

## Journal Pre-proof

AIEOU: Automata-based improved equilibrium optimizer with U-shaped transfer function for feature selection

Shameem Ahmed, Kushal Kanti Ghosh, Seyedali Mirjalili, Ram Sarkar

PII: S0950-7051(21)00545-1

DOI: <https://doi.org/10.1016/j.knosys.2021.107283>

Reference: KNOSYS 107283

To appear in: *Knowledge-Based Systems*

Received date : 30 September 2020

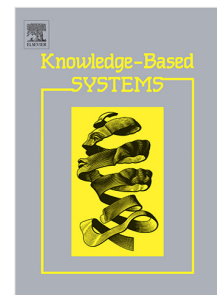
Revised date : 13 April 2021

Accepted date : 2 July 2021

Please cite this article as: S. Ahmed, K.K. Ghosh, S. Mirjalili et al., AIEOU: Automata-based improved equilibrium optimizer with U-shaped transfer function for feature selection, *Knowledge-Based Systems* (2021), doi: <https://doi.org/10.1016/j.knosys.2021.107283>.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2021 Elsevier B.V. All rights reserved.



# AIEOU: Automata-based Improved Equilibrium Optimizer with U-shaped Transfer function for Feature Selection

Shameem Ahmed<sup>a</sup>, Kushal Kanti Ghosh<sup>a</sup>, Seyedali Mirjalili<sup>b,c,d,\*</sup>, Ram Sarkar<sup>a</sup>

<sup>a</sup>Department of Computer Science and Engineering, Jadavpur University, Kolkata 700032, India

<sup>b</sup>Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Fortitude Valley, Brisbane, 4006 QLD, Australia

<sup>c</sup>YFL (Yonsei Frontier Lab), Yonsei University, Seoul, Korea

<sup>d</sup>King Abdulaziz University, Jeddah, Saudi Arabia

## Abstract

The high dimension of any dataset has become an unavoidable challenge in Data Science and Machine Learning. Reducing the number of dimensions by excluding noisy, irrelevant, or correlated information is often referred to as the feature selection (FS). The ultimate goal in FS is to identify an optimal set of dimensions (features) of any dataset to develop an efficient learning model, decrease the computational time and optimize the memory requirement with the help of some methods. Recently, optimization algorithms have gained popularity in different fields because of their flexibility and effectiveness. Equilibrium optimizer (EO) is a physics-based meta-heuristic algorithm, which is inspired from a well-mixed dynamic mass balance on a control volume that has good exploration and exploitation capabilities. In this work, an improved version of EO is proposed with the inclusion of learning based automata to find proper values of its parameters and Adaptive  $\beta$  Hill Climbing (A $\beta$ HC) to find a better equilibrium pool. The method is used as a feature selector, evaluated on 18 standard UCI datasets with the help of K-nearest neighbors (KNN) classifier, and compared with eight state-of-the-art methods including classical and hybrid meta-heuristic algorithms. Moreover, the proposed method is applied on high dimensional Microarray datasets which generally contain a few samples but a large number of features, and often lead to a 'curse of dimensionality'. The obtained results illustrate the supremacy of the proposed method over the other state-of-the-art methods mentioned in the literature. The source code of this work is available at [https://github.com/ahmed-shameem/Feature\\_selection](https://github.com/ahmed-shameem/Feature_selection).

**Keywords:** Meta-heuristic, Optimization, Equilibrium Optimizer, Learning Automata, Adaptive  $\beta$ -Hill Climbing, Feature Selection

\*Corresponding author

Email addresses: [shameemahmed20apr2000@gmail.com](mailto:shameemahmed20apr2000@gmail.com) (Shameem Ahmed), [kushalkanti1999@gmail.com](mailto:kushalkanti1999@gmail.com) (Kushal Kanti Ghosh), [ali.mirjalili@gmail.com](mailto:ali.mirjalili@gmail.com) (Seyedali Mirjalili), [ramjucse@gmail.com](mailto:ramjucse@gmail.com) (Ram Sarkar)

URL: <https://orcid.org/0000-0003-1795-3361> (Shameem Ahmed), <https://orcid.org/0000-0003-0929-5928> (Kushal Kanti Ghosh), <https://orcid.org/0000-0002-1443-9458> (Seyedali Mirjalili), <https://orcid.org/0000-0001-8813-4086> (Ram Sarkar)

Preprint submitted to Elsevier

April 13, 2021

## 1. Introduction

*Supervised learning* algorithms are extremely popular in the sphere of machine learning, and used in a wide spectrum of fields due to their excellent predictive nature. Generally, a model is trained with a fixed-length feature vector, and the feature representations are linked with a class denomination. In real-world complex classification problems, many times, it is very hard to come up with a set of relevant features. Therefore, many candidate features are introduced to better narrate the domain to make it useful for the learning model. Unfortunately, it has been observed that many of these are irrelevant and redundant to the learning model. An irrelevant feature does not affect the learning model possibly in any way, rather a redundant feature does not serve anything new to the learning model (John et al., 1994). In many cases, the size of a dataset or the feature vector is so large that the learning model could not perform well in a real-time situation before removing these unwanted features. On the contrary, theoretically having additional features may have more perceptive power, the real world provides us many examples of why this is not generally the case.

Shrinking the number of redundant features drastically reduces the running time of a learning algorithm (Chandrashekar and Sahin, 2014; Dash and Liu, 1997), which helps in obtaining a better comprehension of the basal intricacies of a practical classification problem (Kohavi and Sommerfield, 2001; Koller and Sahami, 1996a). Feature selection (FS) methods try to choose a subset of features that are relevant to the learning model. This would eventually help reduce the training time of the learning model and storage requirement.

We can divide FS methods broadly into two categories based on the evaluation criteria of the features: Filter method and Wrapper method. Filter method performs some statistical measure or intrinsic properties of the dataset and provides a rank list of the features based on their relevance. This approach measures the relevance of features by their correlation with dependent variables. Few famous filter methods for feature selection are focus (Almuallim and Dietterich, 1991) and relief (Kira and Rendell, 1992). The focus performs a comprehensive search of all possible feature subsets to decide the least set of features which are relevant. But this compatible criterion makes focus very compassionate to noise and/or instabilities in the training set.

As the size of the power set of the features grows exponentially, focus becomes impractical for domains having larger dimension. In relief, rather than selecting a subset of features, each feature is given a suitable weighting which indicates its level of pertinence to the class label. However, this method is profitless at removing redundant and irrelevant features as two predictive but highly correlated features are both likely to be given high relevance weightings. Some other well known filter methods are chi-square (Zheng et al., 2004), mutual information (Koller and Sahami, 1996b), laplacian score (He et al., 2006) etc.

Wrapper method (Hall and Smith, 1999), on the other hand, employs a search throughout the feature space, and measuring the performance of the feature subsets using a machine learning algorithm indicating the goodness of the particular feature subset. Hence, FS process is being wrapped around a learning algorithm, so that the favoritism of the operators that define the search and that of the learning algorithm has been found. While these methods have shown desired success on several complex tasks, they are often excessively expensive to run based on terms of the computational time requirement. An important characteristic of FS is how well a method helps a learning algorithm in terms of achieved classification accuracy measure. Generally, wrapper method produces better results than filter methods (Ghosh et al., 2018).

Metaheuristics are high-level problem-independent techniques that can be applied to a broad range of problems. A metaheuristic knows nothing about the problem it will be applied to, it can treat functions (fitness function) as black boxes. Metaheuristics are derived from not only the insight into the problem structure or on the way in which an intelligent human would solve it but from a completely different range of processes. Due to the non-derivative nature, flexibility, and the ability to avoid local optima, meta-heuristics algorithms are very popular and used extensively in the field of optimization (Mirjalili et al., 2014). The effectiveness of every meta-heuristic algorithm is determined by fine-tuning between its exploration and exploitation capabilities (Ghosh et al., 2020a). Exploration is the ability to evaluate candidate solutions that are not neighbor to the current solution(s). This operation helps the algorithm to escape from a local optimum. Exploitation is when a search is done in the neighborhood of the current solution(s). It can be considered as a local

search. Tuning between these two parameters is very difficult because of the stochastic behavior of the optimization algorithms.

The rest of the paper is organized as follows: Section 2 discusses some past methods related to the optimization algorithms and FS methods, Section 3 presents the motivation and contribution of the present work, Section 4 provides a brief overview of EO and A $\beta$ HC algorithms. Section 5 provides a comprehensive description of the proposed FS method. The experimentation performed to validate the proposed method is reported in Section 6. Additionally, this section displays the achieved results obtained by our novel approach. In Section 7, the proposed method is compared with some state-of-the-art meta-heuristic and hybrid meta-heuristic FS methods. Section 8 displays the findings of the proposed method on high dimensional Microarray datasets. Finally, Section 9 draws the conclusive remarks of this work, discusses its restrictions and provides some possible future extension of this work.

## 2. Literature Survey

Due to its effectiveness and promising results in various fields, optimization algorithms have gained huge popularity over the years, (Xue et al., 2016). Naturally this popularity makes more researchers to bring many such interesting algorithms in the family of optimization algorithms. Meta-heuristic algorithms can be divided into several categories: single solution based and population based (Gendreau and Potvin, 2005), metaphor based and non-metaphor based (Abdel-Basset et al., 2018), nature inspired and non-nature inspired (Fister Jr et al., 2013) etc. Considering the ‘inspiration’ perspective, these algorithms can tentatively be divided into six groups (Molina et al., 2020): Breeding-based evolution, Swarm intelligence, Physics-based, Human inspired, Plants based and Miscellaneous. The Swarm intelligence category can be further divided into five different groups: Aquatic animals, Terrestrial animals, Flying animals, Microorganisms, and others.

Breeding-based evolutionary algorithms are inspired by the principles of natural evolution. These algorithms try to achieve optimality at every iteration based on the procedures of reproduction and fight for survival. Some of the popular algorithms are GA (Leardi, 1994), differential evolution (DE) (Storn and Price, 1997), evolution strategies (ES)

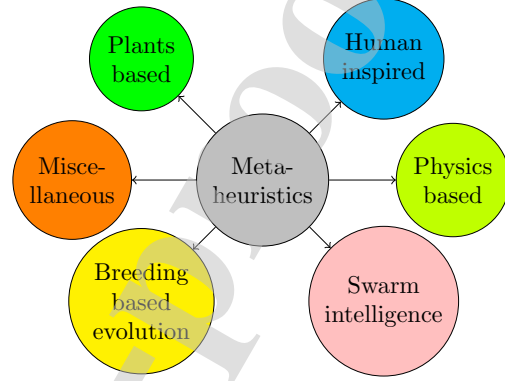


Figure 1: Showing categorization of meta-heuristic optimization algorithms based on their source of ‘inspiration’

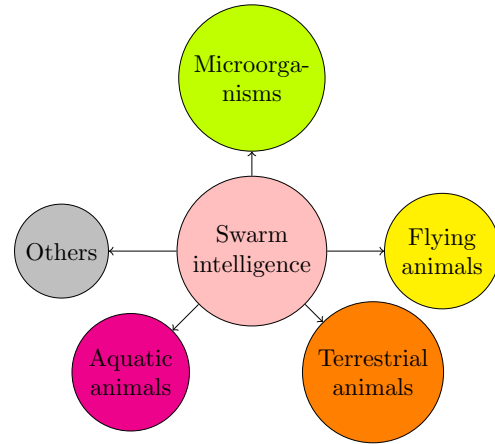


Figure 2: Showing categorization of Swarm Intelligence algorithms

(Beyer and Schwefel, 2002), coral reefs optimization (CRO) (Salcedo-Sanz et al., 2014) etc. The swarm intelligence-based algorithms mimic the collective behavior of a decentralized, self-organized system (insect colonies, bird flocks, etc.) either in nature or in artificial environments. This category can be further subdivided into five different groups:

- inspired from flying animals, e.g., PSO, (Kennedy and Eberhart, 1995; Xue et al., 2019; Zhang et al., 2017), artificial bee colony algorithm (ABC) (Karaboga and Basturk, 2007), cuckoo search algorithm (CS) (Yang and Deb, 2009) etc.
- inspired from terrestrial animals, e.g., GWO, (Mirjalili et al., 2014), ant colony optimization (ACO) (Dorigo et al., 1996) etc.
- inspired from aquatic animals, e.g., whale optimization algorithm (WOA), (Mirjalili and Lewis, 2016), SSA (Mirjalili et al., 2017) etc.
- inspired from microorganisms, e.g., slime mould algorithm (SMA), (Monismith and Mayfield, 2008), bacterial colony optimization (BCO) (Das et al., 2009) etc.
- others, e.g., weightless swarm algorithm (WSA), (Ting et al., 2012), prey-predator algorithm (PPA) (Tilahun and Ong, 2015) etc.

The next category is human-inspired algorithms. These algorithms are motivated by human social concepts like decision making and ideas related to the expansion of ideologies or competition among themselves. Some of them are soccer league competition algorithm (Moosavian and Roodsari, 2014), brain storm optimization algorithm (BSO) (Shi, 2011), imperialist competitive algorithm (ICA) (Atashpaz-Gargari and Lucas, 2007) etc. The plants' based algorithms are the group of all the algorithms whose search method for finding an optimal solution is inspired by plants. Some of the popular algorithms which belong to this category are forest optimization algorithm (FOA) (Ghaemi and Feizi-Derakhshi, 2014), artificial plant optimization algorithm (APO) (Cui and Cai, 2013) etc. The miscellaneous category contains algorithms that have diverse characteristics. There are very few such algorithms, one of them is the ying-yang pair optimization algorithm (YYPO) (Punathanam and Kotecha, 2016) which is inspired by Chinese mythology of yin and yang.

The work proposed here is based on Equilibrium optimizer, which belongs to the category of physics-based algorithms. Hence, we have discussed physics-based algorithms in detail.

The algorithms which come under this category are inspired by the behavior of physical or chemical events. These events can be electric charges, electromagnetism, gravitational force, chemical reaction or it can be even the behavior of water movements (in the context of physics). Some of the popular and widely used physics-based meta-heuristic algorithms are given in Table 1.

SA follows a probabilistic approach for estimating the global optimum of a function. It is a meta-heuristic to estimate global optimization in a large search space for an optimization problem. It is used to solve unconstrained and bound-constrained optimization problems. This method is inspired by annealing in the field of metallurgy, which is a process involving heating and controlled cooling of a material to expand the size of its underlying crystals and reduce their defects.

HS is a well-known meta-heuristic algorithm that mimics the improvisation of a musician to create a perfect harmony that is pleasant to the ears. It transforms the qualitative spontaneity process into a quantitative optimization course with various well-defined mandate and thus converts the charm and coherence of music into an answer for several optimization problems. To improve its exploration and exploitation phase and to create a proper tuning between them, HS uses harmony memory considering rate (HMCR) and pitch adjusting rate (PAR).

GSA is inspired by the Newtonian law of gravitation and laws of motion. This algorithm used the law of gravity and mass interactions. In GSA, the searching agents are considered to be an individual entity with a certain mass while every entity in the considered system interacts with each other with the help of gravitational force. The position of each entity presents a potential solution for the given problem, while the entity's mass is assigned using a fitness function. Simultaneously, the gravitational force among the entities causes the movement of all objects towards the sub-optimum solutions.

BH is inspired by the black hole phenomena, where in each iteration the best candidate is considered to be the black hole. As we all know, a black hole can swallow any star if they come too close to it, mimicking this incident BH also applies the swallowing process and creates another candidate

Table 1: Few physics based meta-heuristic algorithms

Algorithm name	Acronym	Year	Reference
Simulated annealing	SA	1983	(Kirkpatrick et al., 1983)
Harmony search	HS	2001	(Geem et al., 2001)
Gravitational search algorithm	GSA	2009	(Rashedi et al., 2009)
Black hole optimization	BH	2013	(Hatamlou, 2013)
Multi-verse optimizer	MVO	2015	(Mirjalili et al., 2015)
Charged system search	CSS	2016	(Kaveh, 2016)
Sine cosine algorithm	SCA	2016	(Mirjalili, 2016)

solution randomly to search for an optimal solution in the search space.

MVO is inspired by three ideas of cosmology: black hole, white hole, and wormhole. A black-hole is a region in space-time whose gravity is so strong that it swallows everything, even sun-rays. White holes are hypothetical regions in space-time and singularity that can not be entered from outside, although energy and matter can escape from it. A wormhole is a sort of tunnel that joins distant points in space, even two universes using space-time curvature. Using the concept of inflation rates, adaptive WEP, and adaptive TDR with a black hole, white hole, and wormhole, MVO tries to find the optimal solution for any optimization problem.

CSS is inspired by Coulombian law of electrostatics and Newtonian law of mechanics. This algorithm considers every agent as a charged particle with a pre-planned radius. The measure of charge of the particles is considered based on their standard. As a result of being charged, each particle creates a region of influence *i.e.*, an electric field around them, which exerts an attractive/repulsive force on other electrically charged objects. Therefore, charged particles affect each other based on their fitness values and the distance between their centers. The magnitude of the resultant force is determined by using the electrostatics laws, and the quality of the movement is determined using Newtonian mechanics laws.

SCA tries to find out the optimal solution using a mathematical model based on sine and cosine functions. This algorithm also uses several random and adaptive variables to improve the exploration and exploitation capabilities. This methods reflects better exploration ability when sine and cosine functions return value  $> 1$  and  $< -1$ . It shows better exploitation ability when sine and cosine functions return a value between  $-1$  and  $1$ .

EO has been used to solve several real-life problems. Such as the work mentioned Özkaya et al.

(2020) utilizes EO to solve a structural design optimization problem for a vehicle seat bracket. It is an important industrial design problem as the interest in the design of light and low-cost vehicles are increasing due to the harsh competitive conditions and the transition to new vehicles. EO is used Abdel-Basset et al. (2020) for image segmentation purposes which are considered a crucial step in image analysis and research. To find the optimal threshold value of a grayscale image EO is employed. The work proposed Agwa (2020) uses EO for automatic generation control of interconnected power systems. EO is used to tune the gain of the required proportional-integral-derivative (PID). These works present the promising results produced by EO in distinct fields.

Table 2 summarizes a few merits and demerits Dutta et al. (2020) of some well-known meta-heuristic algorithm.

### 3. Motivation and Contributions

Due to the effectiveness of optimization algorithms in several real-life applications, many such algorithms have been introduced by researchers in recent times. However, one may question that why do we need new optimization algorithms? Can we not solve every optimization problem using the existing ones? Well, according to the *No Free Lunch* theorem (NFL) (Wolpert and Macready, 1997), any two algorithms produce equivalent results when they are evaluated on all possible optimization problems. It has been observed that an algorithm may achieve superior results on some problems, but that does not ensure the same on other problems. In other words, this inference of the NFL theorem is found to be true in many real-life problems which are different from one another, and this very fact keeps the researchers' interest alive. As a result of this researchers are coming

Table 2: Advantages and disadvantages of some state-of-the-art FS methods

Algo-rithm	Advantages	Disadvantages
GA Leardi (1994)	<ul style="list-style-type: none"> <li>• Can reach global optima faster</li> <li>• Random mutation helps to solve a wide range of problems</li> </ul>	<ul style="list-style-type: none"> <li>• Slower convergence rate</li> <li>• Crossover and mutation operators are fixed which reduce flexibility</li> </ul>
PSO Kennedy and Eberhart (1995)	<ul style="list-style-type: none"> <li>• Fast convergence</li> <li>• Few parameters</li> </ul>	<ul style="list-style-type: none"> <li>• Can be difficult to define initial parameter values</li> <li>• Converges prematurely and gets trapped into local minimum in complex problems</li> </ul>
HS Geem et al. (2001)	<ul style="list-style-type: none"> <li>• Easy to implement</li> <li>• Fast convergence</li> </ul>	<ul style="list-style-type: none"> <li>• Parameters have fixed values hence if not tuned properly gets trapped in local optima</li> </ul>
GSA Rashedi et al. (2009)	<ul style="list-style-type: none"> <li>• Good convergence rate</li> <li>• Exploration capability is strong</li> </ul>	<ul style="list-style-type: none"> <li>• Not much attention is given to exploitation</li> </ul>
ABC Karaboga and Basturk (2007)	<ul style="list-style-type: none"> <li>• Few Parameters</li> <li>• Strong exploration capability</li> </ul>	<ul style="list-style-type: none"> <li>• Exploitation is weak</li> </ul>
ACO Dorigo et al. (1996)	<ul style="list-style-type: none"> <li>• Can search population in parallel</li> <li>• Positive feedback mechanism helps in finding optimal solution faster</li> </ul>	<ul style="list-style-type: none"> <li>• Uncertain time of convergence</li> <li>• Less diversity in the population due to pheromone trails feedback</li> </ul>
GWO Mirjalili et al. (2014)	<ul style="list-style-type: none"> <li>• Good convergence</li> <li>• Avoidance of local optima</li> </ul>	<ul style="list-style-type: none"> <li>• May fail to find optimal solution</li> </ul>
WOA Mirjalili and Lewis (2016)	<ul style="list-style-type: none"> <li>• Few parameters to adjust</li> <li>• Easy to implement</li> </ul>	<ul style="list-style-type: none"> <li>• Slow convergence rate</li> <li>• Gets trapped in local optima in complex problems</li> </ul>
CS Yang and Deb (2009)	<ul style="list-style-type: none"> <li>• Few parameters to tune</li> </ul>	<ul style="list-style-type: none"> <li>• Low search efficiency for complex problems within multiple peaks</li> </ul>
SCA Mirjalili (2016)	<ul style="list-style-type: none"> <li>• Efficient, simple and flexible</li> </ul>	<ul style="list-style-type: none"> <li>• Pre-mature convergence</li> </ul>

up with new optimization algorithms. FS is considered an optimization problem (Mafarja and Mirjalili, 2017), so researchers are also striving to put forward new and efficient FS methods using optimization algorithms. This is our key motivation behind proposing a new algorithm by modifying the Equilibrium Optimizer (EO) (Faramarzi et al., 2020). EO, published recently, is inspired from a simple well-mixed dynamic mass balance on a controlled volume (i.e., the mass of the system is constant), which uses a mass balance equation to measure the concentration of a non-reactive component in a controlled volume as a function of disparate source and sink mechanisms. EO is applied on various benchmark functions including unimodal, multimodal, and composition functions. It is applied to several engineering optimization problems like tension spring design, welded beam design, and pressure vessel design. The results obtained from these experiments proved EO's utility as it outperformed a wide range of algorithms.

The present work proposes an improved version of the binary form of EO, known as AIEOU, hybridized with another new meta-heuristic algorithm called Adaptive  $\beta$ -hill climbing (A $\beta$ HCl) (Al-Betar et al., 2019). Moreover, we have also tuned the parameters of EO with the help of Learning Automata. A $\beta$ HCl is previously evaluated on various global optimization functions to prove its effectiveness. It has also been used for FS along with Sailfish optimizer (A $\beta$ BSF) (Ghosh et al., 2020a). Recently, some hybrid FS methods have been proposed (Mafarja et al., 2019; Ghosh et al., 2020a; Chatterjee et al., 2020; Ahmed et al., 2020b), which have proved their effectiveness and superiority over other methods. This has also motivated us to come up with a hybrid version of EO.

For hybridizing meta-heuristic algorithms, generally, two different approaches are followed (Talbi, 2009): low-level and high-level. A low-level approach implants one algorithm in another algorithm. Whereas, in the high-level approach, the algorithms are executed in succession. In this work, we have followed the high-level version to hybridize EO and A $\beta$ HCl, following a pipeline model, where the output of one meta-heuristic algorithm is considered as the input for the other one. To the best of our knowledge, this is the first time EO is hybridized with the A $\beta$ HCl algorithm for solving the FS problem. In a nutshell, the main contributions of this work are as follows:

- A new FS method called AIEOU is introduced using EO and another recently proposed meta-heuristic called A $\beta$ HCl algorithm.
- The parameters of EO are tuned with the help of learning automata.
- The proposed hybrid FS method is assessed on 18 standard UCI datasets (Dua and Graff, 2017) using K-nearest Neighbors (KNN) classifier. datasets Ghosh et al. (2018) are high dimensional datasets that contain a large number of features (mostly irrelevant) and a few samples. AIEOU is applied on these datasets to prove its robustness.
- The proposed FS method is compared with many state-of-the-art meta-heuristic-based FS methods.

## 4. Preliminaries

### 4.1. Equilibrium Optimizer

EO is a physics-based meta-heuristic algorithm that takes inspiration from a well-mixed dynamic mass balance where the mass of the system is constant. A mass balance equation is used to represent the concentration of a non-reactive components in a control volume as a function of various source and sink mechanisms. A dynamic mass balance system is defined as a system where the change in mass w.r.t time is equal to the volume of mass entering the system plus the volume being generated within the system minus the volume that leaves the system. It is represented by Equation 1:

$$V \frac{dM}{dt} = R_v(M_{eq} - M) + R_g \quad (1)$$

where V is the control volume, M is the concentration inside V,  $R_v$  is the volumetric flow rate,  $R_g$  is the mass generation rate inside V,  $M_{eq}$  is the concentration at equilibrium state when no further mass generation is involved and  $\frac{dM}{dt}$  is the rate of change of mass w.r.t time. When the left hand side of Equation 1 becomes 0, that indicates that steady state equilibrium is reached. We, can represent  $\frac{dM}{dt}$  as  $f(\frac{R_v}{V})$ , where f is a function and  $\frac{R_v}{M}$  represents the inverse of residence time represented here as  $\alpha$ . After rearranging Equation (1), we will obtain Equation (2):

$$\frac{dM}{\alpha M_{eq} - \alpha M + \frac{R_g}{V}} = dt \quad (2)$$

Integrating both sides of Equation 2 under the limit  $M_o \leq M \leq M$  and  $t_o \leq t \leq t$  we obtain:

$$\int_{C_o}^C \frac{dM}{\alpha M_{eq} - \alpha M + \frac{R_g}{V}} = \int_{t_o}^t dt \quad (3)$$

After solving Equation 3 we get:

$$M = M_{eq} + (M_o - M_{eq})e^{-\alpha(t-t_o)} + \frac{R_g}{\alpha V}(1 - e^{-\alpha(t-t_o)}) \quad (4)$$

We consider:

$$F = e^{-\alpha(t-t_o)} \quad (5)$$

We initialize the population randomly keeping in mind the dimensions of the search space. The equilibrium state is considered as the optimal state where we have to reach. EO uses the four best solutions so far from the population at every iteration to create another potential solution which is calculated by adding the best four potential solutions. Then with these five potential solutions, an equilibrium pool is created.

$$\vec{M}_{eqpool} = \{\vec{M}_{eq1}, \vec{M}_{eq2}, \vec{M}_{eq3}, \vec{M}_{eq4}, \vec{M}_{avg}\} \quad (6)$$

where,  $\vec{M}_{avg}$  is given by:

$$\vec{M}_{avg} = \frac{\vec{M}_{eq1} + \vec{M}_{eq2} + \vec{M}_{eq3} + \vec{M}_{eq4}}{4} \quad (7)$$

Each particle in each iteration updates its position by selecting one member of the equilibrium pool randomly. Generally,  $\alpha \in [0, 1]$ . The term  $F$  helps EO to find a proper tuning between its exploration and exploitation phases. The term  $t$  in  $F$  is decreased with the number of iterations.

$$t = (1 - \frac{Iter}{MaxIter})^{(a_2 \frac{Iter}{maxIter})} \quad (8)$$

where  $Iter$  represents the iteration number and  $MaxIter$  represents the maximum number of iterations. The term  $a_2$  is constant and controls the exploitation ability. We define  $t_o$  as follows:

$$\vec{t}_o = \frac{1}{\alpha} \ln(-a_1 \text{sign}(\vec{r} - 0.5)[1 - e^{-\vec{\alpha}t}]) + t \quad (9)$$

where  $a_1$  is a constant and controls exploration ability of EO. The term  $\text{sign}(\vec{r} - 0.5)$  effects the direction of exploration and exploitation.  $\vec{r}$  is a random vector  $\in [0, 1]$ . Then updating  $F$ , we get:

$$\vec{F} = a_1 \text{sign}(\vec{r} - 0.5)[e^{-\vec{\alpha}t} - 1] \quad (10)$$

The generation rate  $R_g$  is defined as:

$$\vec{R}_g = \vec{R}_{g_o} e^{(-\vec{\alpha}(t-t_o))} \quad (11)$$

where  $R_{g_o}$  is the initial value of  $R_g$  which is formulated as follows:

$$\vec{R}_{g_o} = G\vec{C}P(\vec{M}_{eq} - \vec{\alpha}\vec{M}) \quad (12)$$

Moreover we define  $G\vec{C}P$  as follows:

$$G\vec{C}P = \begin{cases} 0.5 \cdot r_1 & \text{if } r_2 \geq GP \\ 0 & \text{else} \end{cases} \quad (13)$$

where  $r_1$  and  $r_2$  are random numbers  $\in [0, 1]$  and  $GP = 0.5$ . Hence, finally we obtain the formula for position of a particle.

$$\vec{M} = \vec{M}_{eq} + (\vec{M} - \vec{C}_{eq})\vec{F} + \frac{\vec{G}}{\vec{\alpha}V}(1 - \vec{F}) \quad (14)$$

$a_1$ ,  $\text{sign}(r - 0.5)$ ,  $GP$  and the equilibrium pool are responsible for the exploration ability of EO. On the other hand,  $a_2$ ,  $\text{sign}(r - 0.5)$  and equilibrium pool are responsible for the exploitation phase of EO. These parameters control the tuning between exploration and exploitation abilities of EO. The pseudocode of EO is given in the Algorithm 1.

#### 4.2. Learning Automata

A learning automaton (LA) (Narendra and Thathachar, 1974) is regarded as an epitome decision-making unit situated in a stochastic environment that determines the optimal action through a set of actions and by frequent interactions with the environment.

An automaton contains a finite number of actions, where action is chosen based on a specific probability distribution and applied to the environment. The environment evaluates the impact of the applied action and sends back a reinforcement signal to the automaton. The learning algorithm of the automaton utilizes this response to update the existing action probability distributions. By continuing this process, the automaton increases the chances of better actions, which generate favorable responses from the environment. Figure 3 depicts the basic working principle of a LA.

Learning automata are constituted by the sextuple (Hashemi and Meybodi, 2011)  $(\alpha, \gamma, \Phi, V, G, T)$ .  $\alpha$  is a combination of output actions,  $\gamma$  is a combination of input actions,  $\Phi$  is a combination of internal states,  $V$  is a probability vector regulating the

**Algorithm 1** Pseudo-code of Equilibrium Optimizer**Input:** Problem dependent information**Output:** Best Agent

---

```

1: Initialize population for  $i=1, \dots, \text{number of particles}(n)$ 
2: Assign equilibrium candidates' fitness a large number
3: Assign values to the parameters  $a_1; a_2; GP$ ;
4: while  $Iter < MaxIter$  do
5:
6:   for  $i = 1, \dots, n$  do
7:     Calculate fitness of  $i^{th}$  particle
8:     if  $fit(\vec{M}_i) < fit(\vec{M}_{eq1})$  then
9:       Replace  $\vec{M}_{eq1}$  with  $\vec{M}_i$  and  $fit(\vec{M}_{eq1})$  with  $fit(\vec{M}_i)$ 
10:    else if  $fit(\vec{M}_i) < fit(\vec{M}_{eq2})$  then
11:      Replace  $\vec{M}_{eq2}$  with  $\vec{M}_i$  and  $fit(\vec{M}_{eq2})$  with  $fit(\vec{M}_i)$ 
12:    else if  $fit(\vec{M}_i) < fit(\vec{M}_{eq3})$  then
13:      Replace  $\vec{M}_{eq3}$  with  $\vec{M}_i$  and  $fit(\vec{M}_{eq3})$  with  $fit(\vec{M}_i)$ 
14:    else if  $fit(\vec{M}_i) < fit(\vec{M}_{eq4})$  then
15:      Replace  $\vec{M}_{eq4}$  with  $\vec{M}_i$  and  $fit(\vec{M}_{eq4})$  with  $fit(\vec{M}_i)$ 
16:    end if
17:  end for
18:   $\vec{M}_{avg} = \frac{\vec{M}_{eq1} + \vec{M}_{eq2} + \vec{M}_{eq3} + \vec{M}_{eq4}}{4}$ 
19:  Construct the equilibrium pool  $\vec{M}_{eqpool} = \{\vec{M}_{eq1}, \vec{M}_{eq2}, \vec{M}_{eq3}, \vec{M}_{eq4}, \vec{M}_{avg}\}$ 
20:  Assign  $t = (1 - \frac{Iter}{MaxIter})^{\frac{a_2 \cdot Iter}{MaxIter}}$ 
21:
22:  for  $i = 1, \dots, \text{number of particles}(n)$  do
23:    Randomly choose one candidate from the equilibrium pool
24:    Generate random vectors of  $\vec{\alpha}, \vec{r}$ 
25:    Construct  $\vec{F} = a_1 \times sign(\vec{r} - 0.5) \times [\exp^{-\vec{\alpha} \cdot t} - 1]$ 
26:    Construct  $G\vec{CP} = \begin{cases} 0.5 \cdot r_1 & \text{if } r_2 \geq GP \\ 0 & \text{else} \end{cases}$ 
27:    Construct  $\vec{G}_0 = G\vec{CP} \cdot (\vec{M}_{eq} - \vec{\alpha} \cdot \vec{M})$ 
28:    Construct  $\vec{G} = \vec{G}_0 \cdot \vec{F}$ 
29:    Update concentration  $\vec{M} = \vec{M}_{eq} + (\vec{M} - \vec{M}_{eq}) \cdot \vec{F} + \frac{\vec{R}_g}{\vec{\alpha} \cdot V} \cdot (1 - \vec{F})$ 
30:  end for
31:   $Iter = Iter + 1$ 
32: end while

```

---

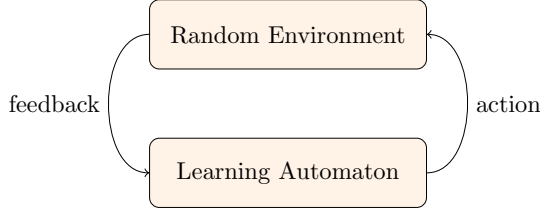


Figure 3: Basic interaction diagram between a learning automaton and its environment

output choices,  $G$  is the output mapping that maps  $\alpha$  to appropriate values in the environment,  $T$  is the learning algorithm used to modify (update)  $V$ .

The learning algorithm  $T$  is a decisive factor affecting the performance of LA. The linear reward-penalty algorithm ( $L_{R-P}$ ) is a unique learning algorithm present in the literature (Thathachar and Sastry, 2002). In a V-model environment, where  $\gamma \in [0, 1]$ , the learning model  $T$  for  $L_{R-P}$  scheme to update the state probability vector  $V$  after receiving the reinforcement signal  $\gamma$  is given by Equation 15 when  $\gamma = 0$  (considered favorable), and Equation 16 when  $\gamma = 1$  (considered unfavorable). Let,  $\alpha_m$  be the action selected at time  $t$  as using the distribution  $V(t)$ .

$$V_n(t+1) = \begin{cases} V_n(t) + x(1 - V_n(t)) & \text{if } n = m \\ V_n(t)(1 - x) & \text{if } n \neq m \end{cases} \quad (15)$$

$$V_n(t+1) = \begin{cases} V_n(t)(1 - y) & \text{if } n = m \\ \frac{y}{r-1} + V_n(t)(1 - y) & \text{if } n \neq m \end{cases} \quad (16)$$

where  $r$  is the number of actions defined in the automaton,  $x$  and  $y$  denote reward and penalty step length, respectively.

#### 4.3. Adaptive $\beta$ -Hill Climbing

Hill climbing algorithm is considered as the simplest form of local search. But due to the fact that it gets stuck at local optima and hence sometimes fails to find the optimal solution,  $\beta$ -hill climbing algorithm (Al-Betar, 2016) ( $\beta$ HHC) was proposed to overcome the shortcomings of hill climbing.  $\beta$ HHC also had some disadvantages of parameter tuning, hence an adaptive version of  $\beta$ HHC is proposed recently, named adaptive  $\beta$ -hill climbing (Al-Betar et al., 2019) ( $A\beta$ HHC).  $\beta$ HHC algorithm uses  $\mathcal{N}$  and  $\beta$  operators for controlling the rate

of exploitation and exploration respectively. Let,  $\mathcal{S} = (s_1, s_2, \dots, s_N)$  be a possible solution, where  $N$  is the dimension of the search space,  $A\beta$ HHC iteratively uses  $\mathcal{N}$  and  $\beta$  operators to generate a better solution  $\mathcal{S}'' = (s_1'', s_2'', \dots, s_N'')$  with the help of a neighbor solution  $\mathcal{S}' = (s_1', s_2', \dots, s_N')$ , which is found out by Equation 17.

$$s'_i = s_i \pm U(0, 1) \times \mathcal{N} \quad \exists i \in [1, N] \quad (17)$$

where  $i$  is a random number  $\in [1, N]$ ,  $N$  is the dimension of the search space,  $\mathcal{N}$  denotes the distance bandwidth between the current solution and neighboring solution and hence helps the algorithm to improve its exploitation capability.  $\beta$  - operator is responsible for exploration. Then the values of new solutions are allocated either from the present solution or randomly from the resembling range with probability value  $\beta \in [0, 1]$ .

$$s''_i = \begin{cases} s_r & \text{if } rand \leq \beta \\ s'_i & \text{if } rand > \beta \end{cases} \quad (18)$$

where  $rand$  and  $s_r$  are random numbers  $\in [0, 1]$  and the range of the dimension of the search space of the current problem respectively.

As stated before,  $\beta$ HHC has disadvantages because it is difficult to find suitable values for its parameters. These parameters affect the capability of the algorithm. So,  $A\beta$ HHC expresses  $\mathcal{N}$  and  $\beta$  as a function of iteration number.  $\mathcal{N}(j)$  is the value of  $\mathcal{N}$  in  $j^{th}$  iteration.  $\mathcal{N}(j)$  is defined as Equation 19 following the work presented in (Mirjalili et al., 2015).

$$\mathcal{N}(j) = 1 - \frac{j^{\frac{1}{K}}}{MaxIter^{\frac{1}{K}}} \quad (19)$$

where  $MaxIter$  represents the maximum number of iterations and  $K$  is a constant.

The value of  $\beta$  in  $j^{th}$  iteration is represented as  $\beta(j)$ . As per (Mahdavi et al., 2007), it is adjusted within a appointed range  $[\beta_{min}, \beta_{max}]$  and defined as Equation 20.

$$\beta(t) = \beta_{min} + (\beta_{max} - \beta_{min}) \times \frac{j}{MaxIter} \quad (20)$$

If the generated neighbor  $\mathcal{S}''$  is an excellent solution compared to the present solution  $\mathcal{S}$ , then it is replaced with  $\mathcal{S}''$  and.

The pseudocode for  $A\beta$ HHC is given in Algorithm 2.

**Algorithm 2** Pseudo-code of adaptive  $\beta$ -hill climbing algorithm

**Input:**  $S = (s_1, s_2, \dots, s_N)$

**Output:**  $S = (s_1, s_2, \dots, s_N)$  (improved)

```

1: Initialize  $\beta_{min}, \beta_{max}, K, t_{max}$ 
2: for  $j = 1, 2, \dots, j_{max}$  do
3:    $S' := S$ 
4:    $N_j = 1 - \left(\frac{j}{j_{max}}\right)^{\frac{1}{K}}$ 
5:    $RandIndex \in (1, N)$ 
6:    $S'_{RandIndex} := S'_{RandIndex} \pm N_j$ 
7:    $S'' := S'$ 
8:    $\beta_j = \beta_{min} + t \times \frac{\beta_{max} - \beta_{min}}{t_{max}}$ 
9:   for  $i = 1, \dots, N$  do
10:    if  $random(0, 1) < \beta_t$  then
11:       $S''[i] := random(LB_i, UB_i)$ 
12:    end if
13:  end for
14:  if  $fitness(S'') \leq fitness(S)$  then
15:     $S := S''$ 
16:  end if
17: end for

```

## 5. Proposed Method

### 5.1. Solution Representation

We consider every particle of EO as a potential solution for our FS approach. The position/concentration of the particles in EO is considered as the vector containing the information about whether a particular feature is selected or not.

The performance of EO is extremely sensitive to the values of its parameters  $a_1$ ,  $a_2$ , and  $GP$ .  $a_1$  controls the exploration ability of EO,  $a_2$  controls the exploitation.  $GP$  controls the probability of participation of the concentration updating by the generation rate. So, tuning these parameters to their proper value is necessary for EO to find the global optima. The values of these parameters are determined through thorough experiments. Though this approach might be able to produce sub-optimal results for a single problem, it would not be considered as a general way to solve optimization problems since different problems may require different parameter settings and this would require exhaustive experiments.

In this work, the parameters of EO have been made adaptive while keeping the model similar to the basic EO. Here, each agent is given the capability to adjust the values of its parameters individu-

ally, which means that each agent can have different values for  $GP$ ,  $a_1$ , and  $a_2$ . So, Equation 8 becomes Equation formula1 for the proposed method, Equation 10 becomes Equation 22 and Equation 13 becomes Equation 23.

$$t_i = \left(1 - \frac{Iter}{MaxIter}\right)^{\left(a_{2i} \cdot \frac{Iter}{MaxIter}\right)} \quad (21)$$

$$\vec{F}_i = a_{1i} \text{sign}(\vec{r} - 0.5)[e^{-\vec{\alpha}t} - 1] \quad (22)$$

$$\overrightarrow{GCP}_i = \begin{cases} 0.5 \cdot r_1 & \text{if } r_2 \geq GP_i \\ 0 & \text{else} \end{cases} \quad (23)$$

where  $GP_i$ ,  $a_{1i}$ , and  $a_{2i}$  are the parameters of  $i^{th}$  agent (solution). These parameters are updated after each iteration using LA. At each iteration  $Iter$ ,  $i^{th}$  agent selects the values of  $GP_i$ ,  $a_{1i}$ , and  $a_{2i}$ . Then the goodness of the parameter selection is evaluated by comparing the fitness of the previous and updated positions in order to update the LA assigned for the agent. A parameter selection process is considered as ‘successful’ when the fitness of the updated agent (solution) concentration is lower than the fitness of the previous agent (solution) concentration. If the parameter selection is successful the assigned LA to the  $i^{th}$  agent will be sent a favorable response and the corresponding decision will be rewarded otherwise will be penalized.

The new value of the parameter  $\epsilon$ ,  $\epsilon \in \{GP_i, a_{1i}, a_{2i}\}$  either will be the same as the current value or will be greater or smaller than the current value by a pre-defined fixed amount  $\delta_\epsilon$ .

$$\epsilon(t+1) = \begin{cases} \epsilon(t) + \delta_\epsilon & \text{increase} \\ \epsilon(t) & \text{standby} \\ \epsilon(t) - \delta_\epsilon & \text{decrease} \end{cases} \quad (24)$$

This approach is modeled by a 3-action learning automata, with each action corresponding to increase, decrease, or no-change. In order to update a parameter, responsible LA selects one of its actions, e.g., {increase, decrease, standby}. Then depending on the selected action, the parameter will be updated as per Equation 24. To sum up, 3 LAs,  $LAGP$ ,  $LA_{a_1}$ ,  $LA_{a_2}$ , are assigned for the three parameters and each LA has 3-decisions, {increase, decrease, standby}. Now, for the three parameters  $GP$ ,  $a_1$ , and  $a_2$ , three values  $\delta_{GP}$ ,  $\delta_{a_1}$ , and  $\delta_{a_2}$  are required to update the corresponding LA.

Figure 4 shows the effect of the threshold  $\delta_{GP}$  on the performance of EO. The fitness value of the best agent obtained for different values of  $\delta_{GP}$  is plotted for all the datasets. Similarly, Figure 5 and Figure 6 shows the effect of  $\delta_{a_1}$  and  $\delta_{a_2}$  on the proposed method respectively. Observing Figures 4 - 6,  $\delta_{GP} = 0.05$ ,  $\delta_{a_1} = 0.5$ ,  $\delta_{a_2} = 0.5$  have been set for further experiments.

### 5.2. Fitness Function

The main target of any FS method is to minimize the number of features and maximize the classification accuracy of a classification problem when this feature subset is used (Pudil et al., 1994). Here we apply AIEOU to find the best feature subset and calculate the accuracy of this subset with KNN classifier. Let  $\mathcal{A}_c$  be the accuracy of the model calculated using a classifier,  $d_s$  be the number of selected features, and  $\mathcal{D}_t$  be the total number of features. Hence,  $(1 - \mathcal{A}_c)$  represents the classification error and  $\frac{d_s}{\mathcal{D}_t}$  represents the fraction of features selected from the original feature set. We define the fitness function as:

$$\downarrow \text{Fitness} = \omega * (1 - \mathcal{A}_c) + (1 - \omega) * \frac{d_s}{\mathcal{D}_t} \quad (25)$$

where  $\omega \in [0, 1]$  denotes weightage given to the classification error.

As our main goal is to minimize the fitness function, we try to reduce the classification error as well as the number of selected features. For each feature subset, we calculate the fitness value and decide which is better depending on whose fitness value is lower. Here, we have used  $\omega = 0.99$ .

### 5.3. Improved Equilibrium Pool

After creating the equilibrium pool with the four best agents in the present population and the one created by taking the average of these four agents, we are applying A $\beta$ H $\alpha$ C on the pool to find if a better solution is available. If we find a better solution, then we replace the previous solution with the better one. After obtaining a better equilibrium pool in terms that the agents have been updated, rather than taking one of the members of the pool randomly we are choosing the best agent in the equilibrium pool. This measure has been taken to avoid the fact that when chosen randomly, we are ignoring the best agent with a probability of 0.8 and basically harming exploration. With these changes, we propose the AIEOU.

### 5.4. Transfer Function

As FS is a binary optimization problem (Ghosh et al., 2020b), its output  $\in \{0, 1\}$ . Where 0 represents that the feature is not chosen as it is redundant whereas 1 represents that the feature is useful and hence chosen. But the possibility of the obtained result going out of the output range is immense. To ensure that the output always stays within the expected range we have to perform binarization on each agent. This task is performed by the U-shaped transfer function (Mirjalili et al., 2020). The U-shaped transfer function, depicted in Figure 7, is given by -

$$U(x) = \delta \cdot |x|^\epsilon \quad (26)$$

$$\gamma = \begin{cases} 1, & \text{if } U(x) \geq \text{rnd} \\ 0, & \text{if } U(x) \leq \text{rnd} \end{cases} \quad (27)$$

Where  $\delta$  and  $\epsilon$  are the two controlling parameters.  $\delta$  defines the slope and  $\epsilon$  defines the width of the basin of the transfer function. The range of this function  $\in [0, 1]$ . If the transfer function produces output  $> \text{rnd}$ , where rnd is a random number with uniform distribution in the range (0, 1), we set the value to be 1 i.e., we consider that attribute is useful and if it is  $\leq \text{rnd}$ , we set the value to be 0 i.e., the attribute is redundant, hence it will not be considered (Mirjalili and Lewis (2013), Mafarja et al. (2019)).

### 5.5. Computational Complexity

The computational complexity depends on the input size of the problem, the maximum number of iterations, and some other operations that we perform. It is usually given by  $O$  notation. The computational complexity of AIEOU is  $O(M_{iter} \times P_{size} \times D_s \times t_{fitness})$ , where  $M_{iter}$  represents the maximum number of iterations,  $P_{size}$  represents the plurality of potential solutions,  $D_s$  represents the dimension of the search space, and  $t_{fitness}$  indicates the required time for calculating the fitness of a particular solution using a classifier. The usage of A $\beta$ H $\alpha$ C is to find a better solution if available in the vicinity of the current solution and LA to tune the parameters. They do not affect the computational cost in terms of  $O$ -notation.

Figure 8 depicts the workflow of the proposed method.

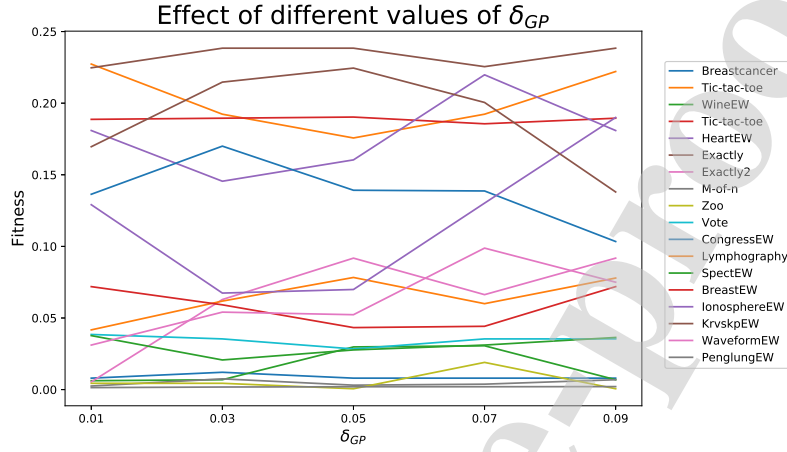


Figure 4: Effect of  $\delta_{GP}$  on the fitness value of the best agent (attained optima) by the proposed method for all the 18 UCI datasets in consideration

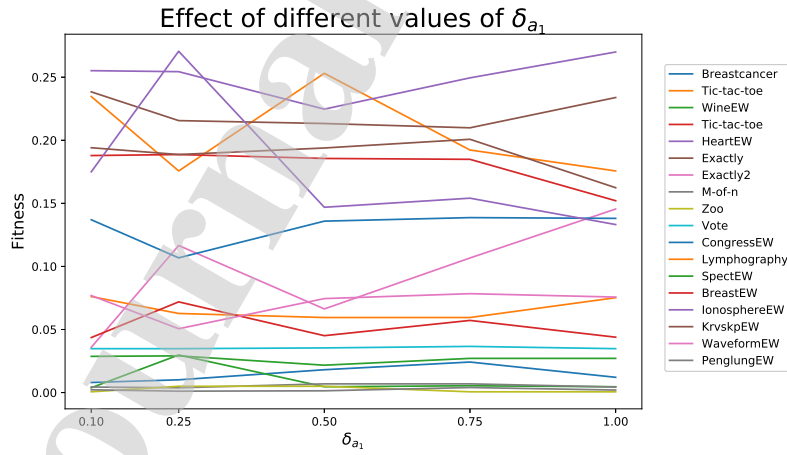


Figure 5: Effect of  $\delta_{a_1}$  on the fitness value of the best agent (attained optima) by the proposed method for all the 18 UCI datasets in consideration

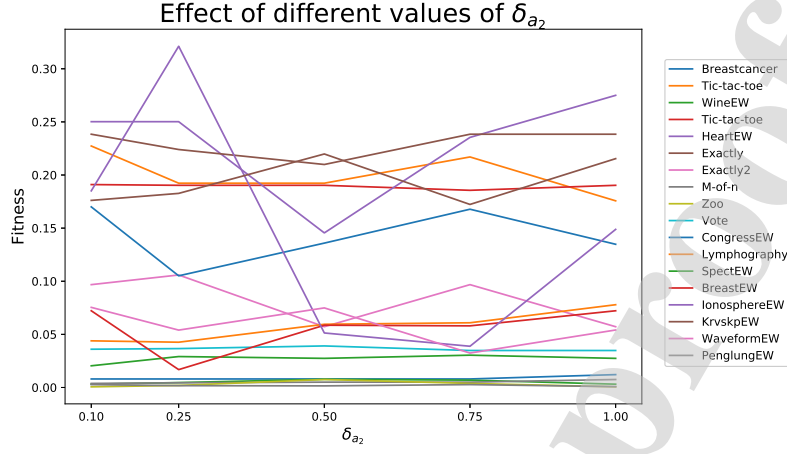


Figure 6: Effect of  $\delta_{a2}$  on the fitness value of the best agent (attained optima) by the proposed method for all the 18 UCI datasets in consideration

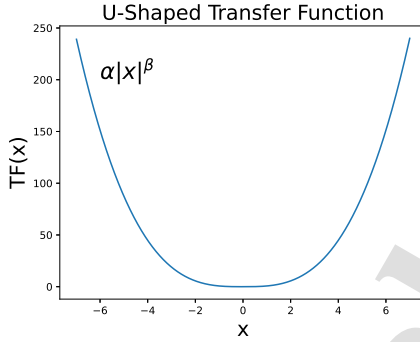


Figure 7: U shaped transfer function

## 6. Experiments

### 6.1. Dataset Details

In order to investigate the performances of EO and AIEOU, 18 standard UCI datasets (Dua and Graff, 2017) are used. The datasets are hand-picked from diverse domains. A primary idea of these datasets is provided in Table 3. As the datasets used here are assorted in terms of the number of features and instances, so it helps us to understand the robustness of the proposed FS method.

### 6.2. Software

The conducted experiments are executed on a system with 7th Gen, i3 processor having 8 GB RAM.

### 6.3. Parameter Settings

For any meta-heuristic algorithm, its parameters play a very important role in finding the optimal solutions. So we perform experiments by changing one parameter w.r.t other. Here, we have shown the effect of different population sizes on finding the optimal solution. Our experiments show the classification accuracy obtained by varying the population size as 5,10,20,30,40 and 50. Figure 9 shows the findings of this experiment. These graphs also show us a comprehensive comparison between EO and AIEOU based on different population sizes. Figure 10 shows the convergence of the best solution after every iteration.

### 6.4. Result and Discussion

To check the effectiveness and superiority of AIEOU over EO, we have applied it on 18 standard UCI datasets, whose descriptions are provided in Table 3. These datasets are widely used for the evaluation of FS methods.

It is pretty clear from Table 4, that AIEOU has better tuning between its exploration and exploitation phase. Out of the 18 datasets, EO could not beat AIEOU in any of them. Only in the case of PenglungEW both produce the same accuracy (100%). AIEOU produces 100% accuracy in 10 datasets (55.55%): Breastcancer, WineEW, Exactly, M-of-n, Zoo, Vote, CongressEW, IonosphereEW, SonarEW, and PenglungEW. In the case of Tic-tac-toe, AIEOU beats EO with a margin of 15%. EO is lagging behind AIEOU by 13%

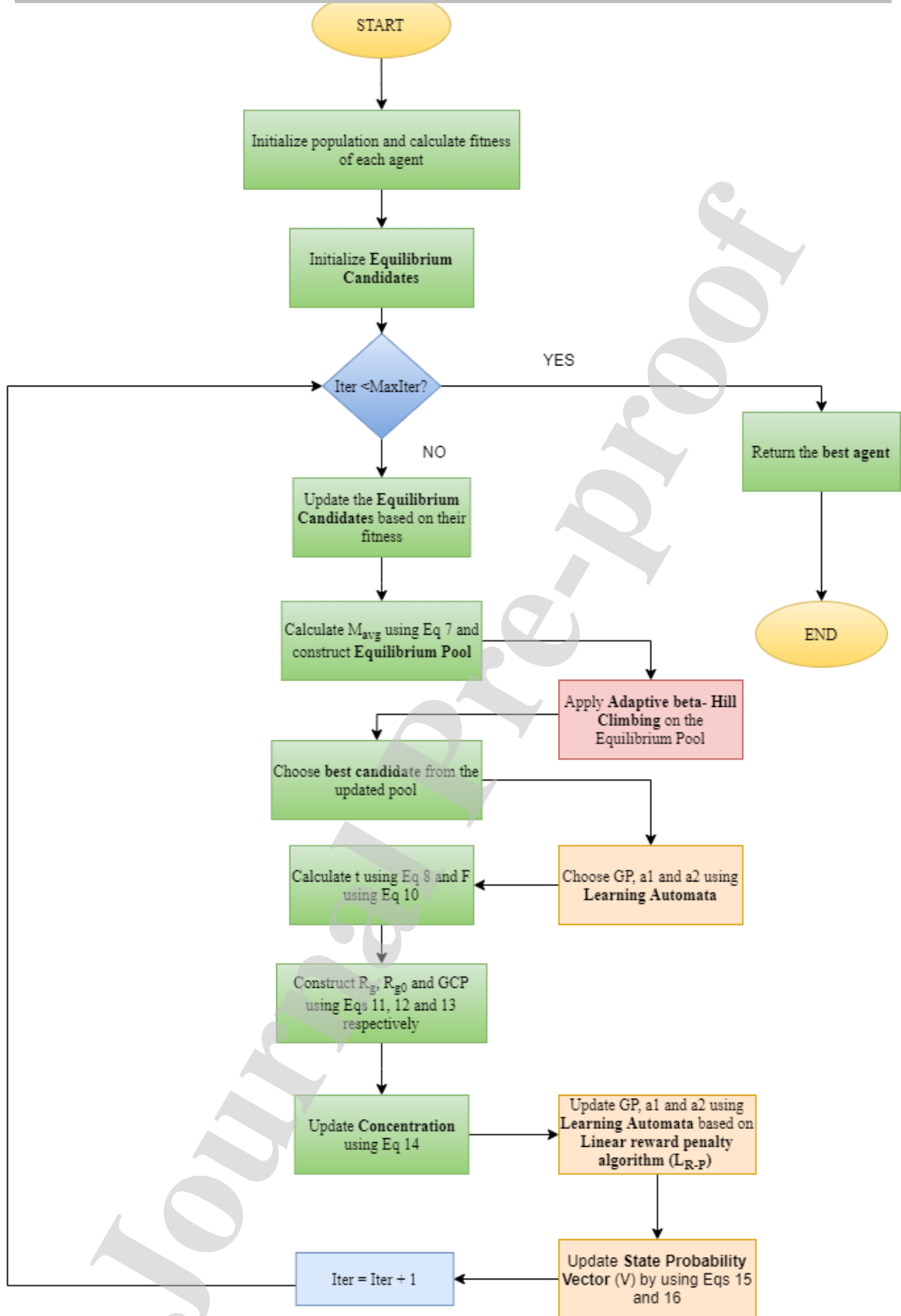


Figure 8: Flowchart of the proposed AIEOU

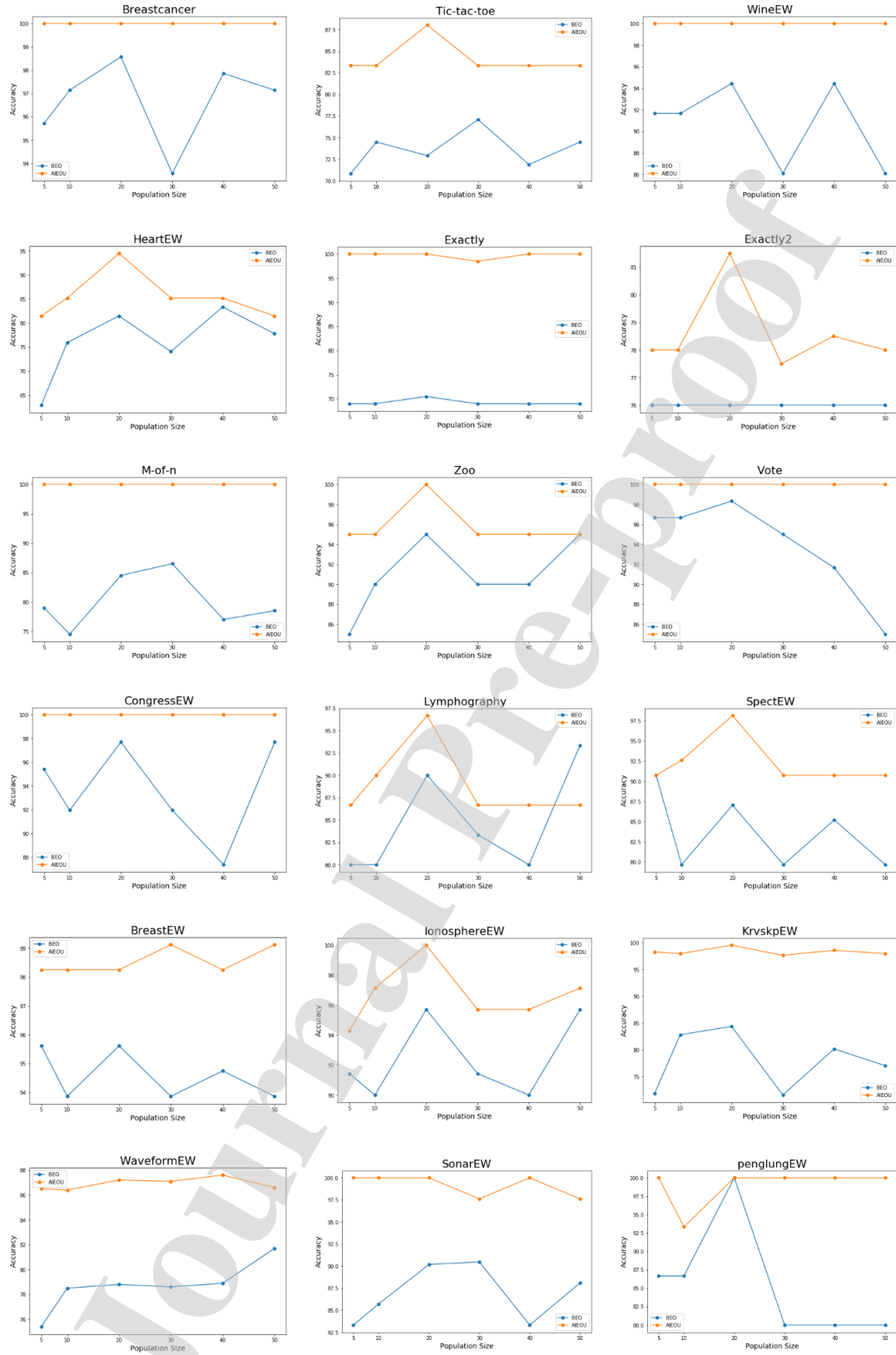


Figure 9: Graphs for achieved classification accuracy using different population size for 18 UCI datasets using EO and AIEOU

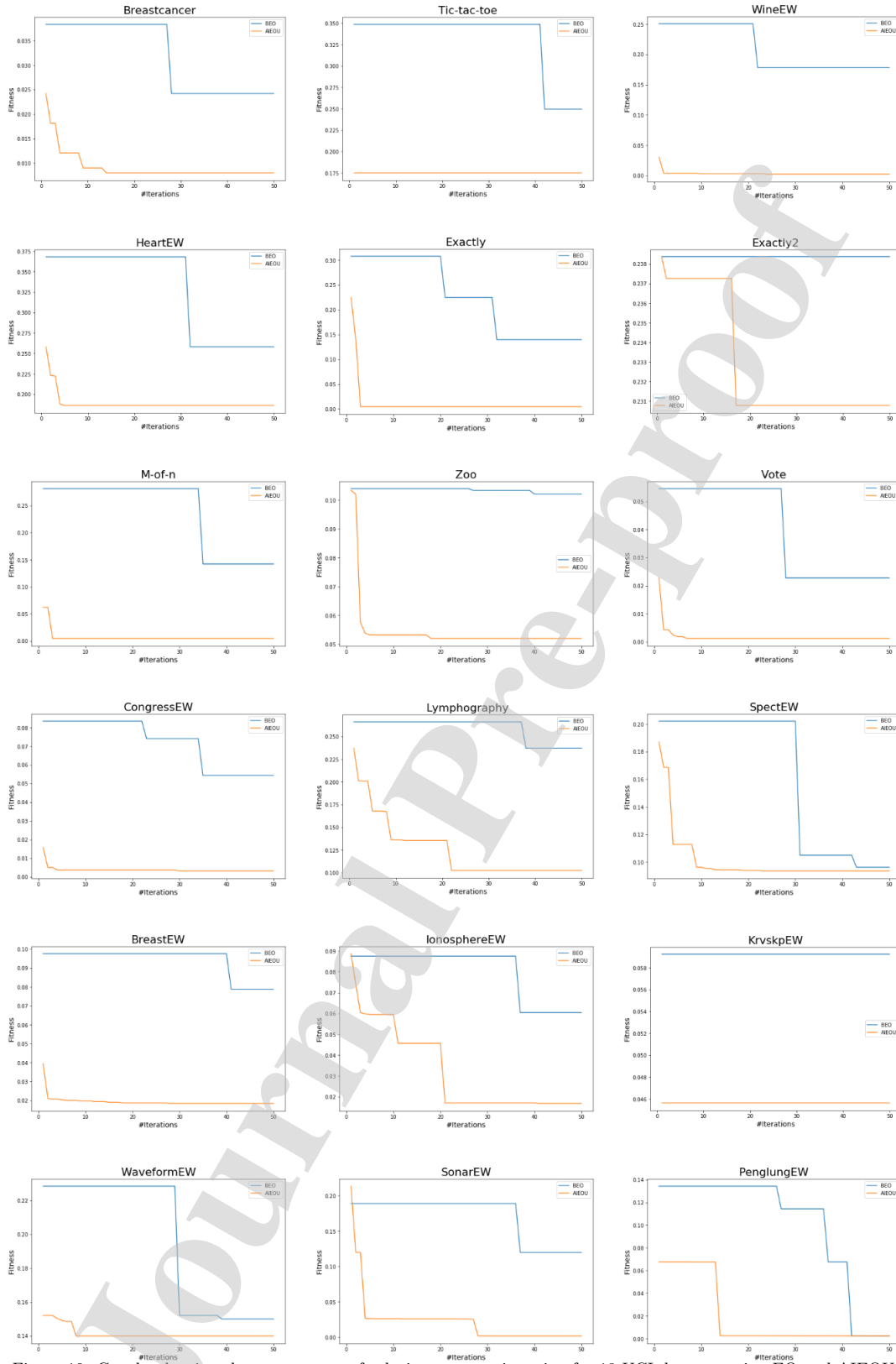


Figure 10: Graphs showing the convergence of solution at every iteration for 18 UCI datasets using EO and AIEOU

Table 3: Brief description of the datasets utilized to assess the proposed FS method

Sl. No.	Dataset	#Attributes	#Samples	#Classes	Dataset Domain
1	Breastcancer	9	699	2	Biology
2	Tic-tac-toe	9	958	2	Game
3	WineEW	13	178	3	Chemistry
4	HeartEW	13	270	2	Biology
5	Exactly	13	1000	2	Biology
6	Exactly2	13	1000	2	Biology
7	M-of-n	13	1000	2	Biology
8	Zoo	16	101	6	Artificial
9	Vote	16	300	2	Politics
10	CongressEW	16	435	2	Politics
11	Lymphography	18	148	2	Biology
12	SpectEW	22	267	2	Biology
13	BreastEW	30	569	2	Biology
14	IonosphereEW	34	351	2	Electromagnetic
15	KrvskpEW	36	3196	2	Game
16	WaveformEW	40	5000	3	Physics
17	SonarEW	60	208	2	Biology
18	PenglungEW	325	73	2	Biology

in the case of HeartEW. Considering Exactly, the difference is 30%. In the case of M-of-n, the margin is 15%. For SpectEW, it is 10%. AIEOU produces much better result by achieving 99.53% accuracy. In the rest datasets, AIEOU beats EO with slight margin. Considering the number of selected features, EO has competed well against AIEOU.

As Table 4 indicates, in case of Breastcancer, Tic-tac-toe, Exactly2, CongressEW, Lymphography and WaveformEW, EO beats AIEOU. They both selects the same number of features in Exactly and M-of-n. In the rest 10 datasets, AIEOU has the upper hand. The difference between EO and AIEOU is quite evident in the case of PenglungEW as the number of features selected ratio is 160:29. From all these observations, we can say for sure that AIEOU has a fine-tuning between exploration and exploitation, which is helping it find better solution than EO. Hence, the proposed method is capable of producing better results.

## 7. Comparison

This section presents the comparison of AIEOU with other 8 state-of-the-art methods to establish its supremacy over them. These methods comprise of most popular meta-heuristics algorithms namely, GA and PSO. It is also compared with adaptive switching grey-whale optimizer (ASGW), hybrid serial grey-whale optimizer (HSGW), and random switching grey-whale optimizer (RSGW), which are

the results of hybridization of GWO and WOA (Mafarja et al., 2019). Then AIEOU is compared with three recently proposed meta-heuristic methods name: Late Acceptance Hill Climbing Based Social Ski Driver Algorithm for Feature Selection (SSDs+LAHC) (Chatterjee et al., 2020), ring theory based harmony search (RTHS) (Ahmed et al., 2020b) and A $\beta$ BSF. A brief description is given in Table 5.

Table 6 shows the comparison table of AIEOU and the 8 state-of-the-art methods based on achieved classification accuracy. It is pretty clear that AIEOU stands at the top over the other methods. AIEOU produces 100% accuracy in 10 datasets (55.55%), namely: Breastcancer, WineEW, Exactly, M-of-n, Zoo, Vote, CongressEW, IonosphereEW, SonarEW and PenglungEW. It grabs the first position in 15 datasets (83.33%).

AIEOU beats SSDs+LAHC in 9 datasets (50%). In the rest 9 datasets it produces equivalent result. RTHS also could not beat AIEOU in any case. AIEOU outperforms RTHS in 5 datasets (27.77%). However, it achieves same classification accuracy in the rest 13 datasets. AIEOU produces equivalent results in 9 datasets with A $\beta$ BSF. It beats A $\beta$ BSF in 6 datasets (33.33%). In the rest 3 datasets A $\beta$ BSF has the upper hand. HSGW achieves same classification accuracy as AIEOU achieves in 5 datasets, produces less classification in 13 datasets (72.22%). RSGW produces same classification accuracy as AIEOU produces in 4 datasets.

Table 4: Performance of EO and AIEOU in terms of achieved classification accuracy and number of selected features using KNN-classifier (highest classification accuracies and lowest no. of selected features are in bold)

Sl. No.	Dataset	Original		EO		AIEOU	
		#Accuracy	#Features	#Accuracy	#Features	#Accuracy	#Features
1	Breastcancer	96	9	98.57	<b>4</b>	<b>100</b>	8
2	Tic-tac-toe	81.1	9	72.92	<b>3</b>	<b>88.02</b>	9
3	WineEW	66.67	13	94.44	4	<b>100</b>	<b>3</b>
4	HeartEW	68.15	13	81.48	<b>1</b>	<b>94.44</b>	3
5	Exatly	72.3	13	70.5	<b>6</b>	<b>100</b>	<b>6</b>
6	Exactly2	73.3	13	76	<b>1</b>	<b>81.5</b>	8
7	M-of-n	87.4	13	84.5	<b>6</b>	<b>100</b>	<b>6</b>
8	Zoo	87	16	95	5	<b>100</b>	4
9	Vote	92.33	16	98.33	2	<b>100</b>	<b>1</b>
10	CongressEW	92.18	16	97.70	<b>1</b>	<b>100</b>	6
11	Lymphography	81.33	18	90	<b>4</b>	<b>96.67</b>	6
12	SpectEW	82.22	22	87.03	9	<b>98.15</b>	<b>6</b>
13	BreastEW	92.63	30	95.61	7	<b>98.25</b>	<b>3</b>
14	IonosphereEW	83.43	34	95.71	6	<b>100</b>	4
15	KrvskpEW	96.1	36	84.35	17	<b>99.53</b>	<b>11</b>
16	WaveformEW	81.44	40	78.8	<b>18</b>	<b>87.2</b>	19
17	SonarEW	80.95	60	90.48	15	<b>100</b>	<b>14</b>
18	PenglungEW	81.33	325	<b>100</b>	160	<b>100</b>	<b>29</b>

Table 5: The state-of-the-art methods used to validate the proposed method

Sl. No.	Method	Acronym	Year	Reference
1	Social ski-driver algorithm with late acceptance hill climbing	SSDs+LAHC	2020	(Chatterjee et al., 2020)
2	Ring theory based harmony search	RTHS	2020	(Ahmed et al., 2020b)
3	Adaptive $\beta$ binary sailfish optimizer	A $\beta$ BSF	2020	(Ghosh et al., 2020a)
4	Hybrid serial grey-whale optimizer	HSGW	2019	(Mafarja et al., 2019)
5	Random switching serial grey-whale optimizer	RSGW	2019	(Mafarja et al., 2019)
6	Adaptive switching grey-whale optimizer	ASGW	2019	(Mafarja et al., 2019)
7	Binary Genetic algorithm	BGA	1994	(Leardi, 1994)
8	Particle swarm optimization	PSO	1995	(Kennedy and Eberhart, 1995)

In the rest of the 14 datasets (77.77%), AIEOU wins by significant margin. However, ASGW beats AIEOU in only one dataset, BreastEW, achieves equivalent results in 5 datasets. In the rest 12 datasets (66.66%), AIEOU outperforms RSGW. BGA and AIEOU produces equivalent results in only 2 datasets. AIEOU produces better classification accuracy in the rest 16 datasets (88.89%). Similar thing can be said for BPSO. It produces equivalent results in 2 datasets but loses in the rest 16 datasets (88.89%) when compared with AIEOU on the basis of achieved classification accuracy.

Table 7 depicts the comparison of AIEOU with the aforementioned 8 state-of-the-art methods based on number of selected features. In 6 datasets (33.33%), namely: HeartEW, Exactly, M-of-n, Zoo, Vote, and IonosphereEW, AIEOU selects the least number of features.

In Breastcancer, it chooses the highest number of features. In Tic-tac-toe also, it chooses the highest number of features along with SSDs+LAHC and RTHS. It holds *second* position along with SSDs+LAHC and A $\beta$ BSF in WineEW. In case of Exactly2, Lymphography and SpectEW, it holds the *third* position. In CongressEW, it holds *fifth* position. AIEOU stands in the *second* position in the case of BreastEW, WaveformEW, SonarEW, and PenglungEW. In KrvskpEW, AIEOU holds *second* position along with BGA.

Each method is run independently 20 times, for each dataset. The Friedman test is a non-parametric statistical test that Friedman (1937). It is used to detect differences in treatments across multiple test attempts. It considers a column as a group and a row as a block. Here, the null hypothesis is that the two sets of results produced by two algorithms are drawn from the same distribution. So if any difference is encountered in the produced results, the only reason is the sampling error. Initially friedman test is performed to find out whether there is a statistical significance between the methods based on p-values. It is important to note that the Friedman test is an omnibus test, like its parametric alternative; that is, it tells us whether there are overall differences, but does not pinpoint which groups in particular differ from each other. To do this we need to run post-hoc tests. Here, we have used the Nemenyi test Pohlert (2014). In statistics, the Nemenyi test is a post-hoc test intended to find the groups of data that differ after a statistical test of multiple comparisons has rejected the null hypothesis that the performance of the comparisons

on the groups of data is similar. It does pair-wise tests of performance. Table 8 provides the pairwise p-values. When multiple hypotheses are tested, the chance of observing a rare event increases, and the likelihood of incorrectly rejecting a null hypothesis as well (i.e., making a Type I error) increases. Here, need to use a Bonferroni adjustment Westfall (1997) on the results we get from the post-hoc tests because we are making multiple comparisons, which makes it more likely that we will declare a result significant when we should not (a Type I error). The Bonferroni correction compensates for that increase by testing each individual hypothesis at a significance level of  $\alpha/m$ , where  $\alpha$  is the significance level and 'm' is the number of hypotheses. For example, if a trial is testing  $m=10$  hypotheses with a desired  $\alpha = 0.05$ , then the Bonferroni correction would test each individual hypothesis at  $\alpha = 0.05/10 = 0.005$ . Here, we have performed the test at 0.05% significance level. So, if the generated p-values are less than or equal to  $0.05 / 8 = 0.00625$ , (after the Bonferroni adjustment, as there are 8 comparisons) then we say that the two set of results belong to different (statistically) distributions, which in turn rejects of the null hypothesis.

Details of the computational time taken by AIEOU and the state-of-the-art methods considered here for comparison are given in Table 9. The time is given in 'seconds' for a single run of every FS method for each of the 18 standard UCI datasets. From this table we can see that AIEOU finds the optimal solution within acceptable amount of time. Those methods which are taking less time than AIEOU, could not find better solution. As the main goal of any FS method is to find the most optimal solution, we can say that AIEOU performs better than other methods considered.

From the above discussions, we can say that AIEOU achieves the best classification accuracy in almost every dataset. It outperforms other 8 methods. Although in case of selecting least features, it stands at *third* position. However, considering achieved classification accuracy and the number of selected features, we can state that AIEOU is a better classifier than the 8 state-of-the-art methods and it will be a good choice for feature selection in other spheres of research and application.

## 8. Additional Test on Microarray Data

The previous section proves the effectiveness of AIEOU for FS on standard UCI datasets. In this

Table 6: Comparison of AIEOU with other 8 state-of-the-art methods based on achieved accuracy (highest classification accuracy is in bold)

Sl. No.	Dataset	AIEOU	SSDs+LAHC	RTHS	A $\beta$ BSF	HSGW	RSGW	ASGW	BGA	BPSO
1	Breastcancer	<b>100</b>	98.93	99.28	<b>100</b>	98.6	97.1	98.5	97.43	96.29
2	Tic-tac-toe	88.02	87.24	83.8	<b>88.54</b>	82.8	85.9	86.5	79.96	79.96
3	WineEW	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	98.88	97.75
4	HeartEW	<b>94.44</b>	91.67	<b>94.44</b>	<b>94.44</b>	92.3	84.8	83.1	87.41	83.7
5	Exactly	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	99.7	99.9	<b>100</b>	<b>100</b>
6	Exactly2	<b>81.5</b>	79	76	81	<b>81.5</b>	77.9	77.7	77	76.8
7	M-of-n	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>
8	Zoo	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	90.2	96.08
9	Vote	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	98.3	99.6	98.4	97.33	96
10	CongressEW	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	97.5	96.1	99.4	96.79	96.33
11	Lymphography	96.67	96.67	96.67	<b>100</b>	93.4	89.3	88.4	83.78	89.19
12	SpectEW	<b>98.15</b>	95.15	<b>98.15</b>	94.44	86.2	81.5	87	89.55	88.81
13	BreastEW	98.25	98.25	98.25	<b>100</b>	98.1	<b>98.2</b>	<b>100</b>	97.54	97.19
14	IonosphereEW	<b>100</b>	96.43	<b>100</b>	98.57	94.4	<b>97.8</b>	97.2	94.89	94.89
15	KrvskpEW	<b>99.53</b>	97.81	98.75	99.06	97.3	97.2	97.1	98.5	97.31
16	WaveformEW	<b>87.2</b>	84.4	84.1	86	74.8	75.7	74.6	78.36	75.6
17	SonarEW	<b>100</b>	97.62	<b>100</b>	97.62	96.4	97.9	94.8	99.04	94.23
18	PenglungEW	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	94.2	<b>100</b>	<b>100</b>	91.89	91.89
Average rank		<b>1.17</b>	2.39	2.17	1.5	3.83	4	4.11	4.28	5.17
Assigned rank		<b>1</b>	4	3	2	5	6	7	8	9

Table 7: Comparison of AIEOU with other 8 state-of-the-art methods based on number of selected features (least number of features are in bold)

Sl. No.	Dataset	AIEOU	SSDs+LAHC	RTHS	A $\beta$ BSF	HSGW	RSGW	ASGW	BGA	BPSO
1	Breastcancer	8	<b>2.5</b>	3	4	5	5.933	4.867	4	4
2	Tic-tac-toe	9	9	9	7	7	7	7	<b>5</b>	6
3	WineEW	3	3	<b>2</b>	3	4.533	5.867	6	4	5
4	HeartEW	<b>3</b>	5	5	5	8.767	6.133	6.367	5	<b>3</b>
5	Exactly	<b>6</b>	<b>6</b>	<b>6</b>	8	6.7	7.1	6.867	<b>6</b>	<b>6</b>
6	Exactly2	8	8	<b>1</b>	11	9	9.2	8	<b>1</b>	<b>1</b>
7	M-of-n	<b>6</b>	<b>6</b>	<b>6</b>	7	6.8	7.1	6.867	<b>6</b>	<b>6</b>
8	Zoo	<b>4</b>	4.5	<b>4</b>	6	5.533	5.3	7.6	<b>4</b>	5
9	Vote	<b>1</b>	4.5	2	3	7.567	8.8	9	5	3
10	CongressEW	6	5.5	4	4	8.867	9.7	8.833	<b>2</b>	3
11	Lymphography	6	6.5	4	15	10.567	10.567	11.2	5	5
12	SpectEW	6	9	4	8	10.233	13.3	10.167	5	6
13	BreastEW	3	9	<b>2</b>	11	16.667	17.5	15.833	8	9
14	IonosphereEW	<b>4</b>	12	8	7	18	20.5	17.3	7	7
15	KrvskpEW	11	20	<b>9</b>	32	24.8	24.8	24.5	11	12
16	WaveformEW	19	22.5	<b>15</b>	30	27	27.333	25.833	<b>15</b>	<b>15</b>
17	SonarEW	14	23.5	<b>8</b>	17	34.3	36.433	35.5	19	22
18	PenglungEW	29	140	<b>13</b>	39	135.33	181.2	170.3	84	130
Average rank		2.39	3.39	<b>1.55</b>	4.17	5.11	6.06	5.28	2.17	2.61
Assigned rank		3	5	<b>1</b>	6	7	9	8	2	4

Table 8:  $p$ -values generated *via* pair-wise Friedman test with the Nemenyi post-hoc test using the results obtained from 20 independent runs of the proposed AIEOU method and all the state-of-the-art methods used for comparison

Dataset	SSDs+LAHC	RTHS	A $\beta$ BSF	HSGW	RSGW	ASGW	BGA	BPSO
Breastcancer	0.001	0.164	0.488	0.001	0.001	0.001	0.001	0.001
Tic-tac-toe	0.9	0.001	0.9	0.001	0.001	0.629	0.009	0.001
WineEW	0.9	0.9	0.392	0.001	0.001	0.003	0.072	0.001
HeartEW	0.787	0.001	0.9	0.001	0.001	0.839	0.107	0.015
Exactly	0.9	0.9	0.254	0.001	0.001	0.019	0.001	0.001
Exactly2	0.004	0.9	0.9	0.001	0.001	0.009	0.011	0.011
M-of-n	0.9	0.9	0.612	0.001	0.001	0.007	0.001	0.001
Zoo	0.392	0.072	0.488	0.115	0.017	0.9	0.9	0.302
Vote	0.115	0.734	0.009	0.001	0.001	0.004	0.001	0.001
CongressEW	0.014	0.001	0.067	0.001	0.001	0.001	0.001	0.001
Lymphography	0.175	0.091	0.822	0.003	0.002	0.9	0.612	0.212
SpectEW	0.857	0.9	0.664	0.001	0.001	0.164	0.001	0.001
BreastEW	0.001	0.124	0.014	0.001	0.001	0.008	0.001	0.001
IonosphereEW	0.085	0.9	0.354	0.001	0.001	0.001	0.001	0.001
KrvskpEW	0.056	0.9	0.9	0.001	0.001	0.007	0.001	0.001
WaveformEW	0.001	0.9	0.142	0.001	0.001	0.001	0.001	0.001
SonarEW	0.212	0.9	0.001	0.001	0.001	0.001	0.099	0.001
PenglungEW	0.107	0.699	0.001	0.001	0.001	0.001	0.030	0.001

Table 9: Computational time comparison of AIEOU with few state-of-the-art methods

Dataset	AIEOU	SSDs+LAHC	RTHS	A $\beta$ BSF	HSGW	RSGW	ASGW	BGA	BPSO
BreastCancer	41.6038	284.4343	79.39269	501.5298	5.645988	6.167367	6.043857	8.558856	9.484692
Tic-tac-toe	51.60454	568.5071	126.7236	897.4343	9.112209	9.299647	9.547518	11.00313	15.04905
Wine	13.27103	148.9643	36.72728	146.2968	2.199575	2.199005	2.266556	5.802927	4.331722
HeartEW	16.40863	169.4982	46.25367	152.3111	2.781135	3.237339	2.92879	7.136589	5.855433
Exactly	60.45272	461.7837	130.8398	644.148	9.705345	12.67615	10.50372	12.72327	13.41135
Exactly2	48.31129	871.7877	113.1011	828.3796	9.616275	9.747207	10.26053	12.68336	15.62203
M-of-n	52.86973	343.448	131.1547	822.6228	9.771276	12.06348	10.3175	12.64771	14.52421
Zoo	10.12563	115.7439	28.89036	124.7213	1.618331	2.868032	1.661179	5.221694	3.393704
Vote	20.0757	167.4907	45.85368	251.1039	3.082977	4.907526	3.218837	6.975272	6.082073
CongressEW	27.29547	199.6871	57.38437	262.4423	4.167467	8.33757	4.403181	8.024364	7.503862
Lymphography	11.84859	131.7186	33.44731	140.0983	2.170386	3.795401	2.052081	5.298428	4.452927
SpectEW	15.44286	236.2166	44.269	198.9711	3.038768	4.312171	3.166366	7.123889	4.902756
BreastEW	26.07796	357.3333	72.12786	337.3407	5.412808	6.427983	5.783034	8.219117	9.956857
Ionosphere	23.08059	284.2316	57.94887	271.5085	4.491678	5.669727	4.695843	7.979099	8.389444
KrVsKpEW	553.4841	2291.531	1272.034	4807.749	86.79092	95.94066	92.80219	75.94349	126.0515
WaveformEW	1519.973	15133.25	2908.037	14748.95	245.2904	244.3141	255.6241	220.4667	310.3541
Sonar	16.57394	186.1112	46.1303	218.9721	3.555642	3.44155	3.552653	7.471765	6.451212
PenglungEW	23.63256	248.9183	43.6011	285.8581	4.14005	4.148993	3.490046	8.047539	9.397815

section, additional experiments are performed to check how AIEOU can scale to high dimensional datasets which are challenging binary optimization problems.

Microarray datasets are high dimensional datasets and FS becomes difficult due to the existence of an extremely large search space. Hence, these datasets are used extensively to test the robustness of any FS model. For this experimentation, three publicly available Microarray datasets have been considered details of which are provided in Table 10.

The classification accuracies obtained over the Microarray datasets Ghosh et al. (2018) have further been compared with some classical as well as recently proposed meta-heuristics such as: GA, Memetic Algorithm (MA) Moscato and Porras (2003), PSO, Ant Lion Optimizer (ALO) Mirjalili (2015), GSA, SSD-LAHC, ECWSA and Adaptive  $\beta$  Coral-Reefs Optimization (A $\beta$ CRO) Ahmed et al. (2020a). The results of these algorithms are reported in Guha et al. (2020); Chatterjee et al. (2020). Table 11 contains the classification accuracy (in %) obtained by these algorithms over the utilized microarray datasets and the number of features used to achieve the accuracy has been provided in parenthesis corresponding to the accuracy.

## 9. Conclusion

Feature selection is an important and fundamental pre-processing phase in the data related domains. In the past few decades, several meta-heuristic algorithms have been proposed to select the most relevant features from various high dimensional datasets. These methods have proved their utility in this field by producing promising results. EO has an efficient exploration capability because of its equilibrium pool. The present work proposes an improved version of EO by improving its equilibrium pool with A $\beta$ HC and finding proper values for its parameters with learning-based automata. As feature selection is a binary optimization problem, U-shaped transfer function is used to limit the output to 0 and 1. The proposed method is named as AIEOU. To establish the effectiveness and superiority of AIEOU, it is evaluated on 18 standard UCI datasets and compared with 8 state-of-the-art FS methods. The achieved results demonstrate that AIEOU is an appropriate choice for FS problem.

As AIEOU is a meta-heuristic algorithm, there can be some research problems where it may fail to

find the optimal solution. Although this is in accordance with *NFL* theorem Wolpert and Macready (1997), it can still be considered as a disadvantage of using the suggested method. The  $\delta$  values (mentioned in section 5) for the parameters of EO can be chosen differently for different datasets. AIEOU can also be applied to some real-world problems like handwritten word or digit recognition, human activity recognition, facial emotion recognition, script recognition, graphology applications, sleep deprivation detection, etc. Further analysis of impact of use of other classifiers like neural networks or random forest can also be made.

## Conflict of Interest

This work has no conflict of interest.

## References

- Abdel-Basset, M., Abdel-Fatah, L., Sangaiah, A.K., 2018. Metaheuristic algorithms: A comprehensive review, in: Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications. Elsevier, pp. 185–231. doi:[10.1016/b978-0-12-813314-9.00010-4](https://doi.org/10.1016/b978-0-12-813314-9.00010-4).
- Abdel-Basset, M., Chang, V., Mohamed, R., 2020. A novel equilibrium optimization algorithm for multi-thresholding image segmentation problems. Neural Computing and Applications doi:[10.1007/s00521-020-04820-y](https://doi.org/10.1007/s00521-020-04820-y).
- Agwa, A., 2020. Equilibrium optimization algorithm for automatic generation control of interconnected power systems. Przegląd Elektrotechniczny 96, 143–148. doi:[10.15199/48.2020.09.30](https://doi.org/10.15199/48.2020.09.30).
- Ahmed, S., Ghosh, K.K., Garcia-Hernandez, L., Abraham, A., Sarkar, R., 2020a. Improved coral reefs optimization with adaptive  $\beta$ -hill climbing for feature selection. Neural Computing and Applications doi:[10.1007/s00521-020-05409-1](https://doi.org/10.1007/s00521-020-05409-1).
- Ahmed, S., Ghosh, K.K., Singh, P.K., Geem, Z.W., Sarkar, R., 2020b. Hybrid of harmony search algorithm and ring theory-based evolutionary algorithm for feature selection. IEEE Access 8, 102629–102645. doi:[10.1109/access.2020.2999093](https://doi.org/10.1109/access.2020.2999093).
- Al-Betar, M.A., 2016.  $\beta$ -hill climbing: an exploratory local search. Neural Computing and Applications 28, 153–168. doi:[10.1007/s00521-016-2328-2](https://doi.org/10.1007/s00521-016-2328-2).
- Al-Betar, M.A., Aljarah, I., Awadallah, M.A., Faris, H., Mirjalili, S., 2019. Adaptive  $\beta$ -hill climbing for optimization. Soft Computing 23, 13489–13512. doi:[10.1007/s00500-019-03887-7](https://doi.org/10.1007/s00500-019-03887-7).
- Almuallim, H., Dietterich, T.G., 1991. Learning with many irrelevant features., in: AAAI, Citeseer. pp. 547–552.
- Atashpaz-Gargari, E., Lucas, C., 2007. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition, in: 2007 IEEE Congress on Evolutionary Computation, IEEE. doi:[10.1109/cec.2007.4425083](https://doi.org/10.1109/cec.2007.4425083).
- Beyer, H.G., Schwefel, H.P., 2002. Natural Computing 1, 3–52. doi:[10.1023/a:1015059928466](https://doi.org/10.1023/a:1015059928466).

Table 10: Description of the Microarray datasets used for experimentation

Sl. No.	Dataset	#Instances	#Features	#Classes
1	DLBCL	77	7070	2
2	Leukemia	72	5147	2
3	SRBCT	83	2308	4

Table 11: Performance comparison of the proposed method with some existing methods on Microarray datasets

Dataset	AIEOU	GA	MA	PSO	ALO	GSA	LAHC-SSD	ECWSA	A $\beta$ CRO
DLBCL	100(37)	97(88)	96.4(105)	96.2(90)	95.1(92)	95.7(59)	96(60)	96(57)	97.6(52)
Leukemia	100(33)	95.8(85)	97(65)	97(50)	97.2(70)	97.6(80)	94(61)	95.2(50)	98.4(47)
SRBCT	100(34)	100(78)	100(50)	99.1(49)	98.7(45)	100(62)	99.2(54)	100(40)	100(37)

- Chandrashekar, G., Sahin, F., 2014. A survey on feature selection methods. *Computers & Electrical Engineering* 40, 16–28. doi:[10.1016/j.compeleceng.2013.11.024](https://doi.org/10.1016/j.compeleceng.2013.11.024).
- Chatterjee, B., Bhattacharyya, T., Ghosh, K.K., Singh, P.K., Geem, Z.W., Sarkar, R., 2020. Late acceptance hill climbing based social ski driver algorithm for feature selection. *IEEE Access* 8, 75393–75408. doi:[10.1109/ACCESS.2020.2988157](https://doi.org/10.1109/ACCESS.2020.2988157).
- Cui, Z., Cai, X., 2013. Artificial plant optimization algorithm, in: *Swarm Intelligence and Bio-Inspired Computation*. Elsevier, pp. 351–365. doi:[10.1016/b978-0-12-405163-8.00016-8](https://doi.org/10.1016/b978-0-12-405163-8.00016-8).
- Das, S., Biswas, A., Dasgupta, S., Abraham, A., 2009. Bacterial foraging optimization algorithm: Theoretical foundations, analysis, and applications, in: *Foundations of Computational Intelligence Volume 3*. Springer Berlin Heidelberg, pp. 23–55. doi:[10.1007/978-3-642-01085-9\\_2](https://doi.org/10.1007/978-3-642-01085-9_2).
- Dash, M., Liu, H., 1997. Feature selection for classification. *Intelligent Data Analysis* 1, 131–156. doi:[10.3233/IDA-1997-1302](https://doi.org/10.3233/IDA-1997-1302).
- Dorigo, M., Maniezzo, V., Colomi, A., 1996. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)* 26, 29–41. doi:[10.1109/3477.484436](https://doi.org/10.1109/3477.484436).
- Dua, D., Graff, C., 2017. UCI machine learning repository. URL: <http://archive.ics.uci.edu/ml>.
- Dutta, T., Bhattacharyya, S., Dey, S., Platos, J., 2020. Border collie optimization. *IEEE Access* 8, 109177–109197. doi:[10.1109/access.2020.2999540](https://doi.org/10.1109/access.2020.2999540).
- Faramarzi, A., Heidarinejad, M., Stephens, B., Mirjalili, S., 2020. Equilibrium optimizer: A novel optimization algorithm. *Knowledge-Based Systems* 191, 105190. doi:[10.1016/j.knsys.2019.105190](https://doi.org/10.1016/j.knsys.2019.105190).
- Fister Jr, I., Yang, X.S., Fister, I., Brest, J., Fister, D., 2013. A brief review of nature-inspired algorithms for optimization. *arXiv preprint arXiv:1307.4186* URL: <https://arxiv.org/abs/1307.4186>.
- Friedman, M., 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association* 32, 675–701. doi:[10.1080/01621459.1937.10503522](https://doi.org/10.1080/01621459.1937.10503522).
- Geem, Z.W., Kim, J.H., Loganathan, G., 2001. A new heuristic optimization algorithm: Harmony search. *SIMULATION* 76, 60–68. doi:[10.1177/003754970107600201](https://doi.org/10.1177/003754970107600201).
- Gendreau, M., Potvin, J.Y., 2005. Metaheuristics in combinatorial optimization. *Annals of Operations Research* 140, 189–213. doi:[10.1007/s10479-005-3971-7](https://doi.org/10.1007/s10479-005-3971-7).
- Ghaemi, M., Feizi-Derakhshi, M.R., 2014. Forest optimization algorithm. *Expert Systems with Applications* 41, 6676–6687. doi:[10.1016/j.eswa.2014.05.009](https://doi.org/10.1016/j.eswa.2014.05.009).
- Ghosh, K.K., Ahmed, S., Singh, P.K., Geem, Z.W., Sarkar, R., 2020a. Improved binary sailfish optimizer based on adaptive  $\beta$ -hill climbing for feature selection. *IEEE Access* 8, 83548–83560. doi:[10.1109/access.2020.2991543](https://doi.org/10.1109/access.2020.2991543).
- Ghosh, K.K., Singh, P.K., Hong, J., Geem, Z.W., Sarkar, R., 2020b. Binary social mimic optimization algorithm with x-shaped transfer function for feature selection. *IEEE Access* 8, 97890–97906. doi:[10.1109/access.2020.2996611](https://doi.org/10.1109/access.2020.2996611).
- Ghosh, M., Adhikary, S., Ghosh, K.K., Sardar, A., Begum, S., Sarkar, R., 2018. Genetic algorithm based cancerous gene identification from microarray data using ensemble of filter methods. *Medical & Biological Engineering & Computing* 57, 159–176. doi:[10.1007/s11517-018-1874-4](https://doi.org/10.1007/s11517-018-1874-4).
- Guha, R., Ghosh, M., Mutsuddi, S., Sarkar, R., Mirjalili, S., 2020. Embedded chaotic whale survival algorithm for filter-wrapper feature selection. *Soft Computing* 24, 12821–12843. doi:[10.1007/s00500-020-05183-1](https://doi.org/10.1007/s00500-020-05183-1).
- Hall, M.A., Smith, L.A., 1999. Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper., in: *FLAIRS conference*, pp. 235–239.
- Hashemi, A., Meybodi, M., 2011. A note on the learning automata based algorithms for adaptive parameter selection in PSO. *Applied Soft Computing* 11, 689–705. doi:[10.1016/j.asoc.2009.12.030](https://doi.org/10.1016/j.asoc.2009.12.030).
- Hatamlou, A., 2013. Black hole: A new heuristic optimization approach for data clustering. *Information Sciences* 222, 175–184. doi:[10.1016/j.ins.2012.08.023](https://doi.org/10.1016/j.ins.2012.08.023).
- He, X., Cai, D., Niyogi, P., 2006. Laplacian score for feature selection, in: *Advances in neural information processing systems*, pp. 507–514.
- John, G.H., Kohavi, R., Pfleger, K., 1994. Irrelevant features and the subset selection problem, in: *Machine Learning Proceedings 1994*. Elsevier, pp. 121–129. doi:[10.1016/b978-1-55860-335-6.50023-4](https://doi.org/10.1016/b978-1-55860-335-6.50023-4).
- Karaboga, D., Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization* 39, 459–471. doi:[10.1007/s10898-007-9149-x](https://doi.org/10.1007/s10898-007-9149-x).
- Kaveh, A., 2016. Charged system search algorithm, in: *Advances in Metaheuristic Algorithms for Optimal Design of Structures*. Springer International Publishing, pp. 45–89. doi:[10.1007/978-3-319-46173-1\\_3](https://doi.org/10.1007/978-3-319-46173-1_3).
- Kennedy, J., Eberhart, R., 1995. Particle swarm optimization, in: *Proceedings of ICNN'95 - International Conference on Neural Networks*, pp. 1942–1948 vol.4.

- doi:[10.1109/ICNN.1995.488968](https://doi.org/10.1109/ICNN.1995.488968).
- Kira, K., Rendell, L.A., 1992. A practical approach to feature selection, in: Machine Learning Proceedings 1992. Elsevier, pp. 249–256. doi:[10.1016/b978-1-55860-247-2.50037-1](https://doi.org/10.1016/b978-1-55860-247-2.50037-1).
- Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. *Science* 220, 671–680. doi:[10.1126/science.220.4598.671](https://doi.org/10.1126/science.220.4598.671).
- Kohavi, R., Sommerfeld, D., 2001. Feature subset selection using the wrapper method: Overfitting and dynamic search space topology. *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*.
- Koller, D., Sahami, M., 1996a. Toward optimal feature selection. Technical Report. Stanford InfoLab.
- Koller, D., Sahami, M., 1996b. Toward Optimal Feature Selection. Technical Report 1996-77. Stanford InfoLab. URL: <http://ilpubs.stanford.edu:8090/208/>. previous number = SIDL-WP-1996-0032.
- Leardi, R., 1994. Application of a genetic algorithm to feature selection under full validation conditions and to outlier detection. *Journal of Chemometrics* 8, 65–79. doi:[10.1002/cem.1180080107](https://doi.org/10.1002/cem.1180080107).
- Mafarja, M., Qasem, A., Heidari, A.A., Aljarah, I., Faris, H., Mirjalili, S., 2019. Efficient hybrid nature-inspired binary optimizers for feature selection. *Cognitive Computation* 12, 150–175. doi:[10.1007/s12559-019-09668-6](https://doi.org/10.1007/s12559-019-09668-6).
- Mafarja, M.M., Mirjalili, S., 2017. Hybrid whale optimization algorithm with simulated annealing for feature selection. *Neurocomputing* 260, 302–312. doi:[10.1016/j.neucom.2017.04.053](https://doi.org/10.1016/j.neucom.2017.04.053).
- Mahdavi, M., Fesanghary, M., Damangir, E., 2007. An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation* 188, 1567–1579. doi:[10.1016/j.amc.2006.11.033](https://doi.org/10.1016/j.amc.2006.11.033).
- Mirjalili, S., 2015. The ant lion optimizer. *Advances in Engineering Software* 83, 80–98. doi:[10.1016/j.advengsoft.2015.01.010](https://doi.org/10.1016/j.advengsoft.2015.01.010).
- Mirjalili, S., 2016. SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems* 96, 120–133. doi:[10.1016/j.knsys.2015.12.022](https://doi.org/10.1016/j.knsys.2015.12.022).
- Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H., Mirjalili, S.M., 2017. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software* 114, 163–191. doi:[10.1016/j.advengsoft.2017.07.002](https://doi.org/10.1016/j.advengsoft.2017.07.002).
- Mirjalili, S., Lewis, A., 2013. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation* 9, 1–14. doi:[10.1016/j.swevo.2012.09.002](https://doi.org/10.1016/j.swevo.2012.09.002).
- Mirjalili, S., Lewis, A., 2016. The whale optimization algorithm. *Advances in Engineering Software* 95, 51–67. doi:[10.1016/j.advengsoft.2016.01.008](https://doi.org/10.1016/j.advengsoft.2016.01.008).
- Mirjalili, S., Mirjalili, S.M., Hatamlou, A., 2015. Multi-verse optimizer: a nature-inspired algorithm for global optimization. *Neural Computing and Applications* 27, 495–513. doi:[10.1007/s00521-015-1870-7](https://doi.org/10.1007/s00521-015-1870-7).
- Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf optimizer. *Advances in engineering software* 69, 46–61. doi:[10.1016/j.advengsoft.2013.12.007](https://doi.org/10.1016/j.advengsoft.2013.12.007).
- Mirjalili, S., Zhang, H., Mirjalili, S., Chalup, S., Noman, N., 2020. A novel u-shaped transfer function for binary particle swarm optimisation, in: *Advances in Intelligent Systems and Computing*. Springer Singapore, pp. 241–259. doi:[10.1007/978-981-15-3290-0\\_19](https://doi.org/10.1007/978-981-15-3290-0_19).
- Molina, D., Poyatos, J., Ser, J.D., García, S., Hussain, A., Herrera, F., 2020. Comprehensive taxonomies of nature- and bio-inspired optimization: Inspiration versus algorithmic behavior, critical analysis and recommendations. *ArXiv abs/2002.08136*.
- Monismith, D.R., Mayfield, B.E., 2008. Slime mold as a model for numerical optimization, in: 2008 IEEE Swarm Intelligence Symposium, IEEE. doi:[10.1109/sis.2008.4668295](https://doi.org/10.1109/sis.2008.4668295).
- Moosavian, N., Roodsari, B.K., 2014. Soccer league competition algorithm: A novel meta-heuristic algorithm for optimal design of water distribution networks. *Swarm and Evolutionary Computation* 17, 14–24. doi:[10.1016/j.swevo.2014.02.002](https://doi.org/10.1016/j.swevo.2014.02.002).
- Moscato, P., Porras, C.C., 2003. An introduction to memetic algorithms. *INTELIGENCIA ARTIFICIAL* 7. doi:[10.4114/ia.v7i19.721](https://doi.org/10.4114/ia.v7i19.721).
- Narendra, K.S., Thathachar, M.A.L., 1974. Learning automata - a survey. *IEEE Transactions on Systems, Man, and Cybernetics SMC-4*, 323–334. doi:[10.1109/tsmc.1974.5408453](https://doi.org/10.1109/tsmc.1974.5408453).
- Özkaya, H., Yıldız, M., Yıldız, A.R., Bureerat, S., Yıldız, B.S., Sait, S.M., 2020. The equilibrium optimization algorithm and the response surface-based metamodel for optimal structural design of vehicle components. *Materials Testing* 62, 492–496. doi:[10.3139/120.111509](https://doi.org/10.3139/120.111509).
- Pohlert, T., 2014. The pairwise multiple comparison of mean ranks package (pmcmr). *R package* 27, 9.
- Pudil, P., Novovičová, J., Kittler, J., 1994. Floating search methods in feature selection. *Pattern Recognition Letters* 15, 1119–1125. doi:[10.1016/0167-8655\(94\)90127-9](https://doi.org/10.1016/0167-8655(94)90127-9).
- Punnathanam, V., Kotecha, P., 2016. Yin-yang-pair optimization: A novel lightweight optimization algorithm. *Engineering Applications of Artificial Intelligence* 54, 62–79. doi:[10.1016/j.engappai.2016.04.004](https://doi.org/10.1016/j.engappai.2016.04.004).
- Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S., 2009. Gsa: a gravitational search algorithm. *Information sciences* 179, 2232–2248. doi:[10.1016/j.ins.2009.03.004](https://doi.org/10.1016/j.ins.2009.03.004).
- Salcedo-Sanz, S., Ser, J.D., Landa-Torres, I., Gil-López, S., Portilla-Figueras, J.A., 2014. The coral reefs optimization algorithm: A novel metaheuristic for efficiently solving optimization problems. *The Scientific World Journal* 2014, 1–15. doi:[10.1155/2014/739768](https://doi.org/10.1155/2014/739768).
- Shi, Y., 2011. Brain storm optimization algorithm, in: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 303–309. doi:[10.1007/978-3-642-21515-5\\_36](https://doi.org/10.1007/978-3-642-21515-5_36).
- Storn, R., Price, K., 1997. *Journal of Global Optimization* 11, 341–359. doi:[10.1023/a:1008202821328](https://doi.org/10.1023/a:1008202821328).
- Talbi, E.G., 2009. *Metaheuristics: from design to implementation*. volume 74. John Wiley & Sons.
- Thathachar, M., Sastry, P., 2002. Varieties of learning automata: an overview. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)* 32, 711–722. doi:[10.1109/tsmcb.2002.1049606](https://doi.org/10.1109/tsmcb.2002.1049606).
- Tilahun, S.L., Ong, H.C., 2015. Prey-predator algorithm: A new metaheuristic algorithm for optimization problems. *International Journal of Information Technology & Decision Making* 14, 1331–1352. doi:[10.1142/s021962201450031x](https://doi.org/10.1142/s021962201450031x).
- Ting, T.O., Man, K.L., Guan, S.U., Nayel, M., Wan, K., 2012. Weightless swarm algorithm (WSA) for dynamic optimization problems, in: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 508–515. doi:[10.1007/978-3-642-35606-3\\_60](https://doi.org/10.1007/978-3-642-35606-3_60).
- Westfall, P., 1997. A bayesian perspective on the bonfer-

- roni adjustment. *Biometrika* 84, 419–427. doi:[10.1093/biomet/84.2.419](https://doi.org/10.1093/biomet/84.2.419).
- Wolpert, D.H., Macready, W.G., 1997. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation* 1, 67–82. doi:[10.1109/4235.585893](https://doi.org/10.1109/4235.585893).
- Xue, B., Zhang, M., Browne, W.N., Yao, X., 2016. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation* 20, 606–626. doi:[10.1109/TEVC.2015.2504420](https://doi.org/10.1109/TEVC.2015.2504420).
- Xue, Y., Xue, B., Zhang, M., 2019. Self-adaptive particle swarm optimization for large-scale feature selection in classification. *ACM Transactions on Knowledge Discovery from Data* 13, 1–27. doi:[10.1145/3340848](https://doi.org/10.1145/3340848).
- Yang, X.S., Deb, S., 2009. Cuckoo search via lévy flights, in: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), IEEE. doi:[10.1109/nabic.2009.5393690](https://doi.org/10.1109/nabic.2009.5393690).
- Zhang, Y., wei Gong, D., Cheng, J., 2017. Multi-objective particle swarm optimization approach for cost-based feature selection in classification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 14, 64–75. doi:[10.1109/tcbb.2015.2476796](https://doi.org/10.1109/tcbb.2015.2476796).
- Zheng, Z., Wu, X., Srihari, R., 2004. Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter* 6, 80–89. doi:[10.1145/1007730.1007741](https://doi.org/10.1145/1007730.1007741).

**Credit author statement**

**Shameem Ahmed:** Conceptualization, Methodology, Writing- Original draft, Software, Investigation

**Kushal Kanti Ghosh:** Writing- Reviewing and Editing, Software, Investigation, Conceptualization

**Seyedali Mirjalili:** Writing- Reviewing and Editing, Supervision, Project administration, Validation

**Ram Sarkar:** Writing- Reviewing and Editing, Supervision, Project administration, Validation, Funding acquisition

## **AUTHOR DECLARATION TEMPLATE**

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He/she is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs. We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author and which has been configured to accept email from [ali.mirjalili@gmail.com](mailto:ali.mirjalili@gmail.com)