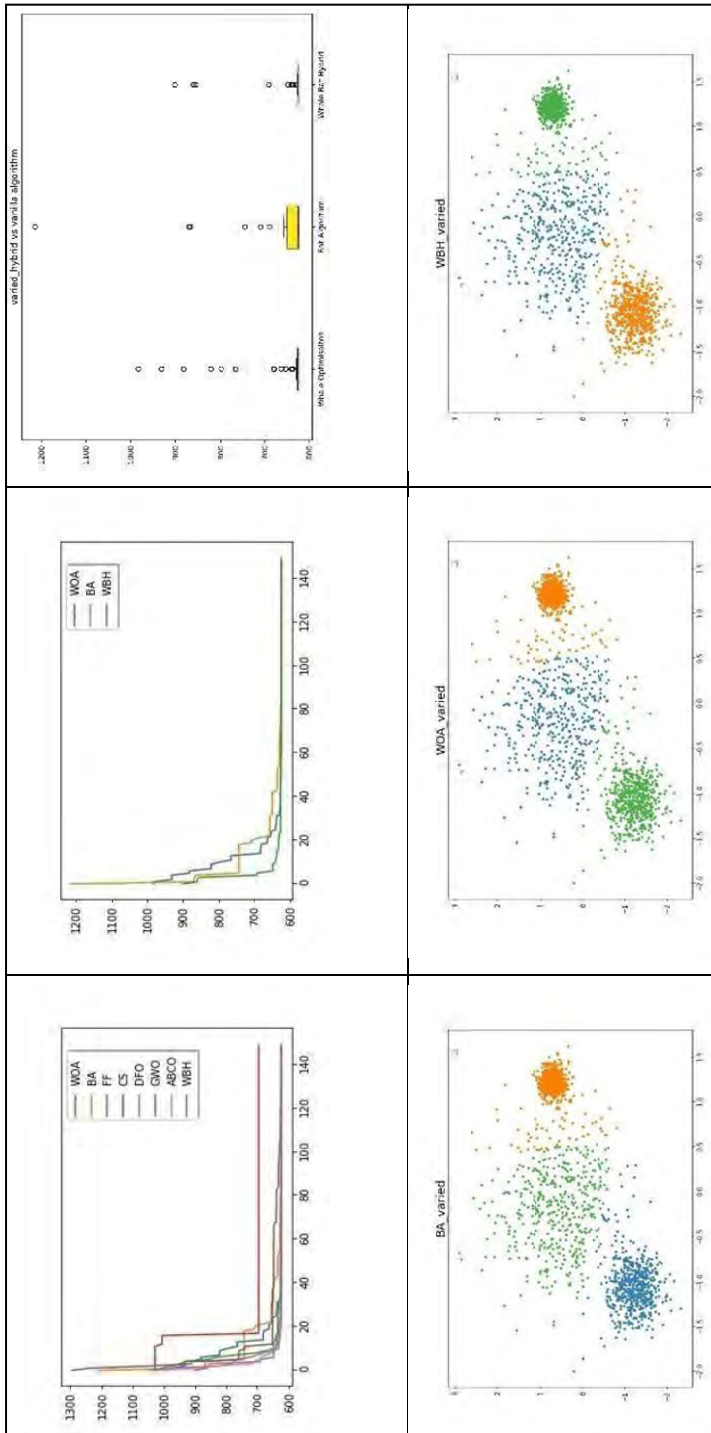
Note: No. of clusters: 3.

Figure 34 Sonar dataset results (see online version for colours)



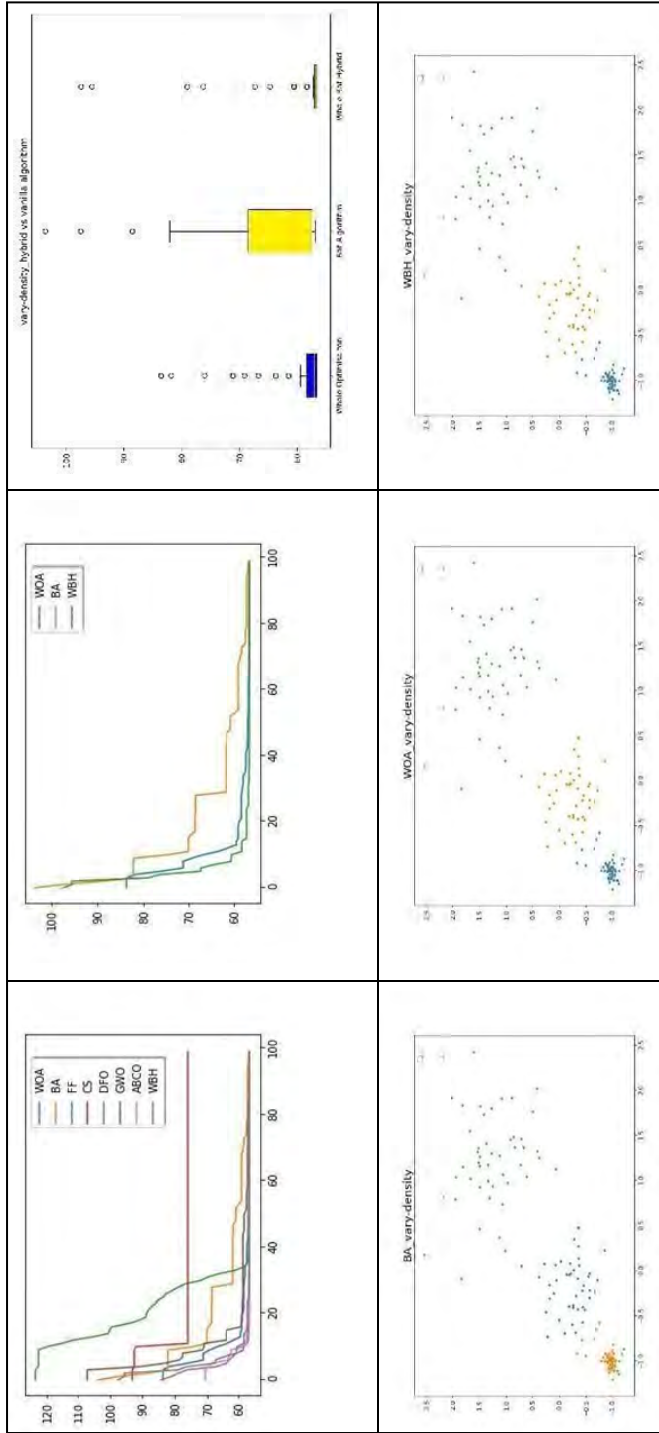
Note: No. of clusters: 2.

Figure 35 Varied dataset results (see online version for colours)



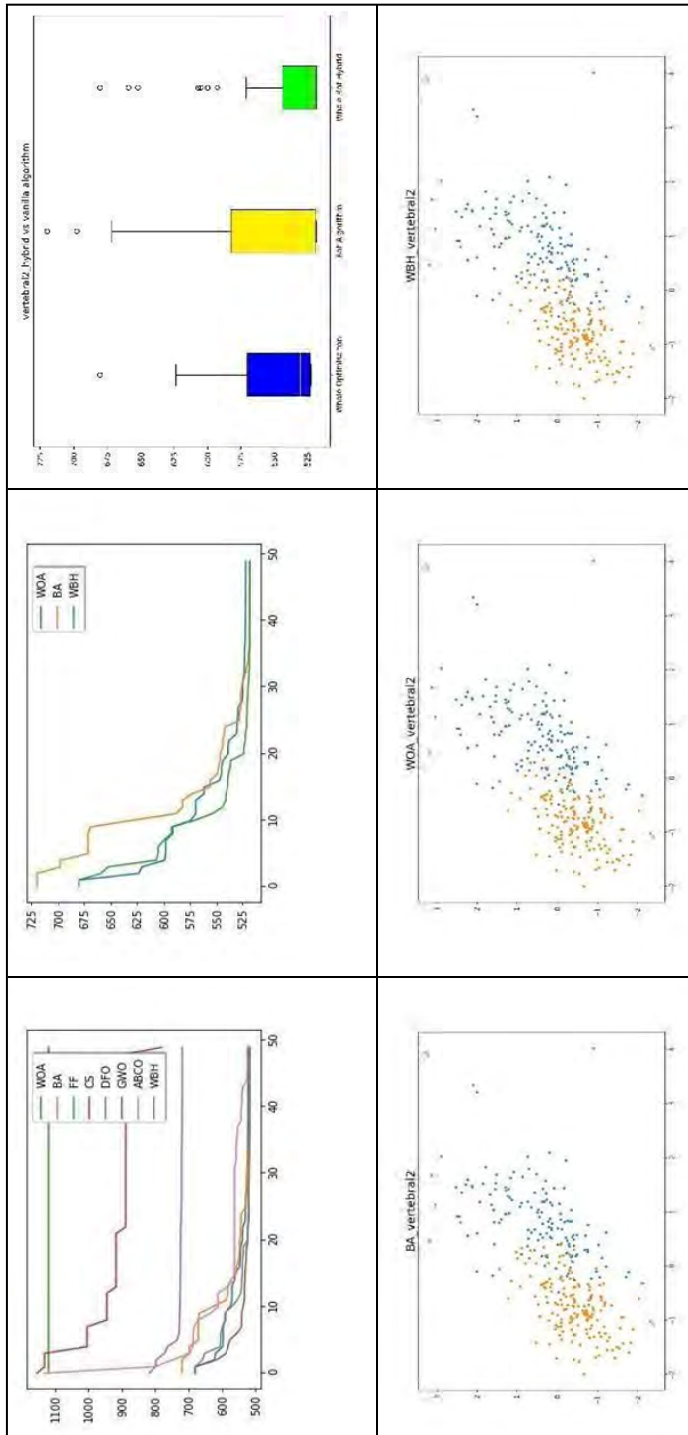
Note: No. of clusters: 3.

Figure 36 Varied-density dataset results (see online version for colours)



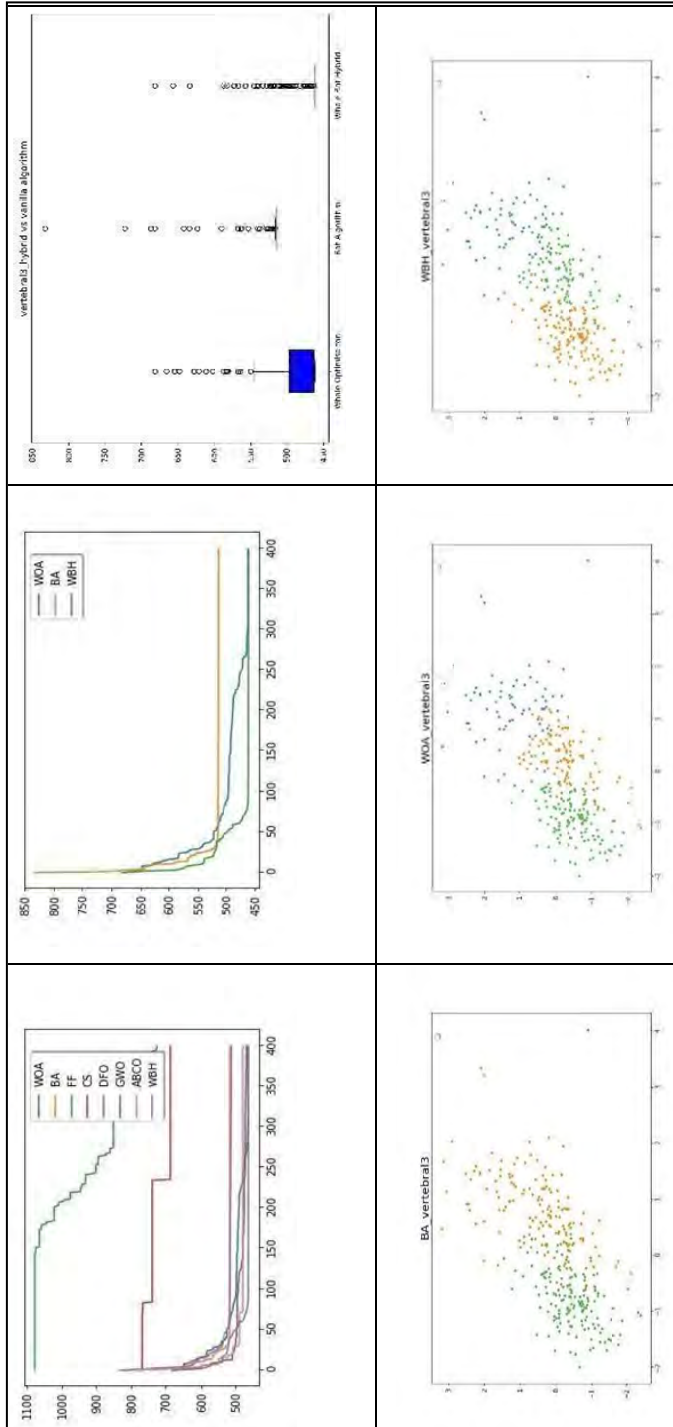
Note: No. of clusters: 3.

Figure 37 Vertebral II dataset results (see online version for colours)



Note: No. of clusters: 2.

Figure 38 Vertebral III dataset results (see online version for colours)



Note: No. of clusters: 3.

Figure 39 Credit card dataset results (see online version for colours)

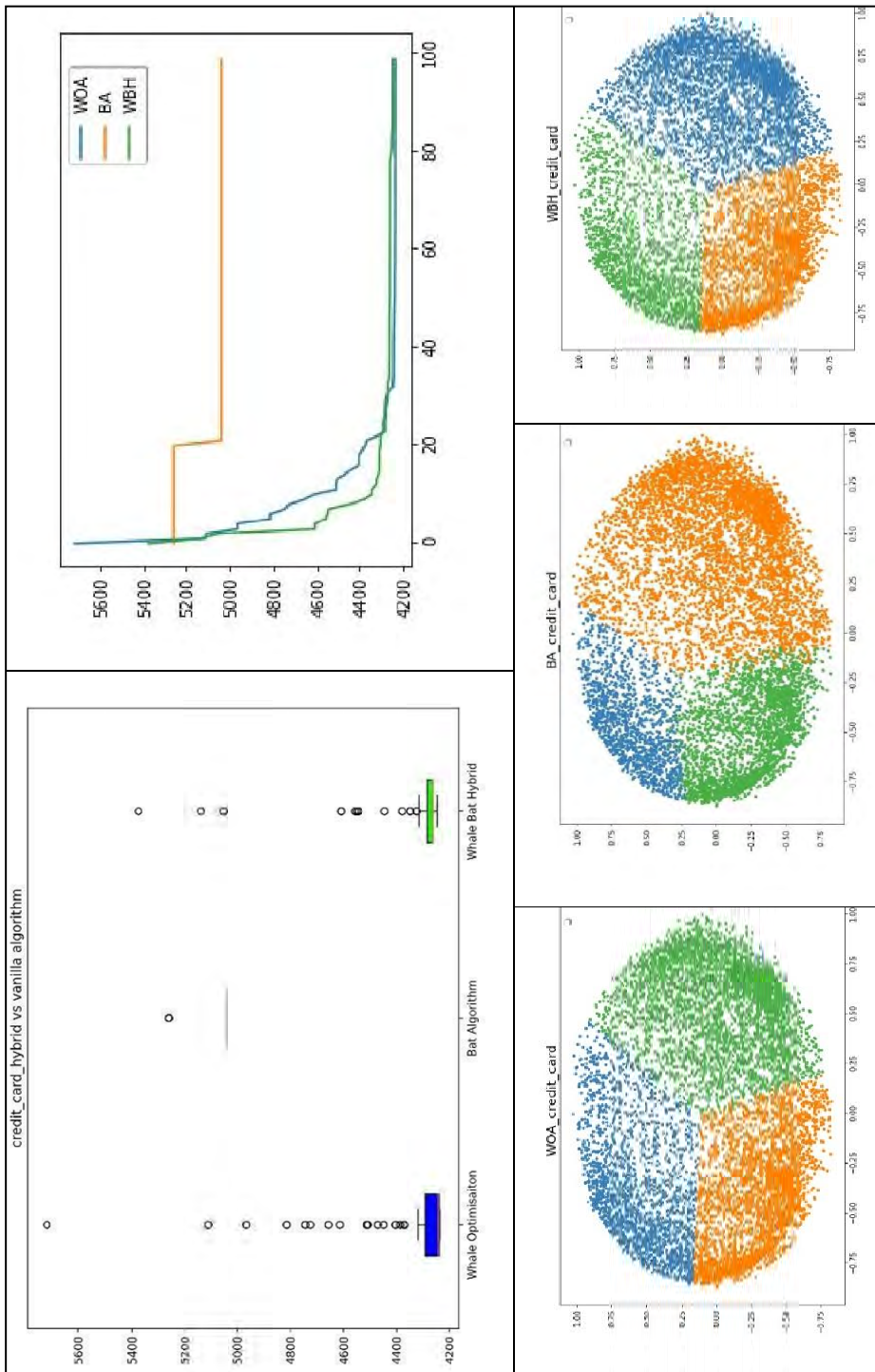
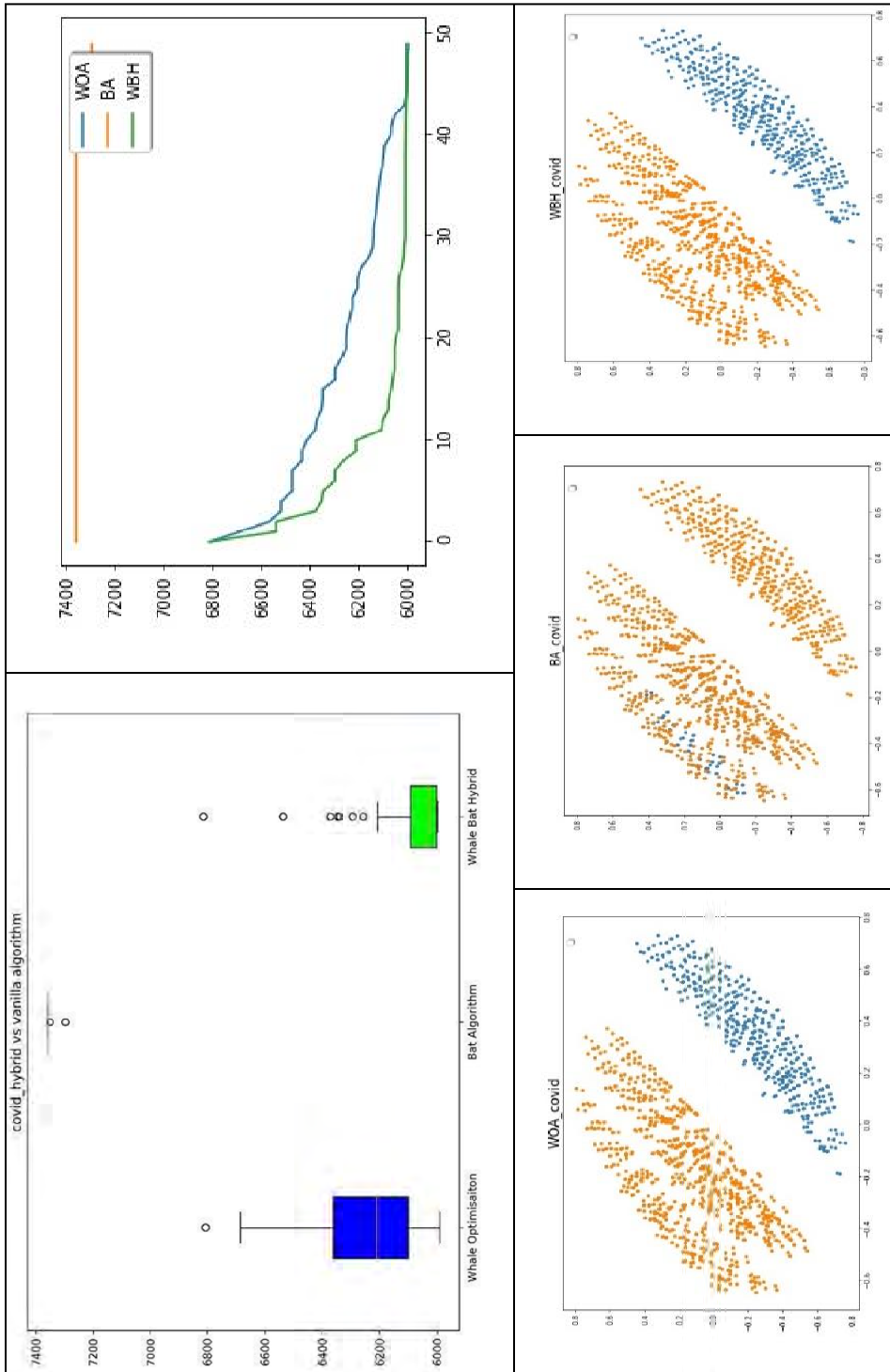


Figure 40 COVID-19 diagnosis dataset results (see online version for colours)



5 Conclusions

From the results collected after testing the algorithms on 14 CEC functions, 20 labelled datasets, and two unlabelled datasets, we can conclude that the hybrid performs comparatively better than the default algorithms, for 90% of the tests, and in quite a few cases, better than all other vanilla algorithms. There are a few scenarios where the hybrid fails to perform as good or better than the default versions. Most of such cases are due to the hybrid imbibing the shortcomings of the default algorithms, hence if either of the vanilla bat or whale algorithm perform below par on a dataset, it results in the hybrid not performing to the best of its abilities. But more often than not, the hybrid is able to converge faster and in times, to a more accurate minimum, when compared to its vanilla counterparts.

The hybrid algorithm is noteworthy for several use cases. Consider an environment where the clustering result is to be obtained with budget constraints that allow only a few iterations, possibly due to resource limitations. The hybrid can converge to the minima faster than the default algorithms, leading to a more accurate result within fewer iterations. The hybrid also decreases the probability of fixating on local minima instead of the global minima, which is an issue in several existing algorithms.

The COD affects all clustering algorithms due to its vast number of features. But the hybrid tends to perform even better in cases with such high dimensionality. For example, in Figure 30 which refers to the ionosphere dataset, which consists of 60 features, it is noticed that the bat suffers the COD effect, but the hybrid is able to minimise to the global minima, even with such a high number of features. Further testing may be necessary to obtain the limit of the number of features after which the hybrid suffers the same effects.

References

- Agarwal, P. and Mehta, S. (2014) 'Nature-inspired algorithms: state-of-art, problems and prospects', *International Journal of Computer Applications*, Vol. 100, No. 14, pp.14–21 [online] <https://doi.org/10.5120/17593-8331>.
- Agarwal, P. and Mehta, S. (2015) 'Comparative analysis of nature inspired algorithms on data clustering', *2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)* [online] <https://doi.org/10.1109/icrcicn.2015.7434221>.
- Dhiman, G. and Kumar, V. (2017) 'Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications', *Advances in Engineering Software*, Vol. 114, pp.48–70 [online] <https://doi.org/10.1016/j.advengsoft.2017.05.014>.
- Eesa, A.S., Brifcani, A.M.A. and Orman, Z. (2013) 'Cuttlefish algorithm – a novel bio-inspired optimization algorithm', *International Journal of Scientific & Engineering Research*, Vol. 4, No. 9, pp.1978–1986.
- Ezugwu, A.E. (2020) 'Nature-inspired metaheuristic techniques for automatic clustering: a survey and performance study', *SN Applied Sciences*, Vol. 2, No. 2 [online] <https://doi.org/10.1007/s42452-020-2073-0>.
- Ezugwu, A.E. et al. (2020) 'Automatic clustering algorithms: a systematic review and bibliometric analysis of relevant literature', *Neural Computing and Applications*, Vol. 33, No. 11, pp.6247–6306 [online] <https://doi.org/10.1007/s00521-020-05395-4>.

- José-García, A. and Gómez-Flores, W. (2016) 'Automatic clustering using nature-inspired metaheuristics: a survey', *Applied Soft Computing*, Vol. 41, pp.192–213 [online] <https://doi.org/10.1016/j.asoc.2015.12.001>.
- Martínez-Álvarez, F. et al. (2020) 'Coronavirus optimization algorithm: a bioinspired metaheuristic based on the COVID-19 propagation model', *Big Data*, Vol. 8, No. 4, pp.308–322 [online] <https://doi.org/10.1089/big.2020.0051>.
- Mehta, S. and Agarwal, P. (2018) 'ABC_DE_FP: a novel hybrid algorithm for complex continuous optimization problems', *International Journal of Bio-Inspired Computation*, Vol. 1, No. 1, p.1 [online] <https://doi.org/10.1504/ijbic.2018.10014476>.
- Mirjalili, S. and Lewis, A. (2016) 'The whale optimization algorithm', *Advances in Engineering Software*, Vol. 95, pp.51–67 [online] <https://doi.org/10.1016/j.advengsoft.2016.01.008>.
- Nanda, S.J. and Panda, G. (2014) 'A survey on nature inspired metaheuristic algorithms for partitional clustering', *Swarm and Evolutionary Computation*, Vol. 16, pp.1–18 [online] <https://doi.org/10.1016/j.swevo.2013.11.003>.
- Oduntan, O.I. and Thulasiraman, P. (2018) 'Hybrid metaheuristic algorithm for clustering', *2018 IEEE Symposium Series on Computational Intelligence (SSCI)* [online] <https://doi.org/10.1109/ssci.2018.8628863>.
- Singh, U. and Salgotra, R. (2017) 'Pattern synthesis of linear antenna arrays using enhanced flower pollination algorithm', *International Journal of Antennas and Propagation*, Vol. 2017, pp.1–11 [online] <https://doi.org/10.1155/2017/7158752>.
- Soto, R. et al. (2020) 'Advances in recent nature-inspired algorithms for neural engineering', *Computational Intelligence and Neuroscience*, Vol. 2020, pp.1–2 [online] <https://doi.org/10.1155/2020/7836239>.
- Tzanetos, A. and Dounias, G. (2020) 'A comprehensive survey on the applications of swarm intelligence and bio-inspired evolutionary strategies', *Learning and Analytics in Intelligent Systems*, pp.337–378 [online] https://doi.org/10.1007/978-3-030-49724-8_15.
- Yang, X.S. and He, X. (2013) 'Bat algorithm: literature review and applications', *International Journal of Bio-Inspired Computation*, Vol. 5, No. 3, p.141 [online] <https://doi.org/10.1504/ijbic.2013.055093>.