Discrete Equilibrium Optimizer Combined with Simulated Annealing for Feature Selection

Ritam Guha

Jadavpur University <https://orcid.org/0000-0002-1375-777X>

Kushal Kanti Ghosh

Jadavpur University <https://orcid.org/0000-0003-0929-5928>

Suman Kumar Bera

Jadavpur University

Ram Sarkar

Jadavpur University <https://orcid.org/0000-0001-8813-4086>

Seyedali Mirjalili (✉ seyedali.mirjalili@griffithuni.edu.au)

Torrens University Australia <https://orcid.org/0000-0002-1443-9458>

Research Article

Keywords: Equilibrium optimizer, Feature selection, Simulated annealing, Knapsack problem, UCI dataset, Microarray dataset

Posted Date: January 5th, 2022

DOI: <https://doi.org/10.21203/rs.3.rs-28683/v2>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Discrete Equilibrium Optimizer Combined with Simulated Annealing for Feature Selection

Ritam Guha^a, Kushal Kanti Ghosh^{a,*}, Suman Kumar Bera^a, Seyedali Mirjalili^{b,c,*}, Ram Sarkar^a

^a*Department of Computer Science and Engineering, Jadavpur University, Kolkata, India - 700032*

^b*Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Fortitude Valley, Brisbane, 4006 QLD, Australia*

^c*YFL (Yonsei Frontier Lab), Yonsei University, Seoul, Korea*

Abstract

This paper proposes a binary adaptation of the recently proposed meta-heuristic, Equilibrium Optimizer (EO), called Discrete EO (DEO) to solve binary optimization problems. A U-shaped transfer function has been used to map the continuous values of EO into the binary domain. To further improve the exploitation capability of DEO, Simulated Annealing (SA) has been used as a local search procedure and the combination has been named as DEOSA. The proposed DEOSA algorithm has been applied over 18 well-known UCI datasets and compared with a wide range of algorithms. The results have been statistically validated using Wilcoxon rank-sum test. In order to test the scalability and robustness of DEOSA, it has been additionally tested over 7 high-dimensional Microarray datasets and 25 binary Knapsack problems. The results clearly demonstrate the superiority and merits of DEOSA when solving binary optimization problems.

Keywords: Equilibrium optimizer, Feature selection, Simulated annealing, Knapsack problem, UCI dataset, Microarray dataset

[☆]This document is a collaborative effort.

^{*}Corresponding author

Email addresses: ritamguha16@gmail.com (Ritam Guha), kushalkanti1999@gmail.com (Kushal Kanti Ghosh), berasuman007@gmail.com (Suman Kumar Bera), ali.mirjalili@gmail.com (Seyedali Mirjalili), ramjucse@gmail.com (Ram Sarkar)

1. Introduction

With the invention of modern hi-tech sophisticated devices, it has become possible to generate tons of data within a very short interval. However, there is a two-fold question: do we need the entire set of data to retrieve some useful information, and is it always feasible to analyze such a huge quantity of data? Statisticians may help to answer these questions. According to [Petrov \(2020\)](#), by the end of 2025, there will be around 180 zettabytes of data which is a staggering number and 90% of this data have been generated in the last two years. As a matter of fact, only 0.5% of data have been analyzed in the year 2012, and the percentage must have decreased more in recent times. One of the reasons for this small percentage is due to the lack of enough tools to analyze this enormous quantity of data. This has increased the need for data processing, and among which one important technique is known as dimension reduction [Chizi & Maimon \(2005\)](#). The motive of dimension reduction is to extract the meaningful information from a large dataset while reducing the number of dimensions of the dataset under consideration.

One of the most sought-after dimension reduction procedures is feature selection (FS) [Ghosh et al. \(2018a\)](#). The objective of FS is to search for the most optimal set of useful and relevant features from the entire feature vector. It is a difficult research problem and its difficulty increases exponentially proportional to the number of features in a feature set. For a feature set with n number of features, a total of 2^n feature combinations are possible, and selecting one combination out of these 2^n combinations becomes extremely complex when the value of n increases significantly. FS algorithms try to search for a significantly good and acceptable solution within a suitable time-bound.

FS procedures can be categorized into two different variants: wrapper and filter. Filter methods look for statistical interpretations of data to find the most informative and appropriate subset of features. Wrapper methods, on the

other hand, use learning algorithms (e.g. classifiers [Allan \(1977\)](#)) to evaluate candidate feature subsets and guide the next round of searches according to the evaluation outcomes. Although wrapper methods are more resource intensive (require lots of processing power) than filter methods (due to not requiring learning algorithms), they are able to find superior solutions in comparison to filter methods. In order to improve performance of such algorithms, a recent trend is to blend multiple algorithms to combine advantages of the individual algorithms. Such hybrid algorithms tend to perform better by improving their exploration and exploitation ability (through inclusion of local or global search techniques). Some researchers also use a combination of filter and wrapper to reach a better solution by using advantages of both the models. These models are known as embedded models [Ghosh et al. \(2019a\)](#); [Guyon et al. \(2005\)](#). The quality of such models depends a lot on how wrapper and filter parts interact with each other which requires an extra level of tuning leading to higher computational complexity.

The wide use and superiority of hybrid methods in the domain of FS have intrigued us to develop a new hybrid procedure that can efficiently perform search operations in the binary search space. In this paper, we have made an effort to address the FS problem using a hybrid variant of a recently proposed optimization algorithm known as Equilibrium Optimizer (EO) [Faramarzi et al. \(2020\)](#). It mimics the process of control volume mass balance used to predict equilibrium and dynamic phases, where the equilibrium state is treated as the optimal solution to the optimization problem. FS can also be considered as an optimization problem, where the goal is to find an optimal feature subset subject to constraints like high classification accuracy and a low number of features. This intuitive similarity between FS and optimization problems has motivated us to modify the EO and apply it to solve FS problems. The highlights of this paper are mentioned below:

1. Modification of EO by mapping continuous values into the binary domain

using a U-shaped transfer function, thereby making it applicable to binary optimization problems.

2. Enhancement of EO's exploitative abilities through the use of Simulated Annealing (SA). The hybrid version is known as DEOSA.
3. Validation of the proposed hybrid FS framework (DEOSA) over 18 well-known UCI datasets.
4. Additional experimentation of DEOSA through applications over 7 high-dimensional Microarray datasets and 25 binary Knapsack datasets in order to test the robustness and scalability of the proposed model.

The remaining paper is structured in the following format: Section 2 provides a detailed literature study of the similar kinds of work carried out by different researchers across the globe. Section 3 provides a comprehensive illustration of the proposed FS model. The experimental outcomes of DEOSA over the UCI datasets are explained in Section 4 while the outcomes of the additional experiments are reported in Section 5. In the end, Section 6 concludes our work and provides directions for future extension of this work.

2. Literature Study

It is quite interesting how solutions to many complex optimization problems are lying hidden in nature waiting to be discovered by us. Throughout the years, researchers have proposed various algorithms to solve FS problems getting inspired by natural phenomena. Starting from the concept of chromosome formation in the Genetic Algorithm (GA) to the concept of control volume mass balance in EO, there is a large number of algorithms that are derived from simple natural occurrences.

The journey of nature-inspired FS algorithms started with GA [Leardi \(1996\)](#). Leardi et al. introduced GA to the domain of FS by using 0 and 1 to denote the absence and presence of a feature (variable) in a chromosome (agent). GA is one of the most popular evolutionary algorithms which mimics the procedure

of child chromosome formation using the concepts of chromosome crossover and mutation. While crossover in GA tries to improve exploration, mutation provides perturbation of the solution to bring exploitation in the search space. Due to its simplicity in nature, GA has been widely used in various optimization and FS problems. In search of a proper trade-off between exploration and exploitation, researchers have proposed different variants of GA. In [Ma & Xia \(2017\)](#), Ma et al. have divided the chromosome population in different tribes and introduced tribe competition to support the evolution process. A modified GA (MGA) has been proposed in [Jiang et al. \(2017\)](#) where the authors have updated certain operations of GA to provide guidance to the chromosomes. For example, in place of typical crossover, they have proposed a crossover procedure which is guided by the objective measures of the parent chromosomes. MGA has been used to perform FS before demand forecasting in outpatient department. Apart from these recent additions, there are numerous other work on GA which can be found in [Yang & Honavar \(1998\)](#); [Huang et al. \(2007\)](#); [Oh et al. \(2004\)](#); [SIEDLECKI & SKLANSKY \(1993\)](#); [Ghosh et al. \(2018a,b\)](#).

After the application of GA in FS, a large number of other nature-inspired optimization algorithms have been modified to solve FS problems. Meta-heuristic algorithms like Ant Colony Optimization (ACO), Gravitational Search Algorithm (GSA), Particle Swarm Optimization (PSO), etc. have gained huge popularity among researchers because of their intuitive simplicity and competent optimization capabilities. PSO is the brainchild of Kennedy and Eberhart. In [Kennedy & Eberhart \(1995\)](#), they have proposed PSO trying to replicate the motion of organisms like birds in a flock or fishes in a school. Different variations of PSO have been proposed by researchers over the time [Tran et al. \(2019\)](#); [Shang et al. \(2016\)](#); [Moradi & Gholampour \(2016\)](#). ACO is based on the food searching process followed by ants in nature. Dorigo et al. have introduced ACO in [Dorigo & Di Caro \(1999\)](#). Rashedi et al. have developed GSA [Rashedi et al. \(2009\)](#) which is inspired by mass interactions and the law of gravity.

All these algorithms are very popular and highly used in the FS domain. As per the *No-free Lunch* theorem related to optimization, however, all optimization procedures behave equally well when they are averaged over all the problems [Joyce & Herrmann \(2017\)](#); [Gómez & Rojas \(2016\)](#); [Adam et al. \(2019\)](#). Intuitively it means that there is no perfect algorithm to solve all optimization problems [Wolpert et al. \(1995\)](#). If algorithm A outperforms another algorithm B in some areas, there must be some other areas where algorithm B outperforms algorithm A. That is why researchers continue to propose various new algorithms to solve different optimization problems including the FS problem. One of the main challenges in any FS problem is to find proper stability between exploitation and exploration. Many FS algorithms are getting formulated on a regular basis to address this balance.

Even in recent times, researchers have found enormous inspiration from nature to formulate various other meta-heuristic algorithms for the FS problem. Binary Grey Wolf Optimizer (bGWO) [Emary et al. \(2016b\)](#) tries to mimic the hunting strategies and leadership ranking in a pack of grey wolves. Whale Optimization Algorithm (WOA) proposed in [Mirjalili & Lewis \(2016\)](#) has been modified in [Hussien et al. \(2018\)](#) using an S-shaped function to adapt it in FS domain. WOA follows the bubble-net hunting strategy of humpback whales using shrinking encircling and spiral movements to perform optimization or FS procedures. Emary et al. [Emary et al. \(2016a\)](#) have proposed the binary variant of Ant Lion Optimizer (ALO) [Mirjalili \(2015\)](#) which uses the five hunting steps of antlions namely random walk, trap building, entrapment of ants, catching preys and finally rebuilding the traps. Various binary variants of Salp Swarm Algorithm (SSA) [Mirjalili et al. \(2017\)](#) have been proposed to perform FS [Sayed et al. \(2018\)](#); [Ibrahim et al. \(2017\)](#). SSA is basically based on the swarming behavior of salps observed in oceans. Binary Butterfly Optimization Algorithm (BBOA), proposed in [Arora & Anand \(2019\)](#), replicates the way butterflies move in search of food using their sense receptors (mainly known as chemoreceptors) to follow the fragrance of food/flowers.

Binary Thermal Exchange Optimization (BTEO), proposed in [Taradeh & Mafarja \(2019\)](#), is based on Newton’s law of cooling. A multi-objective FS variant of Artificial Bee Colony Optimization [Marinakis et al. \(2009\)](#) is proposed in [Zhang et al. \(2019\)](#) where the authors have introduced two new operators based on searches guided by convergence and diversity for employed and onlooker bees respectively. They have named the proposed algorithm as Two-archive Multi-objective ABC (TMABC-FS). Yan et al. have proposed a modified version of Coral Reefs Optimization (CRO) using tournament selection and SA [Yan et al. \(2019\)](#). Another hybridized version of CRO using Adaptive β Hill Climbing can be found in [Ahmed et al. \(2020\)](#). A chaotic version of Chicken Swarm Optimization has been proposed in [Ahmed et al. \(2017\)](#). Some of the researchers have also invested their effort to utilize common human behaviors to solve optimization problems. Social Mimic Optimizer [Ghosh et al. \(2020\)](#) is proposed following the human nature of mimicking more successful humans. For example, Yuhui Shi has acquired a pattern from human brain storming process and developed Brain Storm Optimization (BSO) algorithm in [Shi \(2011\)](#). In 2019, the algorithm has been modified by Zhang et al. to make it applicable in FS [Zhang et al. \(2019\)](#). Apart from the aforementioned algorithms, many more meta-heuristic algorithms have been formulated getting inspired from various natural phenomena. Bat Algorithm [Gupta et al. \(2020\)](#), Tree Growth Algorithm (TGA) [Cheraghalipour et al. \(2018\)](#), Garter Snake Optimization (GSO) [Naghadiani & Jahanshahi \(2017\)](#), AntStar [Faisal et al. \(2016\)](#), Locust Search (LS) [Cuevas et al. \(2015\)](#), Social Spider Algorithm (SSA) [Yu & Li \(2015\)](#), Fruit Fly Optimization (FFO) [Pan \(2012\)](#), Emperor Penguins Colony (EPC) [Harifi et al. \(2019\)](#), Artificial Feeding Birds (AFB) [Lamy \(2018\)](#), Crow Search Algorithm (CSA) [Zolghadr-Asli et al. \(2017\)](#), Spotted Hyena Optimizer (SHO) [Dhiman & Kumar \(2018\)](#), Sailfish Optimizer [Shadravan et al. \(2019\)](#); [Ghosh et al. \(2020\)](#), Water Wave Optimization (WWO) [Zhang et al. \(2018b\)](#), Queuing Search Algorithm [Zhang et al. \(2018a\)](#), are few of such algorithms proposed in the last decade. All these algorithms try to find a near-optimal solutions of the

problem under consideration using different mathematical expressions imitating various processes occurring as parts of a natural phenomenon.

EO is one of the most recent algorithms in the domain of optimization. Faramarzi et al. have proposed EO in [Faramarzi et al. \(2020\)](#). EO uses the concept of control volume mass balance where each particle represents a solution and its concentration represents the position. The best solutions are termed, equilibrium candidates. Other candidates update their concentration based on the equilibrium candidates to reach the final equilibrium state which can be considered as the optimal solution of an optimization problem. The EO algorithm was tested on 58 benchmark problems and three engineering case studies. EO outperformed a wide range of algorithms including GA, PSO, SSA, GA, and GWO. In [Menesy et al. \(2020\)](#), EO has been used to extract the best combination of parameter values for the mathematical model of Proton Exchange Membrane Fuel Cell (PEMFC). The accuracy of the model is computed using the squared error between the experimental voltage and the voltage obtained by the EO-based PEMFC model. Seven important parameters are estimated using different optimization procedures like Shark Smell Optimizer (SSO), ALO, JAYA algorithm and Nelder-Mead simplex method (JAYA-NM), Hybrid GA (HGA), Teaching Learning Based Optimization - Differential Evolution (TLBO-DE), GWO. In every scenario, EO has outperformed the other algorithms.

In regards to the existing binary variants of EO in the literature, there are two popular methods, where one uses s-shaped [Gao et al. \(2020\)](#) and the other uses V-shaped [Zhao & Gao \(2020\)](#) transfer functions. But recently a new U-shaped transfer function has emerged in the domain of FS [Ahmed et al. \(2021\)](#); [Mirjalili et al. \(2020\)](#) that has surpassed the performance of the pre-mentioned two transfer functions. Depending on the basin width and height of the U, the exploration and exploitation range can be controlled through the parameters of the function. This flexibility and some other properties make the U-shaped function a favorable choice for many researchers. For this reason, we have used

the U-shaped transfer function to map the continuous values of EO to the binary counterpart.

This has motivated us to modify EO and make it applicable to solve FS problems. In this paper, we have mapped EO to the binary space of FS using a U-shaped transfer function and hybridized it by utilizing the exploitative capabilities of SA. The final hybrid model termed DEOSA has been applied over 18 well-known datasets taken from UCI machine learning repository to prove its usefulness in the FS domain. On the other hand, as additional experimentation, it has been tested over 7 Microarray and 25 binary Knapsack datasets to prove its applicability in other binary optimization problems as well.

3. Proposed Work

3.1. Equilibrium Optimization: An Overview

EO, first proposed in [Faramarzi et al. \(2020\)](#), is a physics based optimization algorithm inspired by dynamic source and sink models for estimating equilibrium states. This method is based upon straightforward properly fused dynamic mass balance on a control volume, where a mass balance equation is used to determine the concentration of an impassible component in a control volume as a function of its different sink and source procedures. The generic equation of mass-balance is identified with a first order differential equation [Nazaroff & Alvarez-Cohen \(2001\)](#) given by Equation (1).

$$V \frac{dC}{dt} = QC_{eq} - QC + G \quad (1)$$

where V is the control volume, C is control volume concentration, C_{eq} is equilibrium state concentration without any production inside control volume, G is the mass generation rate, $V \frac{dC}{dt}$ denotes mass changing rate, Q is the flow rate, in volume, into and out of the control volume. Equation (1) indicates that change of mass in time is equivalent to the mass entering the system plus mass-generated inside minus mass leaving the system. An equilibrium state is reached

when $V \frac{dC}{dt}$ reaches zero. A rearranged Equation (1) is given in Equation (2).

$$\frac{dC}{\lambda C_{eq} - \lambda C + \frac{G}{V}} = dt \quad (2)$$

where $\lambda = \frac{Q}{V}$, implies turnover rate. Equation (3) indicates integration of Equation (2) over time.

$$\int_{C_0}^C \frac{dC}{\lambda C_{eq} - \lambda C + \frac{G}{V}} = \int_{t_0}^t dt \quad (3)$$

where e_0 and C_0 are initial start time and initial concentration. Equation (4) shows the result of Equation (3).

$$C = C_{eq} + (C_0 - C_{eq})F + \frac{G}{\lambda V}(1 - F) \quad (4)$$

$$F = e^{-\lambda(t-t_0)} \quad (5)$$

As followed by Equation (4), there are three terms representing the updating rule of the concentration of a solution. First-term is equilibrium concentration. One of the best solutions that appeared till the current iteration is used as equilibrium concentration and it is selected randomly from a pool, *namely* equilibrium pool. The second term indicates the concentration difference between a solution and equilibrium state. The third term is involved with the generation rate. Mathematical expressions and explanations of these components are given in the following subsections.

3.1.1. Equilibrium Pool and Candidates

The equilibrium state is the state in which the algorithm converges, and it is expected to be the global optima of the problem under consideration. In the beginning, the equilibrium concentration is not known to the algorithm, rather equilibrium candidates provide the particles with a search pattern. As per [Faramarzi et al. \(2020\)](#), we have considered an equilibrium pool consisting of five concentrations: four best-so-far concentrations obtained during the process and their arithmetic mean. These four concentrations help in the exploration capability of EO and the arithmetic mean helps in exploitation.

$$\vec{C}_{eq,pool} = \{\vec{C}_{eq(1)}, \vec{C}_{eq(2)}, \vec{C}_{eq(3)}, \vec{C}_{eq(4)}, \vec{C}_{eq(avg)}\} \quad (6)$$

3.1.2. Exponential Term, F

The term F in Equation (4) helps EO find a balance between exploration and exploitation. The turnover rate λ is a random vector in $[0, 1]$ since, in general, the turnover rate varies *w.r.t.* time in a real control volume and t is a function of iteration.

$$t = \left(1 - \frac{iter}{maxIter}\right)^{(a_2 \frac{iter}{maxIter})} \quad (7)$$

where $iter$ and $maxIter$ are current and maximum number of iterations, respectively. a_2 is a constant, higher a_2 indicates lower exploration and better exploitation ability.

$$\vec{F} = e^{-\vec{\lambda}(t-t_0)} \quad (8)$$

For guaranteed convergence, the search speed needs to be slowed down, and to perform that t_0 is considered as:

$$\vec{t}_0 = \frac{1}{\vec{\lambda}} \ln(-a_1 \text{sgn}(\vec{r} - 0.5)[1 - e^{-\vec{\lambda}t}]) + t \quad (9)$$

where, a_1 is a constant and higher a_1 value implies higher exploration ability. $\vec{r}$ is a random vector in $[0, 1]$. sgn function indicates the direction of the search process. As per Faramarzi et al. (2020), we have set $a_1 = 2$ and $a_2 = 1$. By substituting Equation (9) in Equation (8), we obtain:

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

3.1.3. Generation Rate, G

In EO, the generation rate (G) plays an important role to provide the optimum solution. G is calculated as:

$$\vec{G} = \vec{G}_0 e^{-\vec{\lambda}(t-t_0)} = \vec{G}_0 \vec{F} \quad (11)$$

where

$$\vec{G}_0 = \overrightarrow{GCP}(\vec{C}_{eq} - \vec{\lambda}\vec{C}) \quad (12)$$

$$GCP = \begin{cases} 0.5r_1 & r_2 \geq GP \\ 0 & r_2 < GP \end{cases} \quad (13)$$

$\overrightarrow{GCP}$ is obtained by repeating the value obtained from Equation (13). r_1 and r_2 are random numbers, $r_1, r_2 \in [0, 1]$. GCP implies a Generation rate Control Parameter, which indicates the probability of generating the term's contribution towards the concentration updating process. GP means Generation Probability, which indicates the probability that informs the number of particles using the last term in Equation (4) to update their states. $GP = 0.5$ is set balancing the exploitation and exploration Faramarzi et al. (2020). So, finally, the concentration updating rule in this algorithm is given by:

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

V is considered as unit, i.e. $V=1$.

3.2. Simulated Annealing

In metallurgy and materials science, annealing van Laarhoven & Aarts (1987) is a heat treatment in which a solid is heated up to a maximum temperature, where the solid becomes liquid and then it is cooled by slowly lowering the temperature. SA Kirkpatrick et al. (1983), inspired from the annealing process, is a single solution based meta-heuristic algorithm which is an enhanced version of the hill climbing Ackley (1987). SA uses a certain probability to accept a bad 'move' to overcome the problem of being trapped in locally optimal solutions. For a particular solution (agent), a neighboring solution is generated Mafarja & Mirjalili (2017) and evaluated using an objective function. If the objective value of the neighbor is better than the current solution, then the current solution is replaced with the neighbor. If the objective value of the neighbor is worse than the current solution, the neighbor is accepted with a probability value generated by the Boltzman equation, $p = e^{-\theta/T_k}$. So, the acceptance probability function is given as:

$$p = \begin{cases} e^{-\theta/T_k} & \text{if } \theta > 0 \\ 1 & \text{if } \theta \leq 0 \end{cases} \quad (15)$$

Here, $\theta = fit(neighbor) - fit(curr.sol^n)$ is the difference between the objective values of the generated neighbor and current solution. T_k denotes temperature

at k^{th} instance of accepting a new solution. Initially $T_k = 2 * |D|$ is considered, where, $|D|$ is dimension of the problem. For further iterations, $T_{k+1} = \alpha T_k$ where $\alpha \in [0, 1]$ is the cooling coefficient,

3.3. Proposed Discrete EO

Now, the main challenge in an FS problem is to search for the best possible feature subset. Especially, in wrapper-based algorithms, since each feature subset needs to be evaluated using a learner (classifier), searching for the best feature subset is highly time-consuming. Hence, a proper optimization technique is desired to reduce the required number of subset evaluations. The comparative results of EO with other methods present in the literature [Faramarzi et al. \(2020\)](#) have inspired us to propose this algorithm as a search technique in a wrapper-based FS method. EO is originally designed for continuous search space, but since FS deals with binary search space, we have designed a binary version of EO to deal with the FS problem. Binary search space implies [Mafarja & Mirjalili \(2017\)](#) that solutions are restricted to binary $\{0, 1\}$ values. In the binary version, each concentration is expressed as a vector of 0's and 1's, where the value of 1 represents the corresponding feature is selected and the value of 0 implies that the corresponding feature is not selected.

In the continuous version of EO, the concentration of the solution is updated as Equation (14). Here we need to use a transfer function [Mirjalili & Lewis \(2013\)](#) to make the binary version of EO. There are numerous transfer functions being used in the domain of FS to convert continuous optimization algorithms into binary approaches. In this work, a U-shaped transfer function is used whose applicability is proved in [Mirjalili et al. \(2020\)](#).

The U-shaped transfer function is provided in Equation (16).

$$T(x) = \alpha \cdot |x|^\beta \quad (16)$$

where α and β are the two controlling parameters. α defines the slope and β

controls the basin width of the transfer function. The utilized transfer function is depicted in Figure 1.

The concentration in the real domain will be converted to binary vector as per Equation (17) using the probability value generated by Equation (16).

$$X_t^d = \begin{cases} 1 & \text{if } rno < T(X_t^d) \\ 0 & \text{if } rno \geq T(X_t^d) \end{cases} \quad (17)$$

where X_t^d is d^{th} dimension of the concentration in t^{th} iteration, rno is a random number, $rno \in [0, 1]$.

Figure 1: Utilised U-shaped Transfer Functions

EO performs exploitation mainly using the Equilibrium pool [Faramarzi et al. \(2020\)](#). This pool controls both exploration and exploitation. During initial iterations, the distance among the equilibrium candidates is high, and the use of these candidates for updating concentrations helps perform a global search. With the increasing number of iterations, the equilibrium candidates come closer to each other and then using these candidates to update the concentrations helps perform local search encircling the candidates, which results in exploitation. Exploration is exclusively taken care of using Generation Probability (GP, Equation (13)). However, there is no such thing to consider exploitation particularly. Hence, we have incorporated the concept of SA to perform a local search, *i.e.*, take care of exploitation specifically. The hybrid version of EO is labeled as DEOSA (Discrete EO with SA).

FS is considered as a multi-objective optimization problem with two different criteria to evaluate the feature subset in consideration: classification accuracy and the number of selected features. To be specific, the objective of FS is to achieve maximum classification accuracy with the minimum number of features. These two criteria are contradictory in nature [Emary et al. \(2016b\)](#), so we have considered the classification error rate instead of accuracy. Equation (18) aggregates both of the objectives together and converts the multi-objective FS problem into a single objective problem.

$$\downarrow Objective = \eta \times \epsilon + \mu \times \frac{|v|}{|D|} \quad (18)$$

where, ϵ denotes the classification error rate, $|v|$ represents the number of features in the subset being evaluated, and $|D|$ represents the total number of features in the dataset. μ and η respectively represent the weight (importance) of the subset length and the classification error, and $\eta + \mu = 1$. We have used the K-Nearest Neighbor (KNN) classifier [Altman \(1992\)](#) to compute the classification error (ϵ).

4. Experimental Results

This section reports the experiments used to demonstrate the applicability of EO in the FS domain. We have used the KNN [Altman \(1992\)](#) classifier to compute the classification accuracy. According to the format followed in [Emary et al. \(2016b\)](#); [Mafarja & Mirjalili \(2017\)](#), we have set the value of k as 5. The datasets are divided in 80–20 train-test format. The proposed DEOSA is developed using Python3 programming language while the graphical illustrations are prepared using MATLAB.

4.1. Dataset Description

To evaluate the performance of DEO and DEOSA, 18 popular UCI datasets have been used from the repository [Blake \(1998\)](#). The datasets are adopted

from diversified backgrounds to ensure the effectiveness of the proposed model. A brief overview of these datasets is outlined in Table 1. Out of the 18 datasets, there are 15 bi-class and 3 multi-class datasets. The datasets vary in terms of both the number of attributes (features) and instances. These variances help in providing a proper evaluation of the proposed algorithm.

Table 1: Description of the datasets used for the FS purpose

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

4.2. Parameter Tuning

For any population-based evolutionary algorithm, the population size and the maximum number of allowable iterations significantly affect the searching procedure. Hence, it is very important to tune these parameters to retrieve optimal results from the algorithm. Population size maintains the extent of interaction of agents in a population and the maximum number of iterations

guides the convergence policy of the algorithm. In an attempt to find a proper combination of values for these parameters, DEO and DEOSA are simulated for different combinations of these parameters.

Figure 2: Parameter variation for 18 UCI datasets

Population size for DEO and DEOSA has been varied with the values of [10, 20, 30, 40, 50, 60] while the number of iterations has been varied as [20, 30, 40], Classification accuracies are plotted against the varying parameters for each dataset in Figure 2. It is evident from the graphs shown Figure 2, the combination of (50, 30) for population size and the maximum number of iterations works the best in the given scenario. Therefore, for the rest of the experimentation, we have fixed these values.

Another very important aspect of any evolutionary algorithm is the iteration-wise improvement in performance. Convergence graphs help to visualize the evolutionary strength of any algorithm. For this analysis, convergence graphs of DEO and DEOSA have been plotted in Figure 3. The graphs indicate that both DEO and DEOSA are able to provide steady and fast convergence over the iterations. As the best value for the maximum number of iterations is found to be 30, the convergence graphs are provided for 30 iterations.

4.3. Results and Discussion

This section reports the results obtained by DEO and DEOSA over the 18 UCI datasets outlined in Table 1. To get a generalized overview of the algorithms, both the algorithms have been run 10 times and the best and average classification accuracies obtained by these experiments have been provided in Table 2. Inspecting the results in this table, it is evident that DEOSA is able to achieve better accuracy in most of the cases as compared to DEO. However, both algorithms have become successful in providing a steady result which is evident from the low standard deviation values. In order to compare these results with the results obtained before FS, a simple comparison is provided in Table 3. For all the datasets, classification accuracy has improved after FS (with huge margins in multiple cases).

Both algorithms provide $\geq 90\%$ accuracy on 15 datasets but the accuracies

Figure 3: Convergence graphs for 18 UCI datasets

are higher for DEOSA. Moreover, DEO has got 100% accuracy on four datasets while DEOSA has scored full for five datasets which are quite remarkable. If we consider only the number of features used to obtain these impressive results, DEO and DEOSA still perform exceptionally well. DEO and DEOSA use only 50% or even a lesser number of features for classification in most of the cases. The contribution of SA is evident in Table 3 from the superiority of DEOSA in providing better results over DEO.

In the case of BreastCancer, Exactly, HeartEW, Lymphography, M-of-n, SpectEW, Wine, and Zoo datasets, both the variants of EO have shown a huge increase in classification accuracy after FS. For the rest of the datasets, we can see increments of about 1–10% which are quite significant improvements as well.

From this discussion, we can safely comment that DEO and DEOSA are able to successfully select the optimal set of features from the datasets. The accuracy provided by both the models using very few features makes both the algorithms very competitive in the field of FS.

Overall, we can say that although DEO and DEOSA perform really well in FS, the results obtained by DEOSA are better than DEO. Hence, it can be concluded that SA plays a significant role in improving the performance of DEO. Basically, in DEO, exploration in the search space is guided by the four particles present in the equilibrium pool. The fifth particle in the pool, which is the average of the other four particles, mainly helps in exploitation. Whereas, the stability between exploitation and exploration is maintained by the exponential term mentioned in Section 3.1.2. However, there may be situations when particles in the equilibrium pool become similar in nature which indicates that they belong to the same part of the search space. As a consequence, it leads to massive exploitation of that specific part. On the other hand, it limits the algorithm to explore the entire search space to look for the global optima. Here comes the role of the SA that actually assists DEOSA with exploring the search

space, and therefore superior performance in DEOSA. In this context, it is to be noted that even though SA increases the exploratory ability of DEO by circumventing local optima, the exploration-exploitation trade-off remains stable due to the usage of the exponential term.

Table 2: Representation of best and average classification accuracies obtained by DEO and DEOSA. The standard deviations of the accuracies are also provided in the table.

Dataset	Criteria	DEO	DEOSA
BreastCancer	Best Accuracy	96.42857143	96.42857143
	Avg. Accuracy	95.85714286	96.42857143
	Std. Deviation	0.947607083	0
BreastEW	Best Accuracy	95.61403509	96.49122807
	Avg. Accuracy	95.1754386	96.05263158
	Std. Deviation	0.438596491	0.438596491
CongressEW	Best Accuracy	95.40229885	96.55172414
	Avg. Accuracy	94.82758621	95.63218391
	Std. Deviation	0.574712644	0.860151123
Exactly	Best Accuracy	100	100
	Avg. Accuracy	93.3	99.35
	Std. Deviation	8.1	0.807774721
Exactly2	Best Accuracy	76	76
	Avg. Accuracy	76	76
	Std. Deviation	0	0
HeartEW	Best Accuracy	90.74074074	90.74074074
	Avg. Accuracy	87.77777778	90.18518519
	Std. Deviation	2.644973492	1.185763748
Ionosphere	Best Accuracy	97.14285714	95.71428571
	Avg. Accuracy	94.28571429	94.85714286
	Std. Deviation	1.428571429	0.699854212
KrVsKpEW	Best Accuracy	95.93114241	96.5571205

	Avg. Accuracy	94.8513302	95.77464789
	Std. Deviation	0.796278825	0.533001138
Lymphography	Best Accuracy	93.33333333	96.66666667
	Avg. Accuracy	90.66666667	93.33333333
	Std. Deviation	1.333333333	1.490711985
M-of-n	Best Accuracy	100	100
	Avg. Accuracy	98.95	99.7
	Std. Deviation	1.171537451	0.9
PenglungEW	Best Accuracy	93.33333333	100
	Avg. Accuracy	93.33333333	94
	Std. Deviation	1.42E-14	2
Sonar	Best Accuracy	97.61904762	97.61904762
	Avg. Accuracy	94.52380952	95.95238095
	Std. Deviation	1.859583256	1.52455339
SpectEW	Best Accuracy	92.59259259	90.74074074
	Avg. Accuracy	90.92592593	90.74074074
	Std. Deviation	0.997252742	0
Tic-tac-toe	Best Accuracy	83.33333333	83.33333333
	Avg. Accuracy	83.28125	83.33333333
	Std. Deviation	0.15625	0
Vote	Best Accuracy	96.66666667	96.66666667
	Avg. Accuracy	96	96.66666667
	Std. Deviation	0.816496581	1.42E-14
WaveformEW	Best Accuracy	84.4	84.9
	Avg. Accuracy	83.14	83.78
	Std. Deviation	0.917823512	0.746726188
Wine	Best Accuracy	100	100
	Avg. Accuracy	99.72222222	100
	Std. Deviation	0.833333333	0
Zoo	Best Accuracy	100	100

Avg. Accuracy	100	100
Std. Deviation	0	0

Table 3: Performance of DEO and DEOSA in terms of classification accuracy and no. of selected features over UCI datasets

Sl. No.	Dataset	Total		DEO		DEOSA	
		Accuracy	#Features	Accuracy	#Features	Accuracy	#Features
1	BreastCancer	60.71428571	10	96.42857143	3	96.42857143	3
2	BreastEW	91.22807018	30	95.61403509	6	96.49122807	8
3	CongressEW	91.95402299	16	95.40229885	3	96.55172414	5
4	Exactly	76.5	13	100	6	100	7
5	Exactly2	69	13	76	1	76	1
6	HeartEW	74.07407407	13	90.74074074	3	90.74074074	5
7	Ionosphere	88.57142857	34	97.14285714	5	95.71428571	7
8	KrVsKpEW	95.08763693	36	95.93114241	13	96.5571205	15
9	Lymphography	73.33333333	18	93.33333333	8	96.66666667	9
10	M-of-n	88.5	13	100	7	100	6
11	PenglungEW	93.33333333	325	93.33333333	98	100	86
12	Sonar	88.0952381	60	97.61904762	26	97.61904762	25
13	SpectEW	75.92592593	22	92.59259259	12	90.74074074	9
14	Tic-tac-toe	82.89583333	9	83.33333333	5	83.33333333	5
15	Vote	90	16	96.66666667	1	96.66666667	1
16	WaveformEW	79.7	40	84.4	19	84.9	27
17	Wine	63.88888889	13	100	3	100	3
18	Zoo	85	16	100	5	100	4

4.4. Comparison

In this sub-section, we present the results of the proposed DEOSA and highlight how it performs *w.r.t.* state-of-the-art methods. We have compared the results of DEOSA with some classical (GA, PSO, GSA), some modern (BBA [Mirjalili et al. \(2013\)](#), bALO-V, bALO-S), and some hybrid (HGSA, CPBGSA [Guha et al. \(2020a\)](#), DGA [Guha et al. \(2019\)](#), WFACOFs [Ghosh et al. \(2019b\)](#),

HSGW, RSGW, ASGW [Mafarja et al. \(2019\)](#)) algorithms. This wide range of algorithms provides a proper comparison environment which, in turn, helps to establish the superiority of DEOSA in FS.

Figure 4: Performance comparison of the proposed FS method with other meta-heuristic based FS methods in terms of classification accuracy over 18 UCI datasets

Figure 5: Average classification accuracy achieved by each method over UCI datasets. The figure clearly demonstrates that this binary version of EO is able to provide competitive results on UCI datasets.

Figure 4 shows the performance of DEOSA in terms of classification accuracy over UCI datasets. Inspecting Figure 4, it can be observed that DEOSA performs best in 7 cases. This algorithm is the second-best over HeartEW and Sonar datasets while HSGW and RSGW perform the best over them respectively. For SpectEW and Tic-tac-toe, it secures the third rank. In Figure 5, the average classification accuracy achieved by each method over all the 18 UCI datasets is reported. In terms of the average accuracy, DEOSA provides 94.35612% classification accuracy, which is the highest among all the algorithms. HSGW and ASGW are the second and third rank holders in terms of average classification accuracy.

Figure 6: Comparison of no. of selected features for 18 UCI datasets

Figure 7: The average number of features selected by each method. The numbers clearly demonstrate that DEOSA uses significantly low dimensional feature vectors to provide the competitive accuracy reported in Figure 5

Figure 6 shows the performance of DEOSA in terms of another important aspect of FS and the very purpose of this research field, which is the number of selected features. We can observe that the DEOSA has selected the lowest number of features in 8 cases. On average over all 18 datasets (Figure 7), DEOSA selects 13.90555556 features, which is second best, following HGSA (10.8).

In order to determine the significance of the obtained results, we have performed the Wilcoxon rank-sum test [Wilcoxon \(1992\)](#) with 5% significance level for each pair of methods used in this section. Table 4 shows the obtained *p-values*, $p \leq 0.05$ is shown in bold.

Table 4: p-values of the Wilcoxon rank-sum test obtained for UCI datasets

	DEOSA	GA	PSO	GSA	BBA	bALO-S	bALO-V	HGSA	CPBGSA	DGA	WFACOFs	HSGW	RSGW	ASGW
DEOSA	-	0.000	0.003	0.001	0.001	0.000	0.000	0.000	0.002	0.023	0.012	0.020	0.029	0.022
GA	0.000	-	0.000	0.000	0.000	0.314	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000
PSO	0.003	0.000	-	0.012	0.305	0.001	0.433	0.070	0.102	0.052	0.102	0.039	0.03	0.032
GSA	0.001	0.000	0.012	-	0.001	0.000	0.070	0.006	0.005	0.005	0.008	0.004	0.014	0.008
BBA	0.001	0.000	0.305	0.001	-	0.000	0.355	0.040	0.015	0.040	0.053	0.013	0.013	0.01
bALO-S	0.000	0.314	0.001	0.000	0.000	-	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002
bALO-V	0.000	0.000	0.433	0.070	0.355	0.000	-	0.295	0.214	0.016	0.067	0.035	0.03	0.039
HGSA	0.000	0.000	0.070	0.006	0.040	0.000	0.295	-	0.177	0.016	0.139	0.076	0.07	0.072
CPBGSA	0.002	0.000	0.102	0.005	0.015	0.000	0.214	0.177	-	0.088	0.170	0.191	0.19	0.19
DGA	0.023	0.000	0.052	0.005	0.040	0.000	0.016	0.016	0.088	-	0.587	0.959	0.91	0.9
WFACOFs	0.012	0.000	0.102	0.008	0.053	0.000	0.067	0.139	0.170	0.587	-	0.865	0.85	0.82
HSGW	0.020	0.000	0.039	0.004	0.013	0.000	0.035	0.076	0.191	0.959	0.865	-	0.61	0.071
RSGW	0.029	0.001	0.03	0.014	0.013	0.000	0.03	0.07	0.19	0.91	0.85	0.61	-	0.053
ASGW	0.022	0.000	0.032	0.008	0.01	0.002	0.039	0.072	0.19	0.9	0.82	0.071	0.053	-

5. Additional Testing

As evident from the previous discussion, DEOSA is an effective algorithm for FS. In this section, additional experiments are performed to check how DEOSA can scale to bigger datasets and other challenging binary optimization problems.

5.1. Application on Microarray Data

Microarray datasets [Guha et al. \(2020b\)](#) are high-dimensional in nature and pose greater problem against FS models due to the requirement of extreme searching. Therefore, such datasets are very effective for assessing the robustness of any FS model. For this experimentation, 7 publicly available Microarray datasets have been considered, and details of which are presented in Table 5.

Table 5: Description of the Microarray datasets used in the present work

Sl. No.	Dataset	No. of instances	No. of features	No. of classes
1	SRBCT	83	2308	4
2	Leukemia	72	5147	2
3	DLBCL	77	7070	2
4	Colon	36	7464	2
5	MLL	102	12533	2
6	Prostrate	72	12533	3
7	AMLGSE2191	54	12616	2

The classification accuracies obtained over the seven Microarray datasets have further been compared with some classical as well as recently proposed meta-heuristics that include GA, Memetic Algorithm, WFACOFs, and Embedded Chaotic Whale Survival Algorithm (ECWSA). The results of these algorithms are reported in [Guha et al. \(2020b\)](#). Table 6 contains the classification accuracy obtained by these algorithms over the 7 Microarray datasets. The number of features used to achieve the accuracy has been provided in parenthesis corresponding to the accuracy.

Table 6: Performance comparison of the proposed method with some existing methods on Microarray datasets

Dataset	Accuracy (in %)							
	GA	MA	WFACOFs	ECWSA-1	ECWSA-2	ECWSA-3	ECWSA-4	DEOSA
AMLGSE2191	100(98)	100(91)	96.3(17)	96.67(17)	100(9)	95.83(16)	95.83(18)	100(35)
Colon	100(81)	100(81)	100(3)	100(36)	100(41)	100(30)	100(43)	100(21)
DLBCL	100(88)	100(105)	100(3)	100(29)	100(24)	100(26)	100(31)	100(9)
Leukemia	100(85)	100(65)	100(5)	97.22(7)	100(8)	100(4)	97.22(5)	100(17)
MLL	100(99)	100(107)	100(22)	96.30(16)	98.15(16)	96.30(9)	96.30(19)	100(24)
Prostrate	100(94)	100(80)	100(25)	100(16)	100(17)	100(8)	100(15)	95.24(12)
SRBCT	100(78)	100(50)	100(19)	100(45)	100(32)	100(34)	100(30)	100(17)

For 6 out of 7 Microarray datasets, DEOSA has obtained 100% classifica-

tion accuracy which is quite remarkable. One thing which is really intriguing is the number of features selected by DEOSA. These numbers are extremely low compared to the total number of features mentioned in Table 5. The count of the features is still less than most of the algorithms used here for the comparison.

5.2. Application on 0/1 Knapsack Problem

In order to check the applicability of DEOSA in other binary optimization problems, it has been further applied over 0/1 knapsack problems, which are considered as a popular combinatorial optimization problem with various applications. The problem states that there are some items where each item contains some value and some weight. The objective is to select a set of items that will maximize the accumulated value subject to a constraint on the total weight. The mathematical formulation of 0/1 knapsack can be stated as:

$$\text{maximize } Z = \sum_{i=1}^n v_i x_i \quad (19)$$

$$\text{subject to : } \sum_{i=1}^n w_i x_i \leq W \quad (20)$$

where x_i is the state of i^{th} item which is 1 if the item is selected, else 0, v_i and w_i are the value and weight of the i^{th} item respectively and W is the maximum allowable weight.

For the experimentation with the binary knapsack problem, 25 popular knapsack datasets have been selected [Kreher](#) from. The results of 30 independent runs have been tabulated in Table 7. The mean and standard deviation of these runs are provided in the table. The results of some other U-shape function-based models have been taken from [Mirjalili et al. \(2020\)](#) for comparison. To maintain a neutral computation environment, the population size and the number of iterations are set to 20 and 500 respectively as mentioned in the paper.

Table 7: Comparison of obtained results for binary Knapsack problems.

Name	Nvar	UBPSO1		UBPSO2		UBPSO3		UBPSO4		DEOSA	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Ks_8a	8	3920000	5160	3920000	0	3924400	0	3910000	18100	3924400	0
Ks_8b	8	3813700	0	3813700	0	3813700	0	3813700	0	3740776	0
Ks_8c	8	3347500	0	3347500	0	3347500	0	3350000	3530	3347452	0
Ks_8d	8	4187700	0	4187700	0	4180000	17500	4187700	0	4082475	0
Ks_8e	8	4955600	0	4955600	0	4930000	47500	4940000	43100	4857227	0
Ks_12a	12	5690000	2730	5680000	9210	5680000	27700	5680000	27500	5688887	0
Ks_12b	12	6480000	14300	6480000	13200	6480000	14800	6490000	13200	6498597	0
Ks_12c	12	5170000	11800	5170000	11800	5160000	15700	5160000	18900	5169676	0
Ks_12d	12	6990000	9950	6980000	31400	6990000	9870	6990000	9950	6992404	0
Ks_12e	12	5337500	0	5337500	0	5330000	20200	5330000	15100	5227215	0
Ks_16a	16	7840000	11200	7840000	13400	7830000	16600	7830000	22400	7842384.5	11703.51
Ks_16b	16	9320000	44300	9330000	39100	9310000	46900	9320000	49900	9341431.3	19418.63
Ks_16c	16	9120000	24900	9130000	24200	9120000	34000	9120000	34600	9141289.3	14477.44
Ks_16d	16	9320000	23300	9330000	21900	9330000	19700	9330000	30300	9326519.033	22462.72
Ks_16e	16	7760000	8230	7750000	33800	7740000	42400	7760000	8470	7760444.8	8221.736
Ks_20a	20	10700000	32100	10700000	39300	10700000	34700	10700000	23500	10659381	31178.24
Ks_20b	20	9790000	33200	9770000	35200	9760000	35500	9770000	45200	9740663	33023.79
Ks_20c	20	10692000	38823	10700000	44100	10700000	27500	10700000	50900	10624542.63	44763.87
Ks_20d	20	13900000	33000	8910000	25000	8880000	38800	8870000	52100	8863593.933	31942.57
Ks_20e	20	9350000	13300	9340000	24700	13912000	32994	9350000	10600	9309090.733	18480.36
Ks_24a	24	13493000	36247	13500000	43500	13500000	37100	13500000	60000	13398986.3	50160.74
Ks_24b	24	12202000	23838	12200000	44000	12200000	34000	12200000	62500	12075179.07	34645.94
Ks_24c	24	12202000	23838	12200000	44000	12200000	34000	12200000	62500	12345731	31244.35
Ks_24d	24	12416000	22522	12400000	27100	12400000	33800	12400000	43200	11715570.53	32635.66
Ks_24e	24	13865000	21572	13900000	28100	13900000	30600	13900000	44000	13829445.6	52481.85

From Table 7, it is clearly visible that DEOSA is able to provide results comparable with the other U-function-based models. The knapsack is a maximization problem, so the higher the mean, the better the result. DEOSA has obtained very high mean values over all 25 datasets. One important thing to observe here is that the standard deviation for the proposed method is signifi-

cantly low which indicates the stability of the algorithm in providing such good solutions.

After analyzing the results obtained by DEOSA over UCI, Microarray, and Knapsack datasets, the advantages of DEOSA can be clearly stated as:

- DEOSA is able to provide very high classification accuracy for all the 18 UCI datasets while utilizing a very low percentage of features from the entire feature set. It proves the applicability of the proposed model in the domain of FS.
- Successful application over Microarray datasets proves the scalability of DEOSA because Microarray datasets are high-dimensional in nature and contain > 2000 features. In 6 out of 7 datasets, DEOSA has achieved perfect accuracy of 100% which is very impressive, On the other hand, the number of features to obtain these accuracies is lesser than most of the algorithms used for comparison in Table 6.
- 0/1 Knapsack problems are used to prove the robustness and applicability of DEOSA in other binary optimization problems. The high values obtained for the 25 binary Knapsack problems demonstrate that DEOSA is suitable for other binary optimization problems also. Further, low standard deviations for all these datasets indicate that DEOSA is very stable while achieving these results and it has very high chances of providing good results for every run of the algorithm.

6. Conclusion

In this work, we have proposed a binary variant of EO (DEO) to make it applicable to the field of FS. To map the continuous values of EO to binary search space, a U-shaped transfer function has been used. The exploitation ability of

DEO has been further enhanced through the application of SA which forms a hybridized variant of DEO, called DEOSA. Both the proposed models have been tested on 18 popular UCI datasets. From the results and comparison with other contemporary FS algorithms, we can conclude that DEO and DEOSA have performed significantly well. We have further carried out a test for the statistical significance of the obtained results using the Wilcoxon rank-sum test. The test results also indicate that the performance of DEO and DEOSA is satisfactory. To prove the scalability and robustness of DEOSA over FS or any binary optimization problem, it has been additionally applied over 7 high-dimensional Microarray datasets and 25 binary Knapsack problems. The results for these sets of experiments are also very convincing. Thus, through this rigorous experimentation, it has been proved that the binary variant of EO works pretty well in the FS domain and the addition of SA leads to further improvement. For future works, we want to try out other transfer functions for the mapping and apply the proposed models in other domains like FS for Facial Emotion Recognition, Human Activity Recognition, etc. We are also interested to test other hybridized variants of DEO by using different meta-heuristic approaches apart from SA.

References

- Ackley, D. H. (1987). *A Connectionist Machine for Genetic Hillclimbing*. Springer US. doi:[10.1007/978-1-4613-1997-9](https://doi.org/10.1007/978-1-4613-1997-9).
- Adam, S. P., Alexandropoulos, S.-A. N., Pardalos, P. M., & Vrahatis, M. N. (2019). No free lunch theorem: A review. In *Approximation and Optimization* (pp. 57–82). Springer International Publishing. URL: https://doi.org/10.1007/978-3-030-12767-1_5. doi:[10.1007/978-3-030-12767-1_5](https://doi.org/10.1007/978-3-030-12767-1_5).
- Ahmed, K., Hassanien, A. E., & Bhattacharyya, S. (2017). A novel chaotic chicken swarm optimization algorithm for feature selection. In *2017 Third International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)* (pp. 259–264). IEEE. URL: <https://doi.org/10.1109/ICRCICN.2017.8288888>.

[//doi.org/10.1109/icrcicn.2017.8234517](https://doi.org/10.1109/icrcicn.2017.8234517). doi:10.1109/icrcicn.2017.8234517.

Ahmed, S., Ghosh, K. K., Garcia-Hernandez, L., Abraham, A., & Sarkar, R. (2020). Improved coral reefs optimization with adaptive β -hill climbing for feature selection. *Neural Computing and Applications*, . doi:10.1007/s00521-020-05409-1.

Ahmed, S., Ghosh, K. K., Mirjalili, S., & Sarkar, R. (2021). Aieou: Automata-based improved equilibrium optimizer with u-shaped transfer function for feature selection. *Knowledge-Based Systems*, 228, 107283.

Allan, K. (1977). Classifiers. *Language*, 53, 285–311. doi:10.1353/lan.1977.0043.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46, 175. doi:10.2307/2685209.

Arora, S., & Anand, P. (2019). Binary butterfly optimization approaches for feature selection. *Expert Systems with Applications*, 116, 147–160. URL: <https://doi.org/10.1016/j.eswa.2018.08.051>. doi:10.1016/j.eswa.2018.08.051.

Blake, C. (1998). Cj merz uci repository of machine learning databases. *University of California at Irvine*, .

Cheraghalipour, A., Hajiaghaei-Keshteli, M., & Paydar, M. M. (2018). Tree growth algorithm (TGA): A novel approach for solving optimization problems. *Engineering Applications of Artificial Intelligence*, 72, 393–414. URL: <https://doi.org/10.1016/j.engappai.2018.04.021>. doi:10.1016/j.engappai.2018.04.021.

Chizi, B., & Maimon, O. (2005). Dimension reduction and feature selection. In *Data Mining and Knowledge Discovery Handbook* (pp. 93–111). Springer-Verlag. doi:10.1007/0-387-25465-x_5.

- Cuevas, E., González, A., Zaldívar, D., & Cisneros, M. P. (2015). An optimisation algorithm based on the behaviour of locust swarms. *International Journal of Bio-Inspired Computation*, 7, 402. URL: <https://doi.org/10.1504/ijbic.2015.073178>. doi:10.1504/ijbic.2015.073178.
- Dhiman, G., & Kumar, V. (2018). Spotted hyena optimizer for solving complex and non-linear constrained engineering problems. In *Harmony Search and Nature Inspired Optimization Algorithms* (pp. 857–867). Springer Singapore. URL: https://doi.org/10.1007/978-981-13-0761-4_81. doi:10.1007/978-981-13-0761-4_81.
- Dorigo, M., & Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)* (pp. 1470–1477 Vol. 2). volume 2. doi:10.1109/CEC.1999.782657.
- Emary, E., Zawbaa, H. M., & Hassanien, A. E. (2016a). Binary ant lion approaches for feature selection. *Neurocomputing*, 213, 54–65. URL: <https://doi.org/10.1016/j.neucom.2016.03.101>. doi:10.1016/j.neucom.2016.03.101.
- Emary, E., Zawbaa, H. M., & Hassanien, A. E. (2016b). Binary grey wolf optimization approaches for feature selection. *Neurocomputing*, 172, 371–381. URL: <https://doi.org/10.1016/j.neucom.2015.06.083>. doi:10.1016/j.neucom.2015.06.083.
- Faisal, M., Mathkour, H., & Alsulaiman, M. (2016). AntStar: Enhancing optimization problems by integrating an ant system and algorithm. *Scientific Programming*, 2016, 1–12. URL: <https://doi.org/10.1155/2016/5136327>. doi:10.1155/2016/5136327.
- Faramarzi, A., Heidarinejad, M., Stephens, B., & Mirjalili, S. (2020). Equilibrium optimizer: A novel optimization algorithm. *Knowledge-Based Systems*, 191, 105190. doi:10.1016/j.knosys.2019.105190.

- Gao, Z.-M., Zhao, J., & Li, S.-R. (2020). The binary equilibrium optimization algorithm with sigmoid transfer functions. In *Proceedings of the 2020 12th International Conference on Machine Learning and Computing* (pp. 193–197).
- Ghosh, K. K., Ahmed, S., Singh, P. K., Geem, Z. W., & Sarkar, R. (2020). Improved binary sailfish optimizer based on adaptive β -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. (2020). Binary social mimic optimization algorithm with x-shaped transfer function for feature selection. *IEEE Access*, 8, 97890–97906. URL: [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. (2018a). 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).
- Ghosh, M., Guha, R., Mondal, R., Singh, P. K., Sarkar, R., & Nasipuri, M. (2018b). Feature selection using histogram-based multi-objective GA for handwritten devanagari numeral recognition. In *Advances in Intelligent Systems and Computing* (pp. 471–479). Springer Singapore. doi:[10.1007/978-981-10-7566-7_46](https://doi.org/10.1007/978-981-10-7566-7_46).
- Ghosh, M., Guha, R., Sarkar, R., & Abraham, A. (2019a). A wrapper-filter feature selection technique based on ant colony optimization. *Neural Computing and Applications*, (pp. 1–19).
- Ghosh, M., Guha, R., Sarkar, R., & Abraham, A. (2019b). A wrapper-filter feature selection technique based on ant colony optimization. *Neural Computing and Applications*, 32, 7839–7857. URL: <https://doi.org/10.1007/s00521-019-04171-3>. doi:[10.1007/s00521-019-04171-3](https://doi.org/10.1007/s00521-019-04171-3).

- Gómez, D., & Rojas, A. (2016). An empirical overview of the no free lunch theorem and its effect on real-world machine learning classification. *Neural Computation*, 28, 216–228. URL: https://doi.org/10.1162/neco_a_00793. doi:10.1162/neco_a_00793.
- Guha, R., Ghosh, M., Chakrabarti, A., Sarkar, R., & Mirjalili, S. (2020a). Introducing clustering based population in binary gravitational search algorithm for feature selection. *Applied Soft Computing*, 93, 106341. URL: <https://doi.org/10.1016/j.asoc.2020.106341>. doi:10.1016/j.asoc.2020.106341.
- Guha, R., Ghosh, M., Kapri, S., Shaw, S., Mutsuddi, S., Bhateja, V., & Sarkar, R. (2019). Deluge based genetic algorithm for feature selection. *Evolutionary Intelligence*, . URL: <https://doi.org/10.1007/s12065-019-00218-5>. doi:10.1007/s12065-019-00218-5.
- Guha, R., Ghosh, M., Mutsuddi, S., Sarkar, R., & Mirjalili, S. (2020b). Embedded chaotic whale survival algorithm for filter-wrapper feature selection. *Soft Computing*, . doi:10.1007/s00500-020-05183-1.
- Gupta, D., Agrawal, U., Arora, J., & Khanna, A. (2020). Bat-inspired algorithm for feature selection and white blood cell classification. In *Nature-Inspired Computation and Swarm Intelligence* (pp. 179–197). Elsevier. URL: <https://doi.org/10.1016/b978-0-12-819714-1.00022-1>. doi:10.1016/b978-0-12-819714-1.00022-1.
- Guyon, I., Gunn, S., Ben-Hur, A., & Dror, G. (2005). Result analysis of the nips 2003 feature selection challenge. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in Neural Information Processing Systems 17* (pp. 545–552). MIT Press.
- Harifi, S., Khalilian, M., Mohammadzadeh, J., & Ebrahimnejad, S. (2019). Emperor penguins colony: a new metaheuristic algorithm for optimization. *Evolutionary Intelligence*, 12, 211–226. URL: <https://doi.org/10.1007/s12065-019-00212-x>. doi:10.1007/s12065-019-00212-x.

- Huang, J., Cai, Y., & Xu, X. (2007). A hybrid genetic algorithm for feature selection wrapper based on mutual information. *Pattern Recognition Letters*, 28, 1825–1844. doi:[10.1016/j.patrec.2007.05.011](https://doi.org/10.1016/j.patrec.2007.05.011).
- Hussien, A. G., Hassanien, A. E., Houssein, E. H., Bhattacharyya, S., & Amin, M. (2018). S-shaped binary whale optimization algorithm for feature selection. In *Recent Trends in Signal and Image Processing* (pp. 79–87). Springer Singapore. URL: https://doi.org/10.1007/978-981-10-8863-6_9. doi:[10.1007/978-981-10-8863-6_9](https://doi.org/10.1007/978-981-10-8863-6_9).
- Ibrahim, H. T., Mazher, W. J., Ucan, O. N., & Bayat, O. (2017). Feature selection using salp swarm algorithm for real biomedical datasets. *IJCSNS*, 17, 13–20.
- Jiang, S., Chin, K.-S., Wang, L., Qu, G., & Tsui, K. L. (2017). Modified genetic algorithm-based feature selection combined with pre-trained deep neural network for demand forecasting in outpatient department. *Expert Systems with Applications*, 82, 216–230. doi:[10.1016/j.eswa.2017.04.017](https://doi.org/10.1016/j.eswa.2017.04.017).
- Joyce, T., & Herrmann, J. M. (2017). A review of no free lunch theorems, and their implications for metaheuristic optimisation. In *Nature-Inspired Algorithms and Applied Optimization* (pp. 27–51). Springer International Publishing. URL: https://doi.org/10.1007/978-3-319-67669-2_2. doi:[10.1007/978-3-319-67669-2_2](https://doi.org/10.1007/978-3-319-67669-2_2).
- Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks* (pp. 1942–1948 vol.4). volume 4. doi:[10.1109/ICNN.1995.488968](https://doi.org/10.1109/ICNN.1995.488968).
- 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).
- Kreher, D. L. (). Knapsack problems. "<https://pages.mtu.edu/~kreher/cages/Data.html>". [Online; accessed 26-July-2020].

- van Laarhoven, P. J. M., & Aarts, E. H. L. (1987). Simulated annealing. In *Simulated Annealing: Theory and Applications* (pp. 7–15). Springer Netherlands. doi:[10.1007/978-94-015-7744-1_2](https://doi.org/10.1007/978-94-015-7744-1_2).
- Lamy, J.-B. (2018). Artificial feeding birds (AFB): A new metaheuristic inspired by the behavior of pigeons. In *Advances in Nature-Inspired Computing and Applications* (pp. 43–60). Springer International Publishing. URL: https://doi.org/10.1007/978-3-319-96451-5_3. doi:[10.1007/978-3-319-96451-5_3](https://doi.org/10.1007/978-3-319-96451-5_3).
- Leardi, R. (1996). Genetic algorithms in feature selection. In *Genetic Algorithms in Molecular Modeling* (pp. 67–86). Elsevier. doi:[10.1016/b978-012213810-2/50004-9](https://doi.org/10.1016/b978-012213810-2/50004-9).
- Ma, B., & Xia, Y. (2017). A tribe competition-based genetic algorithm for feature selection in pattern classification. *Applied Soft Computing*, 58, 328–338. doi:[10.1016/j.asoc.2017.04.042](https://doi.org/10.1016/j.asoc.2017.04.042).
- 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. URL: <https://doi.org/10.1007/s12559-019-09668-6>. 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).
- Marinakis, Y., Marinaki, M., & Matsatsinis, N. (2009). A hybrid discrete artificial bee colony - GRASP algorithm for clustering. In *2009 International Conference on Computers & Industrial Engineering* (pp. 548–553). IEEE. URL: <https://doi.org/10.1109/iccie.2009.5223810>. doi:[10.1109/iccie.2009.5223810](https://doi.org/10.1109/iccie.2009.5223810).
- Menesy, A. S., Sultan, H. M., & Kamel, S. (2020). Extracting model parameters of proton exchange membrane fuel cell using equilibrium optimizer algorithm.

- In *2020 International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE)* (pp. 1–7). IEEE. doi:[10.1109/reepe49198.2020.9059219](https://doi.org/10.1109/reepe49198.2020.9059219).
- 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., 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., & Yang, X.-S. (2013). Binary bat algorithm. *Neural Computing and Applications*, 25, 663–681. URL: <https://doi.org/10.1007/s00521-013-1525-5>. doi:[10.1007/s00521-013-1525-5](https://doi.org/10.1007/s00521-013-1525-5).
- 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* (pp. 241–259). Springer Singapore. doi:[10.1007/978-981-15-3290-0_19](https://doi.org/10.1007/978-981-15-3290-0_19).
- Moradi, P., & Gholampour, M. (2016). A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. *Applied Soft Computing*, 43, 117–130. doi:[10.1016/j.asoc.2016.01.044](https://doi.org/10.1016/j.asoc.2016.01.044).
- Naghdiani, M., & Jahanshahi, M. (2017). GSO: A new solution for solving unconstrained optimization tasks using garter snake’s behavior. In *2017 International Conference on Computational Science and Computational Intelli-*

- gence (CSCI)* (pp. 328–333). IEEE. URL: <https://doi.org/10.1109/csci.2017.55>. doi:10.1109/csci.2017.55.
- Nazaroff, W. W., & Alvarez-Cohen, L. (2001). *Environmental engineering science*. New York: Wiley.
- Oh, I.-S., Lee, J.-S., & Moon, B.-R. (2004). Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26, 1424–1437. doi:10.1109/tpami.2004.105.
- Pan, W.-T. (2012). A new fruit fly optimization algorithm: Taking the financial distress model as an example. *Knowledge-Based Systems*, 26, 69–74. URL: <https://doi.org/10.1016/j.knosys.2011.07.001>. doi:10.1016/j.knosys.2011.07.001.
- Petrov, C. (2020). Big data statistics. URL: <https://techjury.net/stats-about/big-data-statistics/#gref>.
- 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.
- Sayed, G. I., Khoriba, G., & Haggag, M. H. (2018). A novel chaotic salp swarm algorithm for global optimization and feature selection. *Applied Intelligence*, 48, 3462–3481. URL: <https://doi.org/10.1007/s10489-018-1158-6>. doi:10.1007/s10489-018-1158-6.
- Shadravan, S., Naji, H., & Bardsiri, V. (2019). The sailfish optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. *Engineering Applications of Artificial Intelligence*, 80, 20–34. URL: <https://doi.org/10.1016/j.engappai.2019.01.001>. doi:10.1016/j.engappai.2019.01.001.
- Shang, L., Zhou, Z., & Liu, X. (2016). Particle swarm optimization-based feature selection in sentiment classification. *Soft Computing*, 20, 3821–3834. doi:10.1007/s00500-016-2093-2.

- Shi, Y. (2011). Brain storm optimization algorithm. In *Lecture Notes in Computer Science* (pp. 303–309). Springer Berlin Heidelberg. URL: https://doi.org/10.1007/978-3-642-21515-5_36. doi:10.1007/978-3-642-21515-5_36.
- SIEDLECKI, W., & SKLANSKY, J. (1993). A NOTE ON GENETIC ALGORITHMS FOR LARGE-SCALE FEATURE SELECTION. In *Handbook of Pattern Recognition and Computer Vision* (pp. 88–107). WORLD SCIENTIFIC. doi:10.1142/9789814343138_0005.
- Taradeh, M., & Mafarja, M. (2019). Binary thermal exchange optimization for feature selection. In *Studies in Big Data* (pp. 239–260). Springer International Publishing. URL: https://doi.org/10.1007/978-3-030-32587-9_14. doi:10.1007/978-3-030-32587-9_14.
- Tran, B., Xue, B., & Zhang, M. (2019). Variable-length particle swarm optimization for feature selection on high-dimensional classification. *IEEE Transactions on Evolutionary Computation*, 23, 473–487. doi:10.1109/tevc.2018.2869405.
- Wilcoxon, F. (1992). Individual comparisons by ranking methods. In *Breakthroughs in statistics* (pp. 196–202). Springer New York. doi:10.1007/978-1-4612-4380-9_16.
- Wolpert, D. H., Macready, W. G. et al. (1995). *No free lunch theorems for search*. Technical Report Technical Report SFI-TR-95-02-010, Santa Fe Institute.
- Yan, C., Ma, J., Luo, H., & Patel, A. (2019). Hybrid binary coral reefs optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical datasets. *Chemometrics and Intelligent Laboratory Systems*, 184, 102–111. URL: <https://doi.org/10.1016/j.chemolab.2018.11.010>. doi:10.1016/j.chemolab.2018.11.010.

- Yang, J., & Honavar, V. (1998). Feature subset selection using a genetic algorithm. In *Feature Extraction, Construction and Selection* (pp. 117–136). Springer US. doi:[10.1007/978-1-4615-5725-8_8](https://doi.org/10.1007/978-1-4615-5725-8_8).
- Yu, J. J., & Li, V. O. (2015). A social spider algorithm for global optimization. *Applied Soft Computing*, *30*, 614–627. URL: <https://doi.org/10.1016/j.asoc.2015.02.014>. doi:[10.1016/j.asoc.2015.02.014](https://doi.org/10.1016/j.asoc.2015.02.014).
- Zhang, J., Xiao, M., Gao, L., & Pan, Q. (2018a). Queuing search algorithm: A novel metaheuristic algorithm for solving engineering optimization problems. *Applied Mathematical Modelling*, *63*, 464–490. doi:[10.1016/j.apm.2018.06.036](https://doi.org/10.1016/j.apm.2018.06.036).
- Zhang, J., Zhou, Y., & Luo, Q. (2018b). Nature-inspired approach: a wind-driven water wave optimization algorithm. *Applied Intelligence*, *49*, 233–252. URL: <https://doi.org/10.1007/s10489-018-1265-4>. doi:[10.1007/s10489-018-1265-4](https://doi.org/10.1007/s10489-018-1265-4).
- qiu Zhang, W., Zhang, Y., & Peng, C. (2019). Brain storm optimization for feature selection using new individual clustering and updating mechanism. *Applied Intelligence*, *49*, 4294–4302. URL: <https://doi.org/10.1007/s10489-019-01513-5>. doi:[10.1007/s10489-019-01513-5](https://doi.org/10.1007/s10489-019-01513-5).
- Zhao, J., & Gao, Z.-M. (2020). Simulation research on the binary equilibrium optimization algorithm. In *Proceedings of the 2020 12th International Conference on Machine Learning and Computing* (pp. 140–144).
- Zolghadr-Asli, B., Bozorg-Haddad, O., & Chu, X. (2017). Crow search algorithm (CSA). In *Advanced Optimization by Nature-Inspired Algorithms* (pp. 143–149). Springer Singapore. doi:[10.1007/978-981-10-5221-7_14](https://doi.org/10.1007/978-981-10-5221-7_14).