International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Nature Inspired Metaheuristicbased Optimization

Raj Kumar Gautam[1], Vaibhav Garg[2], Ankush Bhagat[3]

[1-3] *B.Tech Students, Information Technology & Maharaja Agrasen Institute of Technolgy,Delhi*

***Abstract*—This paper is a comprehensive study on nature inspired Hyperparameter optimization, with a distinct focus on Honey Badger Algorithm, along with Aquila Optimizer algorithm. The study involves in-depth analysis of the above algorithms, their weaknesses and strengths and comparing them with the theoretical advantages. The implementation of these algorithms, This paper demonstrate the promise of these algorithms on optimization of Hyperparameters like learning rate, number of hidden layers for our various datasets. The findings of this paper show that HBA and Aquila Optimization algorithms offer potential alternatives to the existing approaches, providing more effective and efficient solutions for hyperparameter optimization. This paper contributes to ongoing discourse on the place of nature inspired algorithms and their place in solutions to unconventional places**

I. Introduction

A multitude of hyperparameters must be specified prior to the execution of machine learning (ML) algorithms, including gradient boosting, random forest, and neural networks for regression and classification. These factors, which have a big influence on the ML model's accuracy and precision, are not discovered during training. The model's hyperparameters need to be optimized for optimal performance.

The process of optimizing involves determining the ideal configurations for certain entities while adhering to preset limits and utilizing limited resources. Historically, a variety of techniques, including gradient descent and linear programming, have been used to carry out this procedure. Nevertheless, these approaches are not suitable for optimizing hyperparameters since they are unable to address NP-hard issues and necessitate extensive mathematical derivation, particularly for gradient-based techniques that are prone to get trapped in local optima. Nevertheless, by drawing inspiration from nature, more effective strategies can be developed. Algorithms inspired by nature capitalize on millions of years of evolution and optimization through trial and error. The actions and procedures seen in nature serve as the foundation for these algorithms.

These methods are already in use to solve various optimization problems. For instance, computer image processing has made use of the Bees algorithm, which is based on the natural process of bee foraging. Multiple times, these algorithms have demonstrated their ability to determine the ideal value for a given parameter. There are numerous algorithms inspired by nature. The No Free Lunch (NFL) theorem states that not all optimization issues can be solved by the optimization algorithm in an effective manner.

Through this paper explore the Gravitational Search algorithm , GGSA, SSA and how the optimisation offered by these algorithms can be used to refine the Hyperparameters for other Machine Learning Models.

II. Literature Review

This paper aware that many machine learning models perform best when hyperparameters are optimized. However, a variety of factors, including the large and intricate search spaces, might make choosing the ideal parameters an extremely difficult task.

These were traditionally mostly discovered by trial and error, with the use of techniques like grid search and random search later on. However, these techniques may be costly to compute and time-consuming, particularly for complex models and huge datasets. As a result, this paper began looking for algorithms to make the calculation of these hyperparameters more effective. This investigation led to the discovery of nature-inspired algorithms, which imitate natural occurrences and processes and aid in the effective navigation of the search space.

A. Review of Existing Nature-inspired Algorithms

1) Gravitational Search Algorithm (GSA)

The Gravitational Search Algorithm (GSA), a nature inspired optimization algorithm based on the mathematical model of Newton's Law of Gravitation and motion [1][2]. Based on the law of gravitational forces between two bodies. It has demonstrated its effectiveness in hyperparameter optimization tasks.

The universal force that holds two objects together is called gravity. It aids in keeping us anchored to Earth's surface. Therefore, GSA is predicated on the same idea, emphasizing how, in their pursuit of the lowest energy state (ideal solution), heavier objects (better solution) attract lighter ones (less optimal solution[2].

Now let us see what Newton's Law of Gravitation is and how it is connected to the algorithm's core mechanism.

Consider the optimization as our universe, where many solutions with varying masses (indicating their fitness) dance throughout a higher-dimensional space in search of the lowest energy states, which correspond to the best answers. Our Gravitational Search Algorithm (GSA) uses Newton's Law of Gravitation, our favorite celestial choreographer, to create this dance of various celestial bodies in space[3].

Just like there are planets that revolve around the massive stars or like there are massive galaxies that revolve around the super massive black holes, in GSA solutions with higher "masses" - fitness or goodness - exert a stronger gravitational pull on the lighter counterparts. This pull guides our solutions, towards the promising regions - called optimal search spaces - providing optimal solutions [3]. But GSA is not based on some predefined rigid planetary orbits, But instead allows for the dynamic interaction allowing solutions to improve their "masses" through optimization, causing their pull to intensify, drawing other less optimal solutions to the potential optima causing an iterative loop of changing gravitational forces, ensuring continuous refinement and movement towards the desired solution [3].

However, whereas solutions often dance towards global optimality in this cosmic space of the gravitational search space, there is also a dark route: the path of the seductive allure of local maxima. Similar to heavenly mirages, these peaks have the ability to ensnare unsuspecting agents, compelling them to orbit about the local maxima and producing the appearance of ideal solutions. This paper examine the algorithm itself in order to comprehend this.

This problem is mostly caused by the solutions' reliance on "masses," or fitness values, to lead them through this difficult search area. This technique can produce gravitational wells around local maxima, but it also guarantees that promising solutions will eventually attract one another. The agents that are attracted to these optima wind up in an infinite circle around them since their masses are "insufficient" to break free of this grasp [4].

This was researched and explained by Rasoul Eskandar and Seyed Vahid Pourbabakhani [4] highlighting this limitation, and a need for a mechanism to stop this premature convergence [4].

Because the technique is iterative, there is one more constraint of GSA. Like cosmic moths lured to these flickering stars - heavy agents - the heavier agents cause the other agents to orbit around them incessantly by creating their own gravitational sphere.

Furthermore, solely relying on the forces of gravitation, GSA lacks diversity. Thus limiting its ability to escape these local maxima [5]. While some algorithms incorporate random mutations or perturbations, Agents in GSA follow a predetermined path, failing to escape these local maxima and their region [5]. This was Researched by James Kennedy and Russell Ebarhart on Particle Swarm Optimization, a different nature-inspired algorithm, highlighting the importance of incorporating these exploration-enhancing mechanisms [5].

GSA operations are more complex in nature, which can make the algorithm difficult to implement and understand. Moreover, GSA, having a poor convergence rate, takes a lot of time to find the optimal solution [6].

2) GBest-Guided Gravitational Search Algorithm

"Gbest-guided GSA" or "GGSA" refers to a modified version of the Gravitational Search Algorithm, introducing the concept of "Gbest" guidance [7].

Now, this paper know what GSA is and how it works. And here's what "Gbest" adds to this mix:

- Gbest: It stands for "global best", representing the best solution found so far in the population [7].
- Gbest guidance: Meaning that this "Gbest" solution will also participate in these small interactions. Thus, ending up influencing the other smaller agents leading them to the optimal solution [7].

Drawbacks of GGSA

- Premature Convergence: If the guidance becomes too strong, it can lead to premature convergence, causing potentially better solutions to miss out.
- Loss of Diversity: Just Focusing on the Gbest can reduce the diversity, hindering the exploration of new areas.

Overall, GGSA has shown to provide better convergence speed and performance compared to GSA and its variants [7].

3) Symbiotic Search Algorithm

It is also known as the Symbiotic Organisms Search Algorithm. The Symbiotic Search Algorithm (SSA) as stated in its name, is an algorithm based on the mutually beneficial relationship observed in ecosystems. It provides us with a unique approach for solving these complex search spaces.

Chen and Prayogo first introduced SSA, a new nature-inspired Algorithm in 2014 [8] based on the various symbiotic relationships found in our ecosystem. This algorithm uses the concept of "symbiosis", derived from the Greek word meaning "Living Together" [9]. Chen and Prayogo, in their research, compared the performance of SSA against other well-known algorithms and discussed its characteristics [9].

One big benefit of SSA is that this Algorithm is parameter-free. Thus, there is no need to adjust any parameter and only define the population and number of generations [8].

Now, SSA typically involves a population of individuals representing a potential solution. These individuals are left to interact with each other to undergo symbiotic relationship with each other, resulting in individuals modifying their position, which is guided by a predefined fitness function that is there to evaluate the quality of these solutions in the search space [9].

While SSA with its unique approach promises optimal solutions, there are drawbacks. This paper read that there are no parameters other than population size and number of generations, but this also causes us a lot of trouble. This paper have to carefully choose these parameters, as choosing inappropriate values can lead to slower convergence rates, premature stagnation, or even divergence. Although being a simpler algorithm, SSA mainly depends on the population size, making it computationally expensive. This is mainly noticeable when using large population values with complex fitness functions. Thus, for simpler computations or situations where faster computational resources are not available, simpler alternatives are preferred.

4) Cuckoo Search

The cuckoo search algorithm is a metaheuristic optimization algorithm introduced by Gandomi, Yang, and Alavi in 2011[10].It is designed to solve complex optimization problems, particularly in structural engineering. The algorithm is based on the behavior of cuckoo birds, which lay their eggs in the nests of other bird species. In the CS algorithm, a population of "host nests" represents potential solutions to the optimization problem. Cuckoos randomly lay eggs in these nests, which represent new candidate solutions. The quality of each solution is evaluated, and the best solutions are kept while the worst ones are abandoned. The algorithm also uses Levy flights, a type of random walk, to generate new solutions.

B. Proposed Approaches

1) Honey Badger Algorithm

The Honey Badger algorithm (HBA) is a bio-inspired metaheuristic optimization technique that mimics the foraging behavior of honey badgers. It exhibits a unique balance between exploration (searching new areas) and exploitation (intensifying search in promising regions), making it suitable for solving complex optimization problems. This paper delves into the core principles, pseudocode, and potential applications of HBA, along with a discussion on its strengths and limitations [11].

The Honey Badger Algorithm (HBA), which takes its cues from the natural foraging habits of honey badgers, presents a revolutionary metaheuristic approach to optimization. Stochastic optimization methods are more common because they can explore complex problem spaces more effectively than traditional deterministic optimization techniques, which frequently suffer from local optima trapping [11][12][13]. Among stochastic optimization techniques, HBA is distinguished by providing a special combination of exploitation and exploration strategies that resemble the honey badgers' adaptive feeding habits [13].

Furthermore, metaheuristic algorithms such as HBA can be used to a broad variety of optimization problems in different fields because to their flexibility and gradient-free nature [13]. Because of their flexibility and capacity to escape local optima traps, metaheuristic approaches are well-suited to tackle contemporary optimization problems [12]. As researchers continue to explore and refine metaheuristic algorithms like HBA, their potential for revolutionizing optimization tasks in diverse fields becomes increasingly evident [11].

In search spaces, this algorithm seeks to balance exploitation with exploration, imitating the adaptive techniques used by honey badgers for food. The "digging phase" and the "honey phase" are the two separate stages in which HBA functions [12]. During the digging phase, HBA utilizes its sensing capabilities to approximate the location of potential solutions, akin to how a honey badger sniffs out prey before digging [12]. Once in the vicinity of a promising area, the algorithm explores the surroundings to identify the optimal spot for further investigation, mirroring the meticulous approach of a honey badger preparing to catch its prey [12].

HBA adopts a more focused search approach during the honey phase, similar to a honey badger tracking a honeyguide bird to find a beehive. In order to focus on potential areas of the search space and direct the search towards ideal solutions, this phase emphasizes the use of known information [14].

Throughout the optimization process, HBA seeks to retain population variety, efficiently navigate complex search spaces, and steer clear of suboptimal regions by combining these exploration and exploitation tactics [14]. Experiments on benchmark functions and real-world engineering design issues illustrate the success of the algorithm in tackling a wide range of optimization problems, which may be attributed to its dynamic search behavior and controlled randomization techniques.

2) Aquila Optimizer

The Aquila Optimizer (AO) algorithm, drawing inspiration from the hunting strategies of the Aquila bird, has been extensively evaluated against other stochastic optimization techniques. Through a series of experiments on diverse test functions and real-world engineering problems, the effectiveness and superiority of the AO algorithm have been showcased [15]. The comparisons with established meta-heuristic methods have consistently demonstrated the competitive edge of AO in terms of performance and efficiency [15]. By simulating the hunting behaviors of the Aquila bird, AO introduces a unique approach to optimization that balances exploration and exploitation phases effectively [16]. This innovative algorithm's ability to find optimal solutions for various optimization challenges positions it as a promising contender among stochastic optimization techniques [17]. The rigorous validation and demonstrated superiority of the AO algorithm underscore its potential to address complex optimization problems with efficiency and effectiveness, making it a valuable addition to the realm of meta-heuristic optimization algorithms.

The Aquila Optimizer (AO) algorithm replicates the hunting tactics of the Aquila bird by dividing its optimization procedures into four methods that imitate the bird's hunting maneuvers [17]. These methods involve selecting the search space using high soaring and vertical stoop, exploring diverse search spaces through contour flight and short glide attack, exploiting convergent search spaces with low flight and slow descent attack, and swooping in for the final capture. By

simulating the strategic movements of the Aquila bird during hunting, the AO algorithm effectively manages the exploration and exploitation phases of optimization. This nature-inspired strategy enables the algorithm to navigate search spaces adeptly, adjust to different environments, and optimize solutions efficiently, mirroring the agility and precision observed in the Aquila bird's hunting behaviors[15][16].

III. METHODOLOGY

The proposed solution or methodology for this study involves the application of the Honey badger algorithm(HBA) and Aquila optimizer(AO) for hyperparameter optimization in machine learning models. Here's a general outline of the proposed methodology:

A. Understanding HBA and AO

The first step is to gain a thorough understanding of the HBA and AO. This involves studying the principles behind these algorithms, how they work, and how they can be applied to hyperparameter optimization. This understanding is crucial as it provides insights into their mechanisms, which can then be leveraged to effectively apply them for hyperparameter optimization.

B. Data Preparation

The next step is to prepare the data for the machine learning models. This involves data cleaning, preprocessing, and splitting the data into training and testing sets. The quality of the data and how it is prepared can significantly impact the performance of the models, making this an important step in the process.

C. Model Building

Once the data is prepared, the machine learning models are built using mathematical model. The models would initially be built with default hyperparameters. This provides a baseline performance, which can then be improved through hyperparameter optimization.

IV. RESULT

A. Implementation

1) Honey Badger Algorithm

The Honey Badger Algorithm (HBA) mimics the foraging behavior of honey badgers to tackle optimization problems. It begins by creating a population of candidate solutions, akin to honey badgers spread across the search space. Each solution's fitness is evaluated based on the problem's objective function. HBA utilizes a concept called "foraging memory." This memory is updated for each solution based on its fitness, allowing the honey badgers to remember promising areas. Subsequently, the solutions move within the search space. This movement incorporates a bias towards these remembered locations, promoting exploitation of high-fitness regions. However, a random component is also included to ensure some exploration of new areas. Periodically, selection pressure is applied. Fitter solutions, representing successful foragers, are chosen to form the next generation, while new solutions might be introduced to maintain diversity within the population. This cycle of evaluation, movement, selection, and potential diversification continues until a stopping criterion is met, such as reaching a maximum number of iterations or achieving a desired fitness level. Finally, the best solution discovered by the "honey badgers" during their foraging endeavors is returned.

2) Aquila Optimizer

The Aquila Optimizer (AO) draws inspiration from the hunting behavior of eagles. It maintains a population of search agents, metaphorically representing eagles (aquilae), which are randomly distributed within the search space. Each eagle is assigned a position, velocity, and step size. AO employs the Levy Flight foraging strategy, mimicking the way eagles search for prey. This strategy guides the update of each eagle's position based on its current location, velocity, and a function incorporating random walks with large jumps. Additionally, eagle velocities are periodically updated considering their current velocity, the best solution found so far in the population, and a random component. The step sizes are also dynamically adjusted based on the distance of each eagle from the best solution, simulating the eagles' focus as they approach their target. To improve the population, AO utilizes a selection and replacement mechanism. Each eagle's fitness is compared with the worst solution in the population. If an eagle performs better, it replaces the worse solution, gradually enhancing the overall population quality. This process of position update, velocity and step size adjustment, selection, and replacement continues until a stopping criterion, such as a maximum number of iterations or desired fitness, is met. Finally, the solution with the highest fitness value encountered during the search becomes the output of the algorithm.

B. Final Result

This section presents the key findings obtained from applying the Honey Badger Algorithm (HBA) and Aquila Optimizer to a set of benchmark optimization problems. This paper evaluated Their performance using metrics such as accuracy score, number of layers, and learning rate.

1) Honey Badger Algorithm

a) Accuracy Score

This segment presents the exactness scores of the ML algorithm streamlined utilizing the Honey badger algorithm(HBA).

By optimising the hyperparameters this paper try to maximize the Accuracy score of algorithm for different datasets. Closeness to 1.0 represents higher accuracy and fartherness from 1.0 represent lower accuracy(with their respective value as shown).

2) Aquila Optimizer

a) Acc

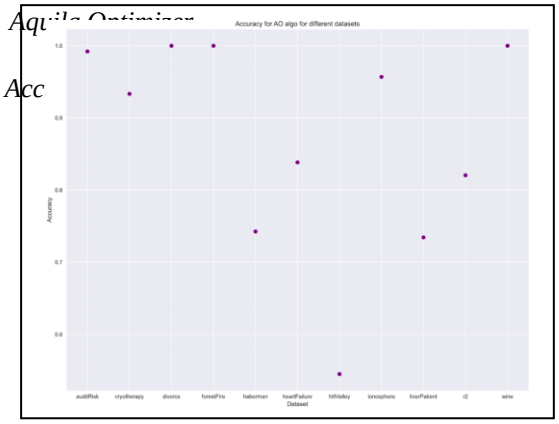

Fig. 1. This figure shows the Accuracy Score of HBA

b) Learning Rate

This paper finding optimized Learning rate for maximizing the accuracy score of Honey badger algorithm(HBA)for different datasets. The Learning rate controls how much the model learns from the information or data at each step of the growing experience

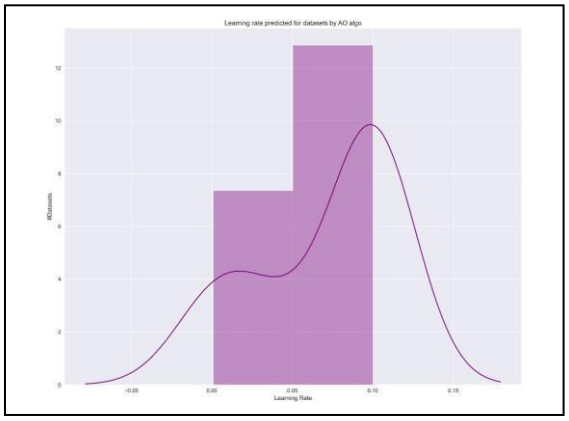

Fig. 2. This figure shows the Learning Rate of HBA

c) Number Of Layers

This segment talks about the optimized number of hidden layers as determined by the Honey badger algorithm(HBA). This paper are finding optimized number of hidden layers for maximizing the accuracy score of Slime mould algorithm for different datasets

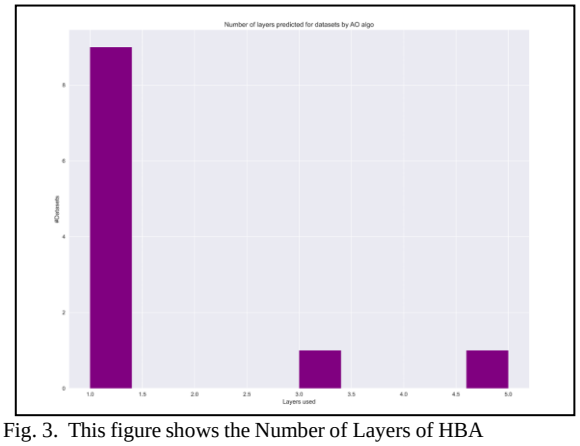

Fig. 3. This figure shows the Number of Layers of HBA

This segment presents the exactness scores of the ML algorithm streamlined utilizing the Aquila optimizer(AO). By optimising the hyperparameters This paper try to maximize the Accuracy score of algorithm for different datasets. Closeness to 1.0 represents higher accuracy and fartherness from 1.0 represent lower accuracy(with their respective value as shown).

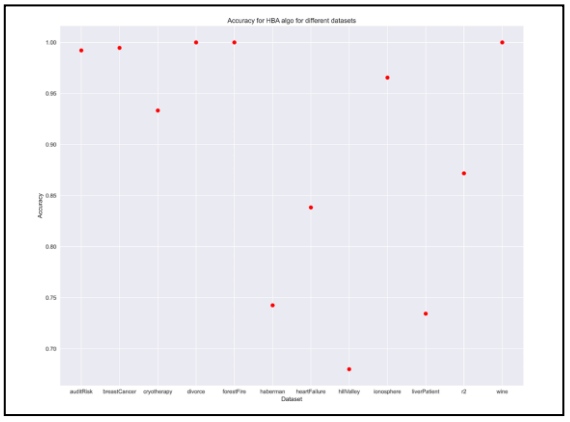

Fig. 4. This figure shows the Accuracy Score of AO

b) Learning Rate

This segment talks about the optimized number of hidden layers as determined by the Aquila optimizer(AO). This paper are finding optimized number of hidden layers for maximizing the accuracy score of Aquila Optimization for different datasets.

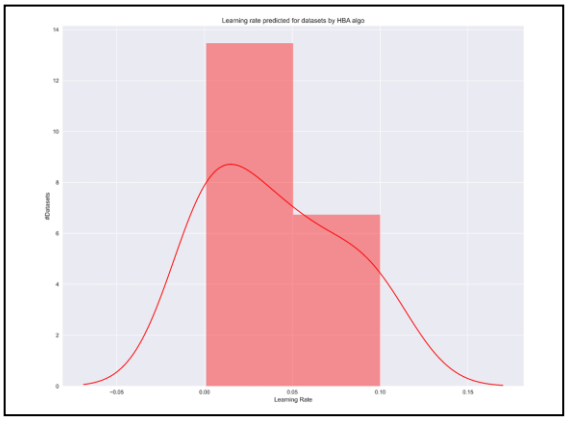

Fig. 5. This figure shows the Learning Rate of AO

c) Number Of Layers

This paper finding optimized Learning rate for maximizing the accuracy score of Aquila optimizer(AO)for different datasets. The Learning rate controls how much the model learns from the information or data at each step of the growing experience

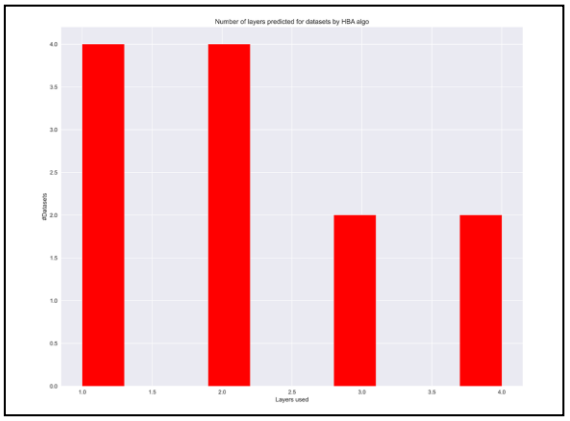

Fig. 6. This figure shows the Number of Layers of AO

V. CONCLUSION

The implementation of nature-inspired hyperparameter optimization algorithms such as the Aquila Optimizer and the Honey Badger Algorithm represents a significant advancement in the field of machine learning. These algorithms, inspired by the foraging behavior of aquila and the intelligent problem-solving skills of honey badgers, have demonstrated the potential to enhance the efficiency and accuracy of hyperparameter tuning processes. The project's findings suggest that these algorithms can outperform traditional optimization methods, particularly in complex search spaces with multiple local optima. By mimicking natural strategies, they offer a robust and adaptive approach to navigating the hyperparameter landscape, leading to improved model performance. The successful application of these algorithms could pave the way for more intuitive and effective machine learning models, reflecting the growing trend of bio-inspired computing in solving intricate computational problems.

REFERENCES

[1] Mittal, H., Tripathi, A., Pandey, A. C., & Pal, R. (2020). Gravitational search algorithm: a comprehensive analysis of recent variants. Multimedia Tools and Applications, 80(5), 7581-7608. https://doi.org/10.1007/s11042-020-09831-4

[2] Zandevakili, H., Rashedi, E., & Mahani, A. (2017). Gravitational search algorithm with both attractive and repulsive forces. Soft Computing, 23(3), 783- 825. https://doi.org/10.1007/s00500-017-2785-2

[3] Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009, June). GSA: A Gravitational Search Algorithm. *Information Sciences*, *179*(13), 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004

[4] Rasoul Eskandar, R., & Pourbabakhani, S. V. (2010). Gravitational Search Algorithm - A New Metaheuristic Optimization Algorithm. Journal of Advances in Computer Science and Technology, 3(4), 229-245.

[5] Kennedy, J., & Eberhart, R. C. (2001). Swarm Intelligence. Morgan Kaufmann Publishers.

[6] Mohakud, R., & Dash, R. (2020). Survey on Hyperparameter Optimization using Nature-Inspired Algorithm of Deep ConvolutionNeural Network. In Smart innovation, systems and technologies (pp. 737-744). https://doi.org/10.1007/978- 981-15-5971-6 77

[7] "Hybrid adaptive 'gbest'-guided gravitational search and pattern search algorithm for automatic generation control of multi-area power system" by Khadanga et al. (2017)

[8] Ezugwu, A. E., & Prayogo, D. (2019, April). Symbiotic organisms search algorithm: Theory, recent advances and applications. *Expert Systems With Applications*, *119*, 184–209. https://doi.org/10.1016/j.eswa.2018.10.045

[9] Cheng, Min-Yuan & Prayogo, Doddy. (2014). Symbiotic Organisms Search: A new metaheuristic optimization algorithm. Computers & Structures. 139. 10.1016/j.compstruc.2014.03.007.

[10] Gandomi, A.H., Yang, XS. & Alavi, A.H. Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with Computers* **29**, 17–35 (2013). https://doi.org/10.1007/s00366-011-0241-y

[11] Hashim, F. A., Houssein, E. H., Hussain, K., Mabrouk, M. S., & Al-Atabany, W. (2022, February). Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems. *Mathematics and Computers in Simulation*, *192*, 84–110. https://doi.org/10.1016/j.matcom.2021.08.013

[12] Düzenli , T., Kutlu Onay, F., & Aydemi r, S. B. (2022, October). Improved honey badger algorithms for parameter extraction in photovoltaic models. *Optik*, *268*, 169731. https://doi.org/10.1016/j.ijleo.2022.169731

[13] Hu, G., Zhong, J., & Wei, G. (2023, August). SaCHBA_PDN: Modified honey badger algorithm with multi-strategy for UAV path planning. *Expert Systems With Applications*, *223*, 119941. https://doi.org/10.1016/j.eswa.2023.119941

[14] Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems Fatma A. Hashima, Essam H. Housseinb, Kashif Hussainc,∗, Mai S. Mabroukd, Walid Al-Atabany

[15] Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A., & Gandomi, A. H. (2021, July). Aquila Optimizer: A novel meta-heuristic optimization algorithm. *Computers & Industrial Engineering*, *157*, 107250. https://doi.org/10.1016/j.cie.2021.107250

[16] J. Zhao, Z. -M. Gao and H. -F. Chen, "The Simplified Aquila Optimization Algorithm," in IEEE Access, vol. 10, pp. 22487-22515, 2022, doi: 10.1109/ACCESS.2022.3153727.

[17] Wang, S., Jia, H., Abualigah, L., Liu, Q., & Zheng, R. (2021, August 30). An Improved Hybrid Aquila Optimizer and Harris Hawks Algorithm for Solving Industrial Engineering Optimization Problems. *Processes*, *9*(9), 1551. https://doi.org/10.3390/pr909