BEO: Binary Equilibrium Optimizer Combined with Simulated Annealing for Feature Selection

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

\textsuperscript{a}Computer Science and Engineering, Jadavpur University, Kolkata, India - 700032
\textsuperscript{b}Institute for Integrated and Intelligent Systems, Griffith University, Nathan, QLD 4111, Australia

Abstract

This work proposed a binary variant of the recently-proposed Equilibrium Optimizer (EO) to solve binary problems. A v-shaped transfer function is used to map continuous values created in EO to binary. To improve the exploitation of the Binary Equilibrium Optimizer (BEO), the Simulated Annealing is used as one of the most popular local search methods. The proposed BEO algorithm is applied to 18 UCI datasets and compared to a wide range of algorithms. The results demonstrate the superiority and merits of EO when solving feature selection problems.

Keywords: Equilibrium optimizer, Feature selection, Simulated annealing, Hybrid optimization, Meta-heuristic optimization

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 but there is a two-fold question: how much of is it useful and do we have the means to analyze that huge quantity of data? Statisticians may help to answer these questions.

\textsuperscript{*}This document is a collaborative effort.
\textsuperscript{*}Corresponding author

Email addresses: kushalkanti1999@gmail.com (Kushal Kanti Ghosh), ritanguha16@gmail.com (Ritam Guha), berasuman007@gmail.com (Suman Kumar Bera), ranjucse@gmail.com (Ram Sarkar), seyedali.mirjalili@griffithuni.edu.au (Seyedali Mirjalili)
According to [1], by the end of 2020, there will be around 40 trillion gigabytes 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 resulted in the beginning of a new field in the domain of data processing which is known as dimension reduction [2]. The motive of dimension reduction is to keep the meaningful information of a large dataset even while reducing the number of dimensions under consideration.

One of the most sought areas after dimension reduction procedures is Feature Selection (FS). 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 algorithm tries 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 [3, 4, 5, 6, 7, 8, 9] look for statistical interpretations of data to find the most informative and appropriate set of features. Wrapper methods [10, 11, 12, 13, 14, 15], on the other hand, use learning algorithms (e.g. classifiers [16]) to evaluate candidate feature subsets and guide the further searches according to the evaluation outcomes. Although wrapper methods are computationally more expensive than filter methods (due to the use of 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. These hybrid algorithms [17, 18, 19, 20] tend to perform better by improving their exploration and exploitation ability (through inclusion of local or global
search techniques). Some researchers are currently using 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 [21, 22]. 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.

In this paper, we have made an attempt to address the FS problem using a recently proposed optimization algorithm known as Equilibrium Optimizer (EO) [23]. 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 low number of features. This intuitive similarity between FS and optimization has motivated us to modify the EO and apply it to solve FS problems. The contributions of this paper are highlighted below:

1. Application of EO to FS for the first time to the best of our knowledge.
2. Modification of EO’s exploitative abilities through the use of Simulated Annealing.
3. Validation of the proposed FS framework over 18 well-known UCI datasets.

The rest of the paper is organized as follows: Section 2 provides a brief review of the similar kinds work carried out by different researchers across the globe. Section 3 provides detailed description of the proposed FS model. The results obtained by the FS version of EO are explained in Section 5. Finally, 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 from natural phenomena. Starting from the concept of chromosome formation in Genetic Algorithm (GA) to the concept of control volume mass balance in EO – there is a large number of algorithms which are derived from simple natural occurrences.

The journey of nature-inspired FS algorithms started with GA [24]. Lead et al. introduced GA to the domain of FS by mapping the continuous variant to the binary search space of FS. 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 the simplicity in nature, GA has been widely used in various optimization and FS problems. In search of proper trade-off between exploration and exploitation, researchers have proposed different variants of GA. In [25], 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 [26] 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 fitness 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 [27, 28, 29, 30, 15, 31, 32].

After the application of GA in FS, a large number of other nature-inspired optimization algorithms were modified to solve FS problems. Algorithms such as 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 optimization capabilities. PSO is the brainchild of Kennedy and Eberhart. In [33], 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
time [34, 35, 36]. ACO is based on the food searching process followed by ants in nature. Dorigo et al. have introduced ACO in [37]. Rashedi et al. have developed GSA [38] which is inspired from mass interactions and law of gravity.

All these algorithms are very popular and highly used in the FS domain. According to No-free Lunch theorem for optimization [39], however, there is no perfect algorithm to solve all optimization problems. That is why researchers continue to propose various new algorithms to solve FS problems. One of the main challenges in any FS problem is to find a proper stability between exploitation and exploration. In recent times, various other metaheuristic algorithms like Grey Wolf Optimizer (GWO) [40], Whale Optimization Algorithm (WOA) [41], Ant Lion Optimizer (ALO) [42], Salp Swarm Algorithm (SSA) [43], etc. have been developed in search of the trade-off.

EO is one of the most recent algorithms in the domain of optimization. Faramarzi et al. have proposed EO in [44]. 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 as 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. To the best of our knowledge, however, EO has not been adapted for FS problem. This has motivated us to update EO and make it applicable to solve FS problems. In this paper, we have mapped EO to the binary space of FS and applied it over 18 well-known UCI datasets to prove its usefulness in the FS domain.

3. Proposed Work

3.1. Equilibrium Optimization: An Overview

Equilibrium Optimizer (EO), first proposed in [44], is a physics based optimization algorithm inspired by dynamic source and sink models for estimating
equilibrium states. This method is based upon simple well mixed dynamic mass balance on a control volume, where a mass balance equation is used to determine the concentration of a nonreactive constituent in a control volume as a function of its various source and sink mechanisms. The generic equation of mass-balance is identified with a first order differential equation [45] given by Equation 1.

\[ V \frac{dC}{dt} = QC_{eq} - QC + G \]  

where \( V \) is the control volume, \( C \) is control volume concentration, \( C_{eq} \) is equilibrium state concentration without any generation 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 equals mass entering the system minus mass leaving the system plus mass generated inside. An equilibrium state is reached when \( V \frac{dC}{dt} \) reaches zero. A rearranged Equation 1 is given in Equation 2.

\[ \frac{dC}{\lambda C_{eq} - \lambda C + \frac{G}{V}} = dt \]  

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

\[ \int_{C_0}^{C} \frac{dC}{\lambda C_{eq} - \lambda C + \frac{G}{V}} = \int_{t_0}^{t} dt \]  

where \( c_0 \) and \( C_0 \) are initial start time and initial concentration. Equation 4 shows the result of Equation 3.

\[ C = C_{eq} + (C_0 - C_{eq})F + \frac{G}{\lambda V}(1 - F) \]  

\[ F = e^{-\lambda(t-t_0)} \]  

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 appeared till current iteration is used as equilibrium concentration and it is selected randomly from a pool, namely equilibrium pool. The second term indicates concentration difference between a solution and equilibrium state. The third term is involved with generation rate. Mathematical
expression and explanation of these components are given in the following sub-
sections.

3.1.1. Equilibrium Pool and Candidates

Equilibrium state is the convergence state of the algorithm, and it is expected
to be the global optima of the problem in consideration. At the beginning, the
equilibrium concentration is not known to the algorithm, rather equilibrium
candidates provide the particles with a search pattern. As per [44], we have
considered equilibrium pool consisting of five concentrations: four best-so-far
concentrations obtained during the process and their arithmetic mean. These
four concentrations help in exploration capability of EO and the arithmetic
mean helps in exploitation.

\[ \vec{C}_{eq,\text{pool}} = \{ \vec{C}_{eq}(1), \vec{C}_{eq}(2), \vec{C}_{eq}(3), \vec{C}_{eq}(4), \vec{C}_{eq(avg)} \} \] (6)

3.1.2. Exponential Term, F

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

\[ t = (1 - \frac{iter}{maxIter})^{a_2 \frac{iter}{maxIter}} \] (7)

where \( iter \) and \( maxIter \) are current and maximum number of iterations, re-
spectively. \( a_2 \) is a constant, higher \( a_2 \) indicates lower exploration and better
exploitation ability.

\[ \vec{F} = e^{-\vec{\lambda}(t-t_0)} \] (8)

In order to guarantee convergence the search speed needs to be slowed down,
and to perform that \( t_0 \) is considered as:

\[ \vec{t}_0 = \frac{1}{\lambda} \ln \left( -a_1 sgn(\vec{r} - 0.5)[1 - e^{-\vec{\lambda}t}] \right) + t \] (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 [44], we have set $a_1 = 2$ and $a_2 = 1$. By substituting Equation 9 in Equation 8, we obtain:

$$\vec{F} = a_1 sgn(\vec{r} - 0.5)[e^{-\vec{\lambda}t} - 1]$$  \hspace{1cm} (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:

$$\vec{G} = \vec{G}_0 e^{-\vec{\lambda}(t-t_0)} = \vec{G}_0 \vec{F}$$  \hspace{1cm} (11)

where

$$\vec{G}_0 = \overrightarrow{GCP}(\vec{C}_{eq} - \vec{\lambda} \vec{C})$$  \hspace{1cm} (12)

$$GCP = \begin{cases} 0.5r_1 & r_2 \geq GP \\ 0 & r_2 < GP \end{cases}$$  \hspace{1cm} (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 Generation rate Control Parameter, that indicates the probability of generating term’s contribution towards the concentration updating process. $GP$ means Generation Probability, that 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 [44]. So, finally the concentration updating rule in this algorithm is given by:

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

$V$ is considered as unit.

### 3.2. Simulated Annealing

In metallurgy and materials science, annealing [46] 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. Simulated Annealing (SA) [47] is a single solution based meta-heuristic algorithm which is
an enhanced version of the hill climbing [48]. 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 [17] and evaluated using fitness function. If the fitness value of the neighbor is better than current solution, then the current solution is replaced with the neighbor. If the fitness 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:

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

Here, \( \theta = \text{fit(neighbor)} - \text{fit(curr.sol)} \) is difference between the fitness value 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, \( T_{k+1} = \alpha T_k \) is adopted, \( \alpha \) is the cooling coefficient, \( \alpha \in [0, 1] \).

### 3.3. Proposed Binary EO

Now, the main challenge in a FS problem is to search for the best possible feature subset. Specially in wrapper based problem, since each feature subset is needed to be evaluated using a learning algorithm (classifier), searching for the best feature subset is highly time consuming. So, 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 [44] 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 FS problem. Binary search space implies [17] 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. We need to use a transfer function \([49]\) to propose the binary version of EO. Different transfer functions are proposed in \([49]\). Amongst those, we have chosen, a V-shaped transfer function to propose binary EO (BEO) due to its significant impact on a wide range of meta-heuristics. The used transfer function, depicted in Figure 1, is given by Equation 16.

\[
T(x) = \frac{x}{\sqrt{1 + x^2}}
\]  

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

\[
X_d^t = \begin{cases} X_{d - 1}^t & \text{if } rno < T(X_d^t) \\ X_d^t & \text{if } rno \geq T(X_d^t) \end{cases}
\]  

where \(X_d^t\) is the \(d^{th}\) dimension of the concentration in the \(t^{th}\) iteration, \(X_{d - 1}^t\) is the \(d^{th}\) dimension of the concentration in the \((t - 1)^{th}\) iteration, \(rno\) is a random number, \(rno \in [0, 1]\).

EO performs exploitation mainly using the Equilibrium pool \([44]\). This pool controls both exploration and exploitation. During initial iterations, distance
among the equilibrium candidates is high and using these candidates to update
the concentrations helps perform global search. With the increasing number of
iterations, the equilibrium candidates come closer to each other and now, using
these candidates to update the concentrations helps perform local search encir-
cling the candidates, which results in exploitation. Exploration is exclusively
taken care of using Generation Probability (GP, Equation 13). But there is no
such thing to consider exploitation particularly. So, we have incorporated the
concept of SA to perform local search i.e., take care of exploitation specifically.
The hybrid version of EO is labeled as BEOSA (binary EO with SA).

FS is considered as a multi-objective optimization problem due to two dif-
ferent criteria to evaluate the feature subset in consideration [50]: classification
accuracy and number of selected features. To be specific, the objective of FS is
to achieve maximum classification accuracy with minimum number of features.
These two criteria are of contradictory purposes [51], so we have considered the
classification error rate instead of accuracy. Equation 18 aggregates both of the
objectives together and converts the FS problem into a single objective problem.

\[
\text{Fitness} = \eta \times \epsilon + \mu \times \frac{|v|}{|D|}
\]  

(18)

where, \( \epsilon \) denotes the classification error rate, \(|v|\) represents the number of fea-
tures in the subset being evaluated and \(|D|\) represents total number of features
in the dataset. \( \mu \) and \( \eta \) respectively represent the weight (importance) of the
subset length and the classification error, and \( \eta + \mu = 1 \). We have used K-Nearest
Neighbor (KNN) classifier [52] to compute the classification error (\( \epsilon \)).

4. Experimental Results

This section explains the experiments used to prove the applicability of EO in
FS domain. We have used K-nearest Neighbors Algorithm (KNN) [52] classifier
to measure classification accuracy. As per the recommendations in [50, 51, 17],
we have set \( K = 5 \). For each dataset, 80% of the instances are used for training
the model and rest 20% are used for testing. Proposed method is implemented
using Python3 [53] and graphs are plotted using Matplotlib [54].
Table 1: Description of the datasets used in the present work

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

4.1. Dataset Description

In order to assess the performance of EO, 18 UCI datasets [55] have been used. The datasets are selected from various different backgrounds. The description of these datasets are presented in Table 1. Table 1 shows that there are 15 bi-class and 3 multi-class datasets. The datasets are diverse in terms of both number of attributes (features) and instances. These variances help in providing proper evaluation of the proposed algorithm.

4.2. Parameter Tuning

There are two parameters which are always very important for any multi-agent evolutionary algorithm: population size and number of maximum itera-
tions. Population size characterizes how a single agent learns from other agents’ experience and iterations provide step-wise evolution of the agents. In order to find the proper values for these two parameters, experiments have been performed by varying one parameter \textit{w.r.t.} the other.

Population size for BEO and BEOSA, is varied as $[5, 10, 20, 30, 50]$. Number of maximum iterations is set as 50, which we have determined from experiments. Classification accuracies are reported in Figure 2 with different population size.

Considering that computational time increases with increasing population size and the accuracies provided in Figure 2, we have decided to fix population size as 20 for further experiments. Figure 3 shows how the fitness values of BEO and BEOSA change with increasing number of iterations.

4.3. Results and Discussion

In this section, we have provided the results obtained by BEO and BEOSA over the datasets mentioned in Section 4.1. From Table 2, it is clear that BEO and BEOSA are able to perform FS efficiently. For all the datasets, classification accuracy has improved after FS (with huge margins in multiple cases).

BEO has achieved more than or equal to 90% accuracy for 14 datasets whereas for BEOSA the count is 15. Moreover, BEO has got 100% accuracy for two datasets while BEOSA has scored full for eight datasets which is quite remarkable. If we keep aside the classification accuracy and focus on the number features used to obtain these impressive results, even then BEO and BEOSA have performed exceptionally well. Almost on all the datasets, BEO and BEOSA have used only 50% or even lesser number of features for classification. Observing the results in Table 2, it quite obvious that the use of SA is significantly helping the algorithm to explore different parts of the search space and achieve better results.

In case of Exactly, HeartEW, Sonar and Wine datasets, both the variants of EO have shown massive increase in classification accuracy after FS. For the rest of the datasets, we can see increases between 1-10% which are quite significant.
Figure 2: Population variation for 18 UCI datasets
Figure 3: Convergence graphs for 18 UCI datasets
From this discussion, we can conclude that BEO and BEOSA are able to successfully select the optimal set of features from the datasets. The accuracy provided by both the models using very few features make both of them very competitive in the field of FS.

Overall we can say that although BEO and BEOSA perform really well in FS, the results obtained by BEOSA are better than BEO. Hence, it can be concluded that SA plays a significant role in improving performance of EO. Basically, in EO, 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 balance between exploration and exploitation is maintained by the exponential term mentioned in section 3.1.2. But, there may be situations when the particles in the equilibrium pool become similar in nature which indicate that they belong to the same part of the search space. As a consequence, it leads to the 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 which actually helps EO to bring in exploration of the search space, thereby aiding EO to perform well. In this context it is to be noted that even though SA increases the exploration ability of EO, but the exploration-exploitation trade-off remains intact due to the usage of exponential term.

5. Comparison

In this section, we present the results of the proposed BEOSA and highlight how it performs w.r.t. state-of-the-art. We have compared the results of BEOSA with three recent works: GSA based [56], ALO based [57], and GWO and WOA based [50].

Figure 4 shows the performance of BEOSA in terms of achieved classification accuracy. Inspecting Figure 4, it can be observed that BEOSA performs best in 12 cases (66.7%). In case of BreastEW, HeartEW and KrVsKpEW, it has
Table 2: Performance of BEO and BEOSA in terms of classification accuracy and no. of selected features

| Sl. no. | Dataset       | Original | BEO       | BEOSA      |
|---------|---------------|----------|-----------|------------|
|         |               | Accuracy | #Features | Accuracy   | #Features | Accuracy | #Features |
| 1       | BreastCancer  | 94.4     | 9         | 98.57      | 2         | 99.29    | 3         |
| 2       | BreastEW      | 96.49    | 30        | 96.49      | 15        | 98.25    | 13        |
| 3       | CongressEW    | 96.55    | 16        | 98.85      | 9         | 100      | 5         |
| 4       | Exactly       | 77.5     | 13        | 99.5       | 7         | 100      | 6         |
| 5       | Exactly2      | 77       | 13        | 78         | 6         | 79.5     | 5         |
| 6       | HeartEW       | 75.92    | 13        | 87.04      | 7         | 90.74    | 6         |
| 7       | Ionosphere    | 91.4     | 34        | 92.86      | 7         | 97.14    | 2         |
| 8       | KrVsKpEW      | 96.7     | 36        | 96.24      | 18        | 97.65    | 20        |
| 9       | Lymphography  | 83.33    | 18        | 90         | 5         | 96.67    | 9         |
| 10      | M-of-n        | 93       | 13        | 99.5       | 7         | 100      | 6         |
| 11      | PenglungEW    | 93.7     | 325       | 100        | 161       | 100      | 59        |
| 12      | Sonar         | 83.33    | 60        | 92.86      | 26        | 100      | 29        |
| 13      | SpectEW       | 88.88    | 22        | 90.74      | 12        | 94.44    | 12        |
| 14      | Tic-tac-toe   | 71.5     | 9         | 79.17      | 4         | 80.21    | 4         |
| 15      | Vote          | 95       | 16        | 98.33      | 5         | 100      | 4         |
| 16      | WaveformEW    | 82.9     | 40        | 83         | 18        | 83.7     | 19        |
| 17      | Wine          | 77.78    | 13        | 100        | 6         | 100      | 3         |
| 18      | Zoo           | 95       | 16        | 95         | 7         | 100      | 7         |
Figure 4: Comparison of Classification Accuracy for 18 UCI datasets
performed as the second best. ASGW, HSGW and HGSA performs best respectively in case of BreastEW, HeartEW and KrVsKpEW. For Ionosphere, BEOSA has performed as third best following the results of RSGW and ASGW, in this order. For Exactly2 and Tic-tac-toe, BEOSA has sixth best results. BEOSA outperforms BALOS-CE and BGSA for all the 18 datasets. In Figure 5, we have reported the average classification accuracy achieved by each method over all the used 18 UCI datasets. In terms of average accuracy, BEOSA has achieved 95.4% classification accuracy and this is the highest among others.

Figure 6 shows the performance of BEOSA in terms of another critical aspect of FS and the very purpose of this research field, number of selected features. BEOSA has selected lowest number of features in 4 cases: IonosphereEW, Tic-tac-toe, WaveformEW, and Wine. It has performed second best in case of 6 datasets. On average over all 18 datasets, BEOSA selects 11.78 features, which is second best, following HGSA (10.8).

In order to determine the significance of the obtained results, we have performed Wilcoxon rank sum test [58] with 5% significance level for each pair of methods used in this section. Table 3 shows the obtained $p-values$, $p \leq 0.05$ are shown in bold.
Figure 6: Comparison of no. of selected features for 18 UCI datasets
Figure 7: Average number of features selected by each method
Table 3: p-values of the Wilcoxon rank sum test

|             | BEOSA | bALOS-QR | bALOV-QR | bALO-S | bALO-V | bALOS-CE | bALOV-CE | BGSA    | HGSA   | HSGW   | RSGW   | ASGW   |
|-------------|-------|----------|----------|--------|--------|----------|----------|---------|--------|--------|--------|--------|--------|
| BEOSA       | -     | 0.000    | 0.004    | 0.001  | 0.001  | 0.000    | 0.000    | 0.000   | 0.023  | 0.012  | 0.020  |        |        |
| bALOS-QR    | 0.000 | -        | 0.000    | 0.000  | 0.000  | 0.314    | 0.000    | 0.000   | 0.000  | 0.000  | 0.000  | 0.000  | 0.000  |
| bALOV-QR    | 0.004 | 0.000    | -        | 0.012  | 0.305  | 0.001    | 0.433    | 0.070   | 0.102  | 0.052  | 0.102  | 0.039  |        |
| bALO-S      | 0.001 | 0.000    | 0.012    | -      | 0.001  | 0.000    | 0.070    | 0.006   | 0.005  | 0.005  | 0.008  | 0.004  |        |
| bALO-V      | 0.001 | 0.000    | 0.305    | 0.001  | -      | 0.000    | 0.355    | 0.040   | 0.015  | 0.040  | 0.053  | 0.013  |        |
| bALOS-CE    | 0.000 | 0.314    | 0.001    | 0.000  | 0.000  | -        | 0.000    | 0.000   | 0.000  | 0.000  | 0.000  | 0.000  |        |
| bALOV-CE    | 0.000 | 0.000    | 0.433    | 0.070  | 0.355  | 0.000    | -        | 0.295   | 0.214  | 0.016  | 0.067  | 0.035  |        |
| BGSA        | 0.000 | 0.000    | 0.070    | 0.006  | 0.040  | 0.000    | 0.295    | -       | 0.177  | 0.016  | 0.139  | 0.076  |        |
| HGSA        | 0.000 | 0.000    | 0.102    | 0.005  | 0.015  | 0.000    | 0.214    | 0.177   | -      | 0.088  | 0.170  | 0.191  |        |
| HSGW        | 0.023 | 0.000    | 0.052    | 0.005  | 0.040  | 0.000    | 0.016    | 0.016   | 0.088  | -      | 0.587  | 0.959  |        |
| RSGW        | 0.012 | 0.000    | 0.102    | 0.008  | 0.053  | 0.000    | 0.067    | 0.139   | 0.170  | 0.587  | -      | 0.865  |        |
| ASGW        | 0.020 | 0.000    | 0.039    | 0.004  | 0.013  | 0.000    | 0.035    | 0.076   | 0.191  | 0.959  | 0.865  | -      |        |
6. Conclusion

In this work, we made an effort to propose a binary variant of EO to make it applicable to the field of FS. To map the continuous values of EO to a binary search space, a V-shaped transfer function was used. Apart from the basic variant of EO (BEO), we also proposed a modification by hybridizing BEO with SA named as BEOSA. Both the proposed models were tested on 18 popular UCI datasets. From the results and comparison with other contemporary FS algorithms, we can conclude that BEO and BEOSA have performed significantly well. We further carried out a test for the statistical significance of the obtained results using Wilcoxon rank sum test. The test results also indicate that the performance of BEO and BEOSA is outstanding. Thus, through this rigorous experimentation, it is proved that the binary variant of EO works pretty good in the FS domain and 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 Facial Emotion Recognition, Microarray etc.

References

[1] C. Petrov, Big data statistics 2020.
URL https://techjury.net/stats-about/big-data-statistics/#gref

[2] B. Chizi, O. Maimon, Dimension reduction and feature selection, in: Data Mining and Knowledge Discovery Handbook, Springer-Verlag, 2005, pp. 93–111. doi:10.1007/0-387-25465-x_5.

[3] K. Kira, L. A. Rendell, A practical approach to feature selection, in: Machine Learning Proceedings 1992, Elsevier, 1992, pp. 249-256. doi: 10.1016/b978-1-55860-247-2.50037-1.

[4] I. Kononenko, Estimating attributes: Analysis and extensions of RELIEF,
[5] M. Robnik-Šikonja, I. Kononenko, Theoretical and empirical analysis of relief and rrelief, Machine learning 53 (1-2) (2003) 23–69.

[6] M. Hall, Correlation-based feature selection for machine learning, Department of Computer Science 19.

[7] L. Yu, H. Liu, Feature selection for high-dimensional data: A fast correlation-based filter solution, in: Proceedings of the 20th international conference on machine learning (ICML-03), 2003, pp. 856–863.

[8] Z. Zhao, H. Liu, Searching for interacting features in subset selection, Intelligent Data Analysis 13 (2) (2009) 207–228.

[9] S. Ghosh, S. Bhowmik, K. K. Ghosh, R. Sarkar, S. Chakraborty, A filter ensemble feature selection method for handwritten numeral recognition, EMR (2016) 007213.

[10] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in Engineering Software 69 (2014) 46–61. doi:10.1016/j.advengsoft.2013.12.007.

[11] S. Mirjalili, A. Lewis, The whale optimization algorithm, Advances in Engineering Software 95 (2016) 51–67. doi:10.1016/j.advengsoft.2016.01.008.

[12] S. Mirjalili, The ant lion optimizer, Advances in Engineering Software 83 (2015) 80–98. doi:10.1016/j.advengsoft.2015.01.010.

[13] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowledge-Based Systems 89 (2015) 228–249. doi:10.1016/j.knosys.2015.07.006.

[14] S. Mirjalili, Sca: a sine cosine algorithm for solving optimization problems, Knowledge-based systems 96 (2016) 120–133.
[15] M. Ghosh, S. Adhikary, K. K. Ghosh, A. Sardar, S. Begum, R. Sarkar, Genetic algorithm based cancerous gene identification from microarray data using ensemble of filter methods, Medical & Biological Engineering & Computing 57 (1) (2018) 159–176. doi:10.1007/s11517-018-1874-4.

[16] K. Allan, Classifiers, Language 53 (2) (1977) 285–311. doi:10.1353/lan.1977.0043.

[17] M. M. Mafarja, S. Mirjalili, Hybrid whale optimization algorithm with simulated annealing for feature selection, Neurocomputing 260 (2017) 302–312. doi:10.1016/j.neucom.2017.04.053.

[18] H. Jia, J. Li, W. Song, X. Peng, C. Lang, Y. Li, Spotted hyena optimization algorithm with simulated annealing for feature selection, IEEE Access 7 (2019) 71943–71962. doi:10.1109/access.2019.2919991.

[19] H. H. Inbarani, A. T. Azar, G. Jothi, Supervised hybrid feature selection based on PSO and rough sets for medical diagnosis, Computer Methods and Programs in Biomedicine 113 (1) (2014) 175–185. doi:10.1016/j.cmpb.2013.10.007.

[20] S. S. Kannan, N. Ramaraj, A novel hybrid feature selection via symmetrical uncertainty ranking based local memetic search algorithm, Knowledge-Based Systems 23 (6) (2010) 580–585. doi:10.1016/j.knosys.2010.03.016.

[21] M. Ghosh, R. Guha, R. Sarkar, A. Abraham, A wrapper-filter feature selection technique based on ant colony optimization, Neural Computing and Applications (2019) 1–19.

[22] I. Guyon, S. Gunn, A. Ben-Hur, G. Dror, 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, MIT Press, 2005, pp. 545–552.
[23] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, Knowledge-Based Systems 191 (2020) 105190. doi:10.1016/j.knosys.2019.105190.

[24] R. Leardi, Genetic algorithms in feature selection, in: Genetic Algorithms in Molecular Modeling, Elsevier, 1996, pp. 67–86. doi:10.1016/b978-012213810-2/50004-9.

[25] B. Ma, Y. Xia, A tribe competition-based genetic algorithm for feature selection in pattern classification, Applied Soft Computing 58 (2017) 328–338. doi:10.1016/j.asoc.2017.04.042.

[26] S. Jiang, K.-S. Chin, L. Wang, G. Qu, K. L. Tsui, Modified genetic algorithm-based feature selection combined with pre-trained deep neural network for demand forecasting in outpatient department, Expert Systems with Applications 82 (2017) 216–230. doi:10.1016/j.eswa.2017.04.017.

[27] J. Yang, V. Honavar, Feature subset selection using a genetic algorithm, in: Feature Extraction, Construction and Selection, Springer US, 1998, pp. 117–136. doi:10.1007/978-1-4615-5725-8_8.

[28] J. Huang, Y. Cai, X. Xu, A hybrid genetic algorithm for feature selection wrapper based on mutual information, Pattern Recognition Letters 28 (13) (2007) 1825–1844. doi:10.1016/j.patrec.2007.05.011.

[29] I.-S. Oh, J.-S. Lee, B.-R. Moon, Hybrid genetic algorithms for feature selection, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (11) (2004) 1424–1437. doi:10.1109/tpami.2004.105.

[30] W. Siedlecki, J. Sklansky, A NOTE ON GENETIC ALGORITHMS FOR LARGE-SCALE FEATURE SELECTION, in: Handbook of Pattern Recognition and Computer Vision, WORLD SCIENTIFIC, 1993, pp. 88–107. doi:10.1142/9789814343138_0005.

[31] M. Ghosh, R. Guha, R. Mondal, P. K. Singh, R. Sarkar, M. Nasipuri, Feature selection using histogram-based multi-objective GA for handwrit-
ten devanagari numeral recognition, in: Advances in Intelligent Systems and Computing, Springer Singapore, 2018, pp. 471–479. doi:10.1007/978-981-10-7566-7_46.

[32] R. Guha, M. Ghosh, S. Kapri, S. Shaw, S. Mutsuddi, V. Bhateja, R. Sarkar, Deluge based genetic algorithm for feature selection, Evolutionary Intelligence doi:10.1007/s12065-019-00218-5.

[33] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN’95 - International Conference on Neural Networks, Vol. 4, 1995, pp. 1942–1948 vol.4. doi:10.1109/ICNN.1995.488968.

[34] B. Tran, B. Xue, M. Zhang, Variable-length particle swarm optimization for feature selection on high-dimensional classification, IEEE Transactions on Evolutionary Computation 23 (3) (2019) 473–487. doi:10.1109/tevc.2018.2869405.

[35] L. Shang, Z. Zhou, X. Liu, Particle swarm optimization-based feature selection in sentiment classification, Soft Computing 20 (10) (2016) 3821–3834. doi:10.1007/s00500-016-2093-2.

[36] P. Moradi, M. Gholampour, A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy, Applied Soft Computing 43 (2016) 117–130. doi:10.1016/j.asoc.2016.01.044.

[37] M. Dorigo, G. Di Caro, Ant colony optimization: a new meta-heuristic, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Vol. 2, 1999, pp. 1470–1477 Vol. 2. doi:10.1109/CEC.1999.782657.

[38] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A gravitational search algorithm, Information Sciences 179 (13) (2009) 2232–2248. doi:10.1016/j.ins.2009.03.004.

[39] D. H. Wolpert, W. G. Macready, et al., No free lunch theorems for search, Tech. rep., Technical Report SFI-TR-95-02-010, Santa Fe Institute (1995).
[40] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in Engineering Software 69 (2014) 46–61. doi:10.1016/j.advengsoft.2013.12.007.

[41] S. Mirjalili, A. Lewis, The whale optimization algorithm, Advances in Engineering Software 95 (2016) 51–67. doi:10.1016/j.advengsoft.2016.01.008.

[42] S. Mirjalili, The ant lion optimizer, Advances in Engineering Software 83 (2015) 80–98. doi:10.1016/j.advengsoft.2015.01.010.

[43] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Advances in Engineering Software 114 (2017) 163–191. doi:10.1016/j.advengsoft.2017.07.002.

[44] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, Knowledge-Based Systems 191 (2020) 105190. doi:10.1016/j.knosys.2019.105190.

[45] W. W. Nazaroff, L. Alvarez-Cohen, Environmental engineering science, Wiley, New York, 2001.

[46] P. J. M. van Laarhoven, E. H. L. Aarts, Simulated annealing, in: Simulated Annealing: Theory and Applications, Springer Netherlands, 1987, pp. 7–15. doi:10.1007/978-94-015-7744-1_2.

[47] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680. doi:10.1126/science.220.4598.671.

[48] D. H. Ackley, A Connectionist Machine for Genetic Hillclimbing, Springer US, 1987. doi:10.1007/978-1-4613-1997-9.

[49] S. Mirjalili, A. Lewis, S-shaped versus v-shaped transfer functions for binary particle swarm optimization, Swarm and Evolutionary Computation 9 (2013) 1–14. doi:10.1016/j.swevo.2012.09.002.
[50] M. Mafarja, A. Qasem, A. A. Heidari, I. Aljarah, H. Faris, S. Mirjalili, Efficient hybrid nature-inspired binary optimizers for feature selection, Cognitive Computation 12 (1) (2019) 150–175. doi:10.1007/s12559-019-09668-6.

[51] E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary grey wolf optimization approaches for feature selection, Neurocomputing 172 (2016) 371–381. doi: 10.1016/j.neucom.2015.06.083.

[52] N. S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, The American Statistician 46 (3) (1992) 175. doi: 10.2307/2685209.

[53] G. van Rossum, F. L. Drake, The Python Language Reference Manual, Network Theory Ltd., 2011.

[54] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing in Science & Engineering 9 (3) (2007) 90–95. doi:10.1109/MCSE.2007.55.

[55] C. Blake, Cj merz uci repository of machine learning databases, University of California at Irvine.

[56] M. Taradeh, M. Mafarja, A. A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, H. Fujita, An evolutionary gravitational search-based feature selection, Information Sciences 497 (2019) 219–239. doi:10.1016/j.ins.2019.05.038.

[57] M. M. Mafarja, S. Mirjalili, Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection, Soft Computing 23 (15) (2018) 6249–6265. doi:10.1007/s00500-018-3282-y.

[58] F. Wilcoxon, Individual comparisons by ranking methods, in: Breakthroughs in statistics, Springer New York, 1992, pp. 196–202. doi: 10.1007/978-1-4612-4380-9_16.