www.revistageintec.net
ISSN: 2237-0722
revista GEINTEC
Gestão, Inovação e Tecnologias

Improved Grasshopper Optimization Algorithm based Feature Selection with Evolutionary Outlay-Aware Deep Belief Network Classifier (IGOA-EOA-DBNC) for High Dimensional Datasets

Dr.M. Praveena[1]; Dr.V. Jaiganesh[2]

[1]Assistant Professor, Department of Computer Science, Dr.SNS Rajalakshmi College of Arts and Science, Coimbatore, Tamil Nadu, India.

[2]Professor, Department of Computer Science, Dr.NGP Arts and Science College, Coimbatore, Tamil Nadu, India.

Abstract

Background: *High dimensional datasets contain the curse of dimensionality, and hence data mining becomes a more difficult task. Feature selection in the knowledge data and discovery process provides a solution for this curse of dimensionality issue and helps the classification task reduce the time complexity and improve the accuracy.*

Objectives: *This paper aims to recognize a bio-inspired algorithm that best suits feature selection and utilizes optimized feature selection techniques. This algorithm is used to design machine learning classifiers that are suitable for multiple datasets and for both high dimensional datasets, moreover to carry out performance analysis with regards to the accuracy of a classification and the processing time for classification.*

Methods: *This study employs an improved form of grasshopper optimization algorithm to perform feature selection task. Evolutionary outlay aware deep belief network is used to perform the classification task.* ***Findings:*** *In this research, 20 UCI benchmark data sets are taken with full 60 features and 30000 instances. The datasets are Mammography, Monks-1, Bupa, Credit, Parkinson's, Monk-2, Sonar, Ecoli, Prognostic, Ionosphere, Monk-3, Yeast, Car, Blood, Pima, Spect, Vert, Prognostic, Contraceptive, and Tic-Tac-Toe endgame. Table 1 describes the dataset details, number of instances, datasets and features. The overall performance is performed using MATLAB 6.0 tool, which runs on Microsoft Windows 8, and the configuration is Core 13 processor with 1 TB hard disk and 8GB RAM. Performance standards, like classification accuracy and the processing time for classification, is achieved.*

Novelty: *Interestingly, the Improved Grasshopper Optimization Algorithm uses error rate and classification accuracy of the Evolutionary Outlay Aware –Deep Belief Network Classifier as fitness function values. This combined work of classification and feature selection is briefly represented as IGOA-EOA-DBNC. Twenty datasets are selected for testing the performance regarding elapsed time and accuracy, which gives better results.*

Received: 22.03.2021 – Accepted: 25.04.2021

Key-words: Data Mining, Feature Selection, Grasshopper Optimization Algorithm, Deep Belief Network, Evolutionary Algorithm, Classification Accuracy, Elapsed Time.

1. Introduction

In computer science, Data mining is one of the effective methods, likewise data innovation. The data mining process removes invisible data from the dataset with the help of the learning data disclosure movement for the last two decades. Machine Learning algorithms are combined to perform data mining assignments. This algorithm increases driving ramifications in the research field. ML is one of the type of artificial intelligence (AI) that enables a computer to be prepared without being directly customized. ML learning is based on disseminating computer programmes that can adapt and evolve in response to new data over time. The three types of ML algorithms are support, supervised and unsupervised learning. The enhancement of ML is interconnected to data mining [1]. Machine learning and data mining look to adjust an enquiry from beginning to end, for examples. ML uses data to identify design in data and adjust program activities from now on to distinguish data from human knowledge in data mining applications.

The assignment of deducting importance from named preparing data which consists of a collection preparing precedents, is known as Supervised Machine Learning. In this, each occurrence involves an information object (typically a vector amount) and fundamental yield esteem(likewise be suggested as a supervisory flag). Using the preparation data, the supervised learning algorithm performs investigation tasks and builds a mapping precedent capacity. This algorithm determines the Classmark for secured cases and the identical conditions with precision.

The supervised algorithm creates more minor complexes from the preparation data to suggest circumstances in an "adjusted" scheme. This algorithm is used in various utilization regions, including fabricating, securities exchange prediction, money, testing, promoting, etc. Improved Grasshopper Optimization Based Feature Selection with Evolutionary Outlay-Aware Deep Belief Network Classification (IGOA-EOA-DBNC) for important datasets results from this research. To reduce the classification time and to increase the prediction accuracy, the IGOA-EOA-DBNC is used. The remainder of the paper is laid out as follows: The work is introduced in Section 1. Section 2 discusses some of the associated works. Section 3 explains the proposed works. Discussions and outcomes are presented in Section 4. Section 5 contains the conclusions. The remainder of the paper is laid out as follows: The work is introduced in first section. Section 2 discusses some of the associated works. Section 3 explains the proposed works. Discussions and outcome are presented in Section 4. Section 5 contains the conclusion.

2. Proposed Work

2.1. Outline of Grasshopper Optimization Algorithm

The two main stages of nature-inspired algorithms are exploration and utilization, aiming to increase the convergence speed and avoid local optima when trying to find a target. The search agents controlled to operate within the search space. The search agent moves quickly during the discovery process. Grasshoppers like normal objective, are responsible for all processes (food source). The simulation of Grasshopper[13] character is expressed as

$$X_i = r_1 S_i + r_2 G_i + r_3 A_i \tag{1}$$

The i-th position of Grasshopper is given as X_i the social interaction S_i is explained in the 2nd equation., where G_i is the force of gravity on the i-th Grasshopper, the wind advection A_i is defined in Eq.[5]. Within the interval [0,1] the random numbers are expressed as r_1, r_2, and r_3

$$S_i = \sum_{j=1,\, j\neq i}^{N} s\left(d_{ij}\right)\hat{d}_{ij} \tag{2}$$

The Euclidean distance d_{ij} is the distance between j-th and i-th grasshopper, which is derived as

$$d_{ij} = \left|x_j - x_i\right| .$$

Where, the unit vector from i-th to j-th Grasshopper is calculated as, $\hat{d}_{ij} = \frac{x_j - x_i}{d_{ij}}$. The force strength is given as S, and it is computed as below in Eq.3,

$$s(r) = f e^{\frac{-r}{l}} - e^{-r} \tag{3}$$

The attraction intensity is represented as f, and the attractive length scale is 1. More details on the impact of attraction and repulsion on the character of artificial grasshoppers can be found in the original GOA paper (Saremi et al., 2017). The authors conducted extensive experiments to examine the characteristics of grasshoppers with varying values of 1 and f, and they discovered that repulsion can occur between any two grasshoppers if their distance is within the range [0,2.079]. When a grasshopper is 2.079 units apart from other Grasshopper, it reaches the comfort zone.

$$G_i = -g \times \hat{e}_g \qquad (4)$$

'g' represents the gravitational constant. A unity vector pointing towards the earth's centre is expressed as $\hat{e}_g$.

$$A_i = u \times \hat{e}_w \qquad (5)$$

In which u stands for constant drift and a unit vector in the wind's direction is defined as $\hat{e}_w$. Since nymph grasshoppers lack wings, wind direction has a significant impact on their movements. The i-th Grasshopper's new location is determined by the position of all other grasshoppers. The position of the target (food source) and its current position is described in Eq.(6).

$$X_i^d = c_1\left(\sum_{j=1,\, j\neq i}^{N} c_2 \frac{ub_d - lb_d}{2} s\left(\left|x_j^d - x_i^d\right|\right)\frac{x_j - x_i}{d_{ij}}\right) + \hat{T}_d \quad (6)$$

In which the D^{th} dimension upper bound is given as ub_d , where the lb_d is the lower bound in the D^{th} dimension, the target the value of D th dimension value is defined as $s(d) = fe\overset{-d}{T} - e^{-d};\hat{T}_d$ and, the decreasing coefficient c contract the attraction region, repulsion region and the comfort zone. Consider that S is the same as the S portion in Eq.(1). The gravity component G, on the other hand, has been ignored, and the wind direction A accepts the target $\hat{T}_d$. The adaptive parameter c has been used twice in Eq.6 to model grasshoppers decelerating as they approach and eventually consume a food source. The outer c_1 is used to decrease search coverage for the target grasshopper as the iteration count increases, and then inner c_2 is used to reduce the effect of repulsion and attraction between grasshoppers, which is proportional to the number of iterations.

To manage the degree of exploratory and using characters in GOA, the parameter c is reduced proportionally to the number of executed iterations. In this process, the utilization degree increases with the increase in iteration count. The comfort zone is minimized proportionally to the number of iterations. In Eq.7, parameter c is computed

$$c = c_{Max} - l\frac{c_{Max} - c_{Min}}{L} \qquad (7)$$

Here L denotes the maximum number of iterations which is given as 1, and c_{Max}, c_{Min} defines the maximum and minimum values of parameter c, which are mentioned as 1 and 0.00001, respectively.

In Algorithm 1, the error code of the GOA algorithm is represented, and set of grasshoppers are generated, and the fitness for these generated grasshoppers are calculated during the search process. The position of the best Grasshopper is upgraded in each iteration until the stop condition is met. The accurate value of the global optimum is depicted when the best Grasshopper is found.

Algorithm–1: Grasshopper Optimization Algorithm

```
Set up $c_{Max}, c_{Min}$ ,Max and Min Iterations
Establish a population of solutions $X_i(i = 1, 2, ..., n)$
Evaluate solution in each population
Assume T as best solution
while t < Max Iterations do
    Upgrade c
    for each solution, do
            Equalize the distance in-between grasshoppers in [1, 4]
            Upgrade location of the current solution
            If the present Grasshopper leaves the boundaries, return it.
    If there exists a better solution in population, then upgrade T
    t = t + 1
return T
```

2.2. Improved Grasshopper Optimization Algorithm

It is difficult to find the accurate feature subset FS in the wrapper-based method because the selected subset is needed to be evaluated for each optimization process of the population. Therefore, a proper optimization method is required in order to reduce the number of evaluations. Thus, the GOA algorithm is used, and the search area is marked by the binary values [0,1] based on FS. As the binary values are easy to use in the calculation process, the problems in the FS can be resolved by using the efficient grasshopper algorithm.

The position of each Grasshopper is changed on the basis of the position of other grasshoppers, which is calculated using Eq. (6). The link between the present Grasshopper and the target is shown by the mutation rate which is varied according to the vector. If the elements are within

ISSN: 2237-0722
Vol. 11 No. 2 (2021)
Received: 22.03.2021 – Accepted: 25.04.2021

the boundary of the binary search area, the corresponding features of the target are extracted or else the grasshoppers are returned back to the search area with a probability of 50% as found in Eq.(8).

$$X_{t+1}^{d} = \begin{cases} Target_{t}^{d} \\ 1 \quad r_3 \geq 0.5 \quad \Delta X_{t}^{d} \geq 0 \\ 0 \quad r_3 < 0.5 \quad \Delta X_{t}^{d} < 0 \end{cases} \tag{8}$$

here X_{t+1}^{d} refers to d^{th} the dimension of Grasshopper, X_{t}^{d} shows the value of d^{th} dimension in step vector and r_3 denotes a random number with values [0,1]. The first portion of Eq.(8) depicts the charge of the utilization process, while the second and third portions show the sudden adjustments made to the solution in the various search areas. The improved pseudo-code method is explained in algorithm 2 as below.

Algorithm – 2: Improved Grasshopper Optimization Algorithm

```
Initialize $c_{Max}$, $c_{Min}$, and Max _Iterations
Initialize $X_i (i = 1, 2, ..., \dim)$ for evaluating solution in each population
Set **Target** as the best solution
**while** t < Max_Iterations **do**
    Upgrade **c**
    **for** every solution, **do**
        Control the distances between grasshoppers
        Step vector $\Delta X_t$ of the current solution is updated using Eq. 8
        **for** i=1: dim **do**
            if $\Delta X_t(i) > 0$ then
                $X_{t+1}(i) = TargetPosition(i);$ ;
            **else**
                **if** rand > = 0. 5 then
                    $\Delta X_{t+1}(i) = 1;$
                else
                    $\Delta X_{t+1}(i) = 0;$
        t=t+1
```

Return Target

By carrying out the process of mutation along with a new location with a proper mutation rate, the accurate solution is found by updating the obtained solutions in the IGOA approach. The premature convergence is avoided by the mutation operator so as to improve the diverseness of the algorithm. A high mutation rate is used for controlling the mutation operator, increasing the chance of searching more positions in the search area and limiting the population from converging to optimal solutions. Likewise, the lower mutation rate may cause the population to fall in local optima rather than global optima. The mutation rate r is linearly reduced from 0.9 to 0 on the basis of iteration i. and is calculated using eq.(9) as follows.

$$r = 0.9 + \frac{-0.9 * (i - 1)}{Max\_Iteration - 1} \qquad (9)$$

2.2.1. Fitness Function

The evolution process of this method uses a learning algorithm like EOA-DBNC, and so the accuracy of classification of selected features are increased based on the fitness function. By selecting the proper subset of various features, the classification will become accurate. So, the number of selected attributes are needed to be reduced, and the accuracy in classification is to be improved by the designed FS approach. Such that the fitness function is calculated using Eq.(10) for attaining the above-mentioned advantages of FS approach. Also, the error rate in classification can be efficiently reduced. The fitness function is calculated as below,

$$Fitness = \alpha \gamma_R (D) + \beta \frac{|R|}{|N|} \qquad (10)$$

Here the error rate is referred as | R |, the number of selected features is represented as| N |, the length of the subset is denoted as D, and the two parameters $\alpha \in [0,1]$ and $\beta = (1 - \alpha)$.

2.3. Background of DBN

The parameters in original data is obtained by every layer of the hidden variable so as to reduce the higher-order correlation. The parameters of the visible variables are arranged in the lower layer of DBN while solving the classification issues by using DBN. The probability distribution is created over the labels of data by the DBN.

Assign the values for the data set $S = \{\{x_1, y_1\}, \{x_2, y_2\}, ..., \{x_N, y_N\}\}$ comprising a total number of N record pairs $\{x_n, y_n\}$, which x_n and y_n denotes the nth data sample and target label, and H shows the hidden layers with a complex feature mapping function. The classification is performed by the soft-max layer, which is the outer layer of DBN and is characterized as $\theta_s = \{W_s, b_s\}$. If there are K neurons in the outer layer, then the prediction probability of class j is found, and comprising weights $W_s^{(j)}$ and bias $b_s^{(j)}$,

$$P(y = j \mid x) = \frac{\exp\left(b_s^{(j)} + x_H^T W_s^{(j)}\right)}{\sum_{k=1}^{K} \exp\left(b_s^{(k)} + x_H^T W_s^{(k)}\right)} \tag{11}$$

In which x_H denotes outer layer output and prediction function is calculated as

$$f(x) = \underset{1 \le j \le K}{\arg\max}\, P(y = j \mid x) \tag{12}$$

Here DBN parameters $\{\theta_1, \theta_2, ..., \theta_H, \theta_s\}$ are highly optimized with regard to negative log-likelihood loss over the training set.

2.4. Outlay-Aware Deep Belief Network

The total outlay over the training set is reduced by using the outlay-aware learning method. Assume the number of classes as K, data sample as x, and the outlay of misclassified data sample x as $C_{i,j}$. Also $C_{i,j}$ =, when the value of i is equal to j, indicating that the outlay for accurate classification is 0. In case of misclassification outlays, the record is classified into the class having minimum assumed outlay. The minimum assumed outlay $R(i \mid x)$ for classifying input vector x into class i is depicted as

$$R(i \mid x) = \sum_{j=1,\, j \ne i}^{K} P(j \mid x) C_{i,j} \tag{13}$$

In which $P(j \mid x)$ denotes the probability estimation of classification of record into class j. The overall risk R is calculated by a decision rule using the probability $P(x_n)$ as,

$$R = \sum_{n=1}^{N} \sum_{i=1}^{K} R(i \mid x_n) P(x_n) \tag{14}$$

An absolute classifying unit can predict the exact solution by evaluating the risk of classifying input to every class and can identify the label reaching the minimum expectation risk on the basis of Bayes decision theory. The misclassification gives the retribution for classification errors outlays so

ISSN: 2237-0722
Vol. 11 No. 2 (2021)
Received: 22.03.2021 – Accepted: 25.04.2021

that the outlays should not be negative. The probability of sample data $x \in S$ belongs to class j, and the stochastic variable y can be measured as below,

$$P(y = j \mid x) = soft\max{}_j (b + Wx) \\ = \frac{\exp(b_j + W_j x)}{\sum_i \exp(b_i + W_i x)} \qquad (15)$$

The misclassification threshold values are utilized for converting the posterior probabilities as class labels so as to reduce the misclassification outlays. The new probability $P\xi$ is measured by executing misclassification threshold values $1 - C_{i,j}$ on the acquired probability $P(y = j \mid x)$.

$$P\xi(y = j \mid x) = P(y = j \mid x) \cdot (1 - C_{i,j}) \qquad (16)$$

The hypothesized prediction of the sample is measured using the equation,

$f(x) = \arg\max_j P\xi(y = j \mid x)$ … (17), here x refers maximum probability

The prior probability of various classes is significantly imbalanced, and the misclassification outlay is established at the outer layer so as to show the imbalanced classes. Additionally, the existing training algorithms will assume uniform class distribution with the same outlay of misclassification like $\forall i, j \in [1, 2, ..., K]$, if $i = j, C_{i,j} = 0$, if $i \neq j, C_{i,j} = 1$ which is not possible in lives applications.

2.5. Evolutionary Outlay-Aware Deep Belief Network

The EA algorithm is proposed to optimize the outlays of misclassification. An EOA-DBNC is designed by combining outlay function into classification pattern with misclassification classes. Initially, an individual of misclassification outlay is initialized randomly. A DBN is provided with training data set. The training error is calculated based on the specific outlay-aware hypothesized prediction performance once the outlays are related to the output of DBN. The individual of the next generation is created by choosing the outlays of misclassification by the calculation performed on the training set. The accurate selected outlays are acquired and related with the output layer of DBN for forming EOA-DBNC in Fig.1. The test data outputted from the EOA-DBNC process is evaluated for finding the performance of the method. The steps of the EOA-DBNC algorithm is discussed below,

Training Process of EOA-DBNC

Pre-Training Phase:

1. Assume S_t as the training set.
2. Run EOA-DBNC algorithm.

Enhancing Phase:

1. Set up a population of misclassification outlays randomly.
2. Create a new population using mutation and crossover processes on the basis of the differential operator.
3. Add the corresponding outlays of misclassification with the training data set so as to calculate the error on training data.
4. Choose appropriate outlays and remove the inappropriate outlays, based on evaluation performance.
5. Repeat the mutation and crossover process continuously so as to choose the appropriate outlays

2.5.1. Chromosome Encoding

The chromosome encoding process is performed in EAs for efficiently expressing the significant variables for attaining improved performance. Mostly, the misclassification outlays are generally not known. In this new method, the outlays are acquired by allowing each chromosome to express the outlays of misclassification for various classes. Therefore, the appropriate chromosomes are selected as outlays of misclassification using the EOA-DBNC method. In Fig.2 the chromosome encoding and evolution processes are depicted.

2.5.2. Population Initialization

The primary population is acquired through a uniform random sampling process of each variable within the specific range. The evolution unit is formed by choosing the appropriate outlays of misclassification. The evolution unit of misclassification outlays is formed as a continuous process for every population.

2.5.3. Adaptive differential Evolution (DE) Operation

The adaptive DE process is carried out after the initialization process by three processes: crossover, mutation, and selection. The steps involved in algorithm 1 are explained as follows: The mutation is performed with DE mutation technique for generating mutated individuals on the basis of the current population. A binomial crossover is performed after the mutation process for creating offspring individuals. Each and every individual in DE comprises corresponding crossover probability rather than a constant value. A selection process is performed for selecting the correct one among the parent and offspring individuals on the basis of the obtained fitness values. The control parameters are mechanically upgraded to proper values by carrying out parameter adaptation in each generation of individuals. The crossover probability is created independently for each individual using normal distribution having to mean as μC_r and standard deviation as 0.1. Likewise, the mutation factor is created independently for each individual using Cauchy distribution having location parameter as μ_F and scale parameter as 0.1. Finally, the mean μC_r and location parameter μ_F are updated. The encoding of the chromosome is carried out with misclassification outlays as numerical style. The population is continuously evolved throughout each population's evolution process by carrying out the processes like mutation, crossover, evaluation, and selection.

2.5.4. Fitness Evaluation

The fitness evaluation is performed for selecting the proper misclassification outlays. In this method, the individual chromosome is established into individual DBN in the form of outlays. The appropriate outlays are created by utilizing the mean value of the training set, which is the important factor for the optimization process.

2.5.5. Termination Condition

The population creation enlarged, and convergence and diversity are prolonged within the population by formulating the EAs. The terminating condition of the algorithm is based on the number of generations. The fitness values obtained by this method are unchangeable for more than 25 generations of individuals. The algorithm is terminated after reaching the maximum number of generations or if the convergence condition is met.

2.5.6. EOA-DBNC Creation

Ultimately, the optimization process will terminate with an accurate individual utilized as outlays to enhance the capability of the EOA-DBNC method.

```
Algorithm - 2: Overall Working of EOA-DBNC
Provide Inputs like N: Size of population, X: Imbalanced Records, G: Maximum number of generations, Y: Class Labels, Cr: Crossover probability: F: Mutation Factor, Assign $\mu C_r = 0.5, \mu_F = 0.5, A = \phi, \beta = 0.5$, $range = (c_{\min}, c_{\max})$ : Range of chromosome values, D: Dimension of solution space
First Step - Initialization: Create an initial population $\{c_0^1, ..., c_0^N\}$ in the solution space by uniform sampling. The initial value of the ith individual is defined as $c_0^i = c_{\min} + rand(0,1) \cdot (c_{\max} - c_{\min})$. Calculate every candidate solution as $c_0^i (i = 1, ..., N)$ by the specific $DBN(x \in S_t, y \mid c)$ for acquiring a vector representing fitness function $F(c_0^i)$.
Second Step - Evolution:
for g = 1,..., G do
Configure all successful mutation factors $F_i$ at $S_F = \phi$;
Configure all successful crossover probabilities $Cr_i$ at $S_{Cr} = \phi$;
for $i = 1, ..., N_p$ do
Create $Cr_i = randn\ (\mu_{Cr}, 0.1)$, $F_i = rand\ c_i\ (\mu_F, 0.1)$
Mutation: Selecting the indices j and k, randomly and then form a new candidate solution $c_g^{i'}$ from $c_g^i$, $c_g^j$ and $c_g^k$ by $c_g^{i'} = c_g^i + F_i \cdot (c_g^j - c_g^k)$ which represents DE operator.
Create $i_{rand} = rand\,\mathrm{int}(1, D)$
Crossover:
if $i = i_{rand}$ or $rand\ (0,1) < Cr_i$ then $u_g^i = c_g^{i'}$ g
else $u_g^i = c_g^i$
end if
Selection:
if $F(c_g^i) \geq F(u_g^i)$ then $c_{g+1}^i = c_g^i$
else $c_{g+1}^i = u_g^i, c_g^i \to A, Cr_i \to S_{Cr}, F_i \to S_F$
end if
Extract solutions from A in a random manner so that $|A| \leq N$
Parameter Adaptation:
$\mu C_r = (1 - \beta) \cdot \mu C_r + \beta \cdot mean(S_{CR})$
$\mu_F = (1 - \beta) \cdot \mu_F + \beta \cdot mean(S_F)$
end
Third Step: EOA-DBNC Creation: Create an EOA-DBNC for acquiring the best individual $c_{best}$ from the training data set $S_t$ by misclassification value.
Fourth Step: Run-time Evaluation: Estimate EOA-DBNC using test data set $S_{test}$.
```

3. Performance Metrics

The accuracy of a classification and the time required for the classification process are performed in this stage.

Classification Accuracy: It is the rate of accurate classifications for the independent testing task, which is measured using the cross-validation method. It is calculated by finding the ratio of number of accurate predictions with number of inputted samples.

Classification Time: The total time required by the method is calculated for setting up the training model.

4. Results and Discussions

Table 1 - Dataset Name, No. of Instances and No. of Features

S.No.	Dataset Name	No. of Instances	No. of Features
1	Blood	748	5
2	Bupa	345	7
3	Car	1728	6
4	Contraceptive	1473	9
5	Credit	30000	24
6	Diagnostic	569	32
7	Ecoli	336	8
8	Ionosphere	351	34
9	Mammography	961	6
10	Monks - 1	432	7
11	Monks - 2	432	7
12	Monks - 3	432	7
13	Parkinsons	197	23
14	Pima	768	8
15	Prognostic	198	34
16	Sonar	208	60
17	Spect	267	22
18	Tic-Tac-Toe Endgame	958	9
19	Vert	310	6
20	Yeast	1484	8

Table 2 - Predictive Accuracy of the Algorithms

Dataset	**Algorithms**						
	TREE	**RBF**	**CBB**	**IGA - FLANN**	**ISVMC**	**APSO-CELMC**	**IGOA-EOA-DBNC**
Blood	0.78	0.78	0.78	0.83	0.89	0.93	0.99
Bupa	0.67	0.69	0.70	0.76	0.85	0.91	0.97
Car	0.96	0.97	0.96	1	1	1	1.00
Cont.	0.7	0.68	0.70	0.77	0.88	0.92	0.98
Credit	0.86	0.86	0.86	0.89	0.94	0.97	1.00
Diag.	0.95	0.98	0.95	0.99	1	1	1.00
Ecoli	0.99	0.98	0.99	0.99	1	1	1.00
Iono.	0.89	0.94	0.90	0.96	1	1	1.00
Mamm.	0.82	0.81	0.82	0.85	0.94	0.97	1.00
Mks-1	0.99	0.96	0.99	1	1	1	1.00
Mks-2	0.68	0.71	0.68	1	1	1	1.00
Mks-3	1	1	1	1	1	1	1.00
Park	0.9	0.9	0.92	0.97	1	1	1.00
Pima	0.75	0.77	0.75	0.81	0.93	0.96	1.00
Prog.	0.79	0.8	0.77	0.84	0.95	0.95	1.00
sonar	0.79	0.83	0.80	0.89	0.95	0.96	1.00
Spect	0.84	0.85	0.82	0.88	0.95	0.97	1.00
Tic	0.94	0.97	0.94	1	1	1	1.00
Vert	0.83	0.83	0.82	0.87	0.94	0.96	1.00
Yeast	0.66	0.64	0.68	0.68	0.81	0.86	0.97

Table 3 - Time Taken by the Algorithms for Classification (in milliseconds)

Dataset	**Algorithms**						
	TREE	**RBF**	**CBB**	**IGA - FLANN**	**ISVMC**	**APSO-CELMC**	**IGOA–EOA-DBNC**
Blood	2377	2839	2721	1032	649	772	646
Bupa	2056	2496	3193	1948	1123	929	780
Car	2157	2385	2795	1682	947	644	511
Cont.	2978	2635	2696	1732	902	615	472
Credit	19563	19674	19701	15782	8099	6580	6409
Diag.	7233	7676	7790	3982	2184	1608	1464
ecoli	2367	2665	2686	1846	799	676	500
Iono.	7558	7569	7663	3901	2001	1584	1443
Mamm.	2210	2612	2672	1738	864	830	712
Mks-1	2185	2076	2662	1936	895	692	549
Mks-2	2940	2120	2757	1726	826	748	611
Mks-3	2062	2497	2756	1639	904	705	578
Park	4526	4879	4731	2393	1275	906	746
Pima	2813	2992	2797	1392	780	535	409
Prog.	2208	2644	2656	1888	947	674	540
Sonar	8297	8788	8738	5189	2548	1723	1554
Spect	5655	5455	5804	2291	1277	1066	900
Tic	2574	2745	2807	1749	985	534	383
Vert	2054	2645	2618	1638	849	671	503
Yeast	2971	2222	2655	1843	909	576	429

Fig. 1 - Predictive Accuracy Comparison for the Datasets 1 to 5

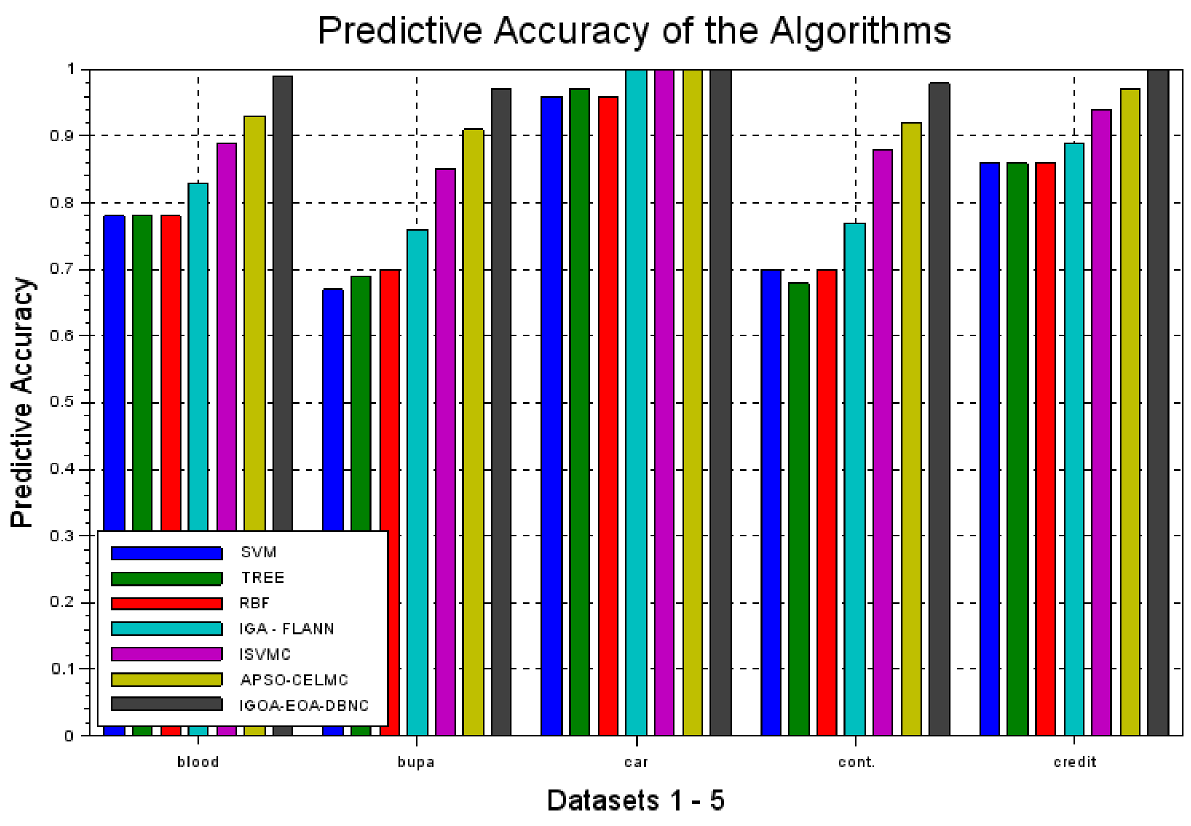


Fig. 2 - Predictive Accuracy Comparison for the Datasets 6 to 10

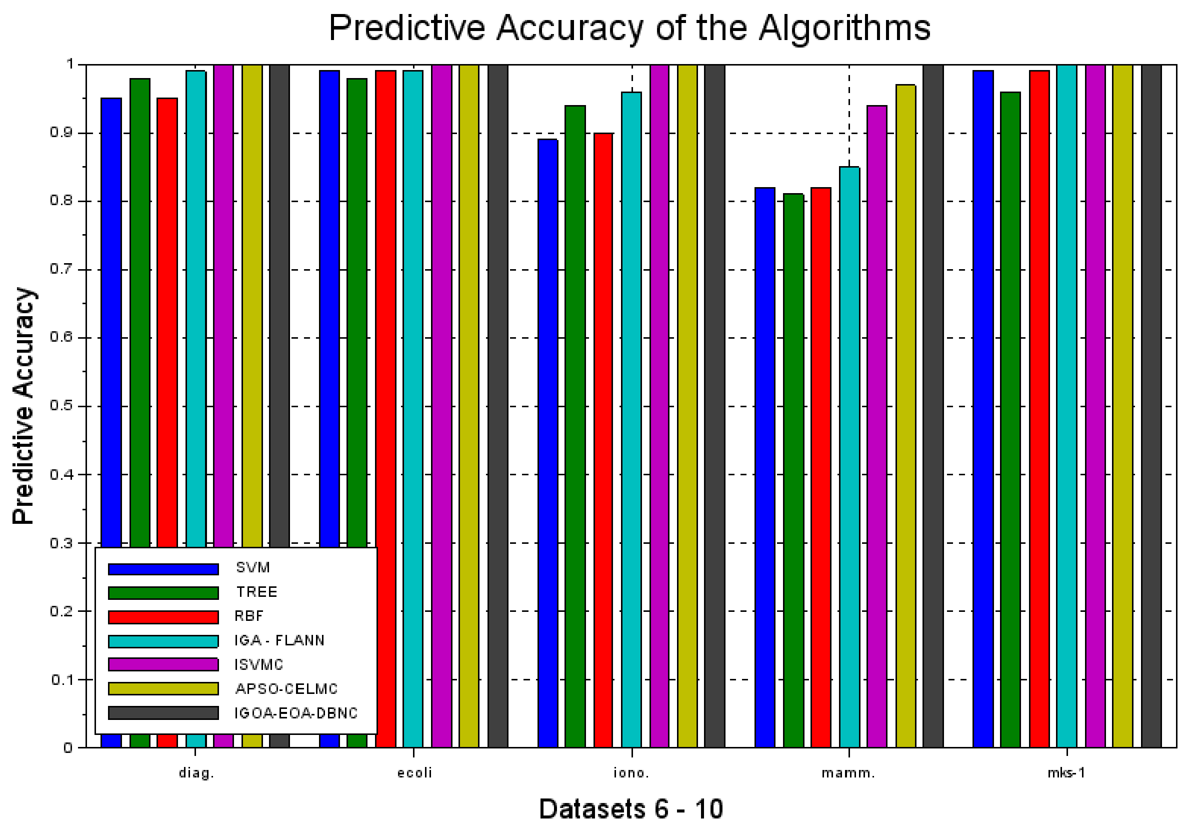


Fig.3 - Predictive Accuracy Comparison for the Datasets 11 to 15

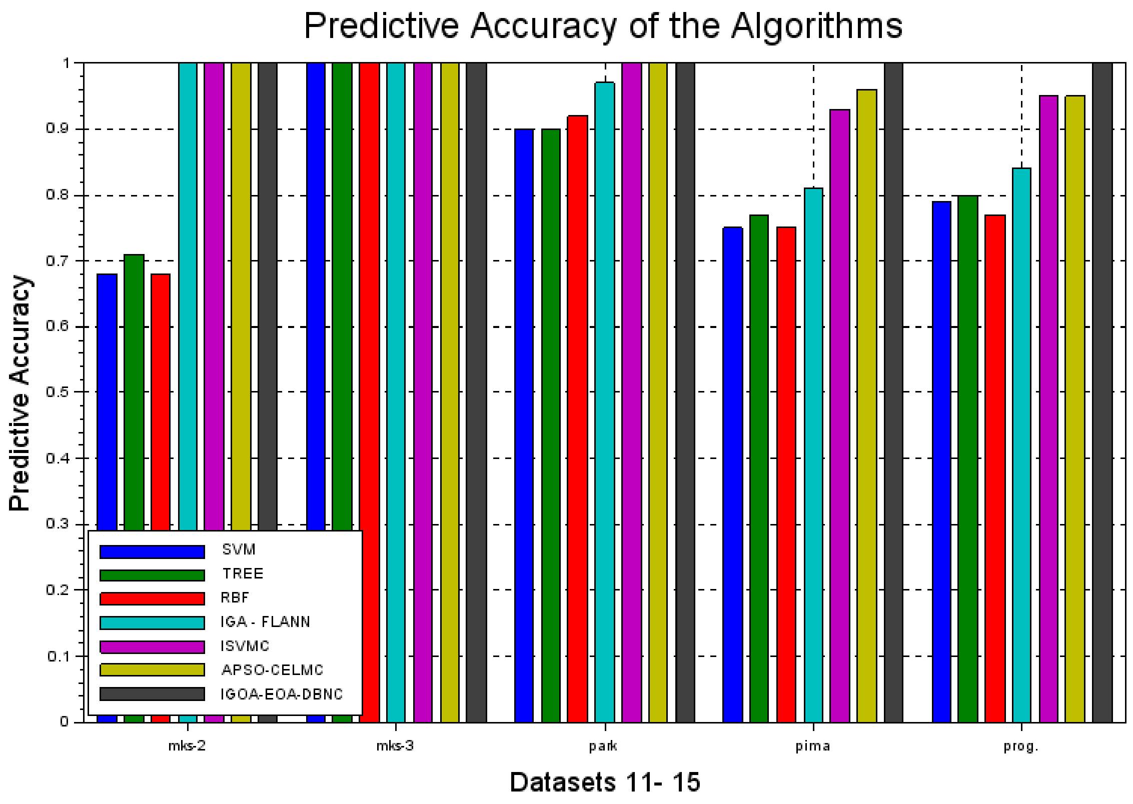


Fig. 4 - Predictive Accuracy Comparison for the Datasets 16 to 20

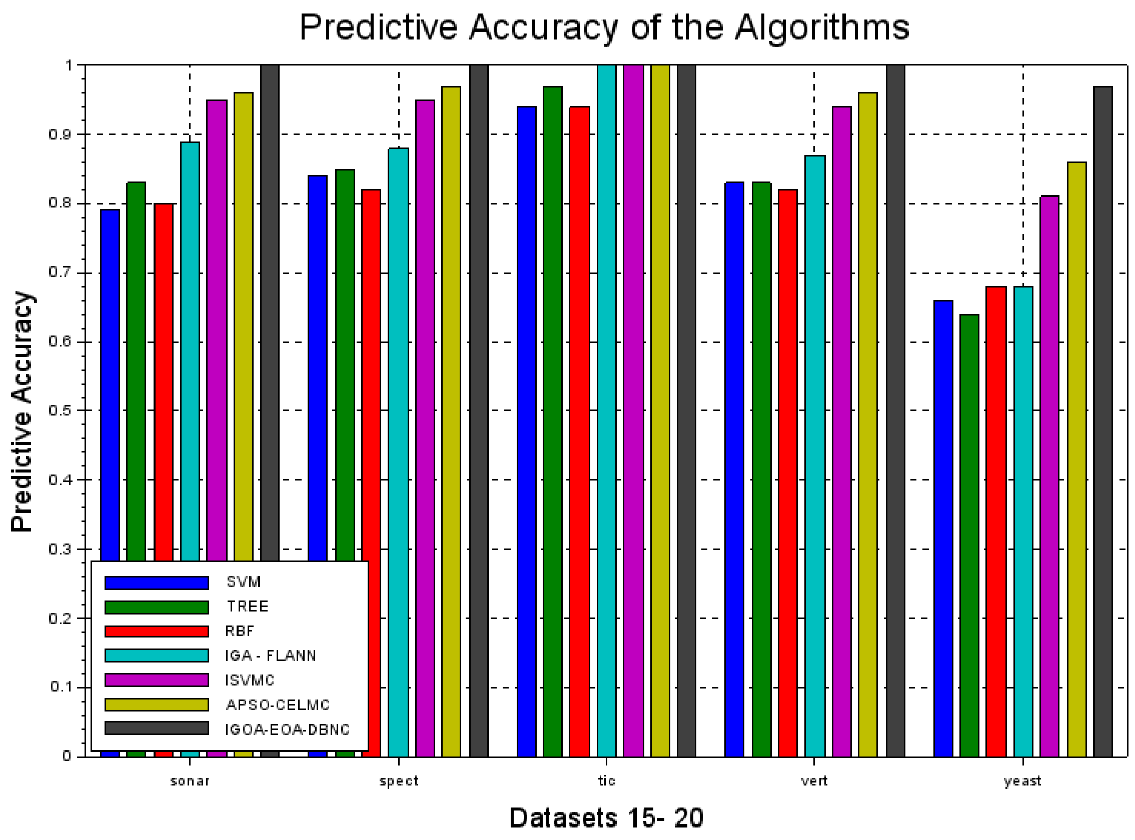


Fig. 5 - Time Taken for Classification by the Algorithms Comparison for the Datasets 1 to 10

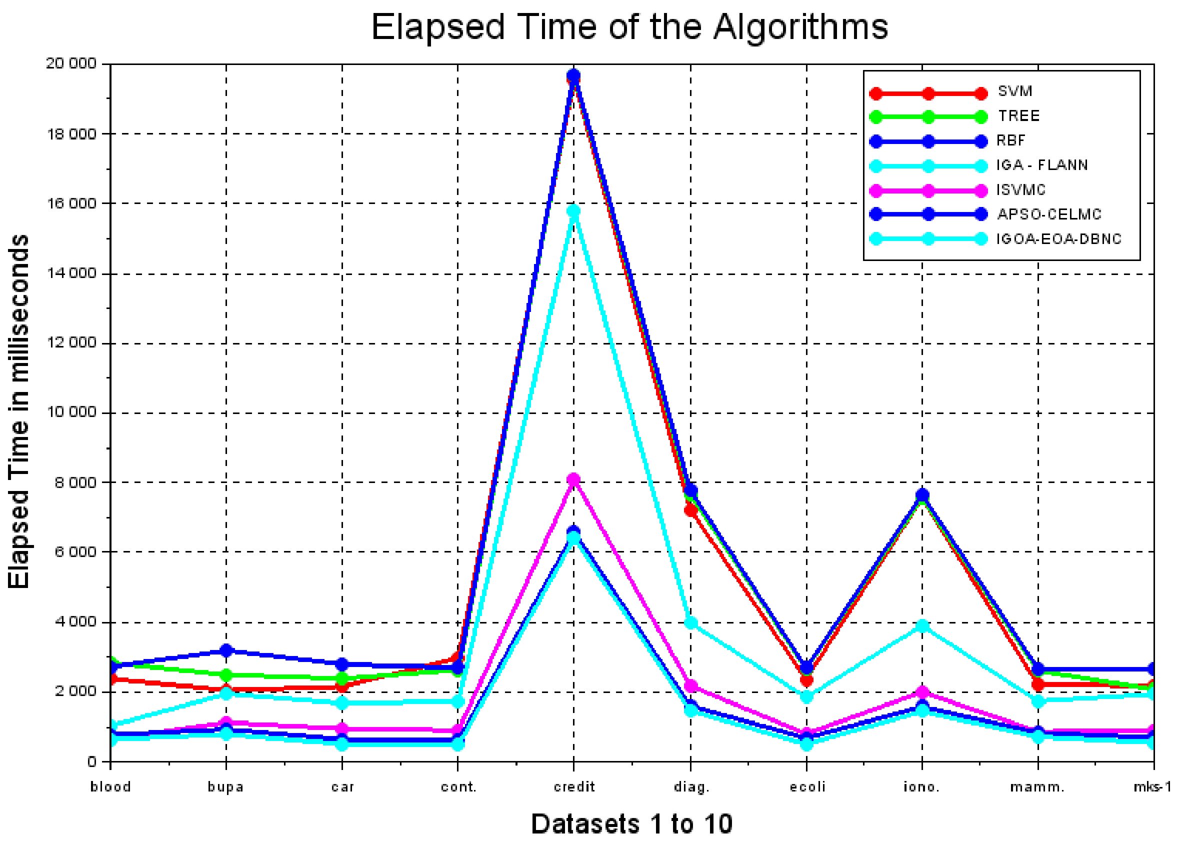
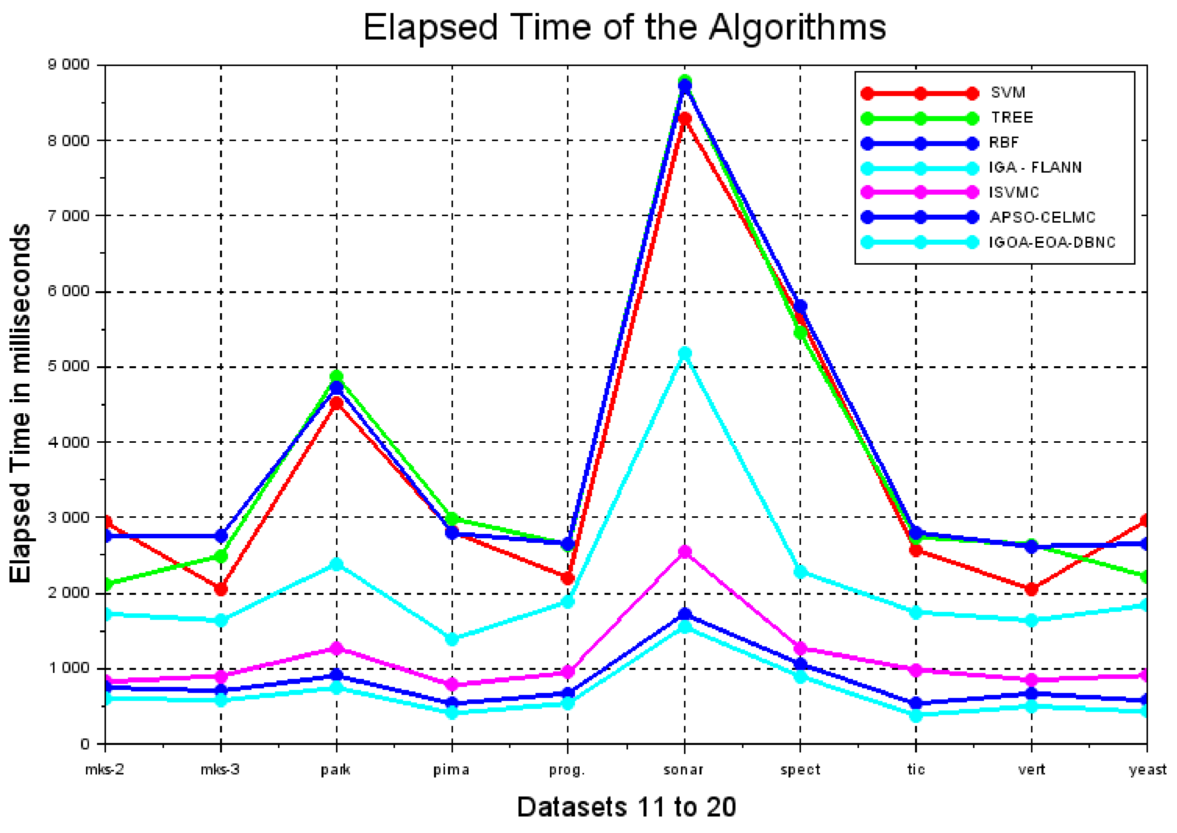


Fig. 6 - Time Taken for Classification Algorithms in Comparison with Datasets 11 to 20



Twenty different data sets are extracted from UCI machine learning archives such as monks-1, monks-2, monks-3, Bupa, car, blood, credit, ecoli, diagnostic, mammography, ionosphere, Pima, Parkinson's, prognostic, spect, sonar, yeast, Tic-Tac-Toe Endgame and vert. The number of instances, the name of the dataset, and a number of features are depicted in Table 1. One MATLAB tool, Core i3 processor with 8 GB RAM and 1 TB hard disk that operates in Microsoft Windows 8 OS are used for implementations. The high prediction accuracy and low classification time are ensured in the newly designed IGOA-EOA-DBNC method when compared with existing IGA-FLANN [13], ISVMC [14] and APSO-CELMC [15].

5. Conclusion

The feature selection process ensures improved accuracy in classification and low measuring time for carrying out data mining on high dimensional data. This research method uses a better form of grasshopper optimization algorithm for the classification of high dimensional data. Twenty datasets are utilized for calculating the efficiency of the data classifier such that 99.98% of accurate classification outputs can be attained in less time consumption.

References

Polat, D., & Çataltepe, Z. (2012). Feature selection and classification on brain computer interface (BCI) data. *In 20th Signal Processing and Communications Applications Conference (SIU)*, 1-4.

Zhang, G.P. (2000). Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 30*(4), 451-462.

Lewis, D.D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. *In European conference on machine learning, Springer, Berlin, Heidelberg*, 4-15.

Lin, K.C., Zhang, K.Y., Huang, Y.H., Hung, J.C., & Yen, N. (2016). Feature selection based on an improved cat swarm optimization algorithm for big data classification. *The Journal of Supercomputing*, *72*(8), 3210-3221.

Lin, K.C., Huang, Y.H., Hung, J.C., & Lin, Y.T. (2015). Feature selection and parameter optimization of support vector machines based on modified cat swarm optimization. *International Journal of Distributed Sensor Networks*, *11*(7), 365869.

Lin, K.C., Chen, S.Y., & Hung, J.C. (2015). Feature selection and parameter optimization of support vector machines based on modified artificial fish swarm algorithms. *Mathematical problems in engineering*, *2015*.

Pena-Reyes, C.A., & Sipper, M. (2000). Evolutionary computation in medicine: an overview. *Artificial Intelligence in Medicine*, *19*(1), 1-23.

Cha, S.H., & Tappert, C.C. (2009). A genetic algorithm for constructing compact binary decision trees. *Journal of pattern recognition research*, *4*(1), 1-13.

Kennedy, J. (2010). Particle swarm optimization, in: Encyclopedia of Machine Learning, Springer, US, 760–766.

Brahma, P.P., Wu, D., & She, Y. (2015). Why deep learning works: A manifold disentanglement perspective. *IEEE transactions on neural networks and learning systems*, *27*(10), 1997-2008.

Deng, L., & Yu, D. (2014). Deep learning: methods and applications. *Foundations and trends in signal processing*, *7*(3–4), 197-387.

Salakhutdinov, R., & Hinton, G. (2012). An efficient learning procedure for deep Boltzmann machines. *Neural computation*, *24*(8), 1967-2006.

Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: theory and application. *Advances in Engineering Software*, *105*, 30-47.