symmetry

MDPI

Article

Genetic Algorithm Based on Natural Selection Theory for Optimization Problems

Musatafa Abbas Albadr [1,*], Sabrina Tiun [1], Masri Ayob [1] and Fahad AL-Dhief [2]

[1] CAIT, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi 43600, Malaysia; sabrinatiun@ukm.edu.my (S.T.); masri@ukm.edu.my (M.A.)
[2] School of Electrical Engineering, Department of Communication Engineering, Universiti Teknologi Malaysia, UTM Johor Bahru, Johor 81310, Malaysia; fahadtaha37@yahoo.com
* Correspondence: mustafa_abbas1988@yahoo.com; Tel.: +60-104213847

Received: 29 August 2020; Accepted: 5 October 2020; Published: 23 October 2020

Abstract: The metaheuristic genetic algorithm (GA) is based on the natural selection process that falls under the umbrella category of evolutionary algorithms (EA). Genetic algorithms are typically utilized for generating high-quality solutions for search and optimization problems by depending on bio-oriented operators such as selection, crossover, and mutation. However, the GA still suffers from some downsides and needs to be improved so as to attain greater control of exploitation and exploration concerning creating a new population and randomness involvement happening in the population at the solution initialization. Furthermore, the mutation is imposed upon the new chromosomes and hence prevents the achievement of an optimal solution. Therefore, this study presents a new GA that is centered on the natural selection theory and it aims to improve the control of exploitation and exploration. The proposed algorithm is called genetic algorithm based on natural selection theory (GABONST). Two assessments of the GABONST are carried out via (i) application of fifteen renowned benchmark test functions and the comparison of the results with the conventional GA, enhanced ameliorated teaching learning-based optimization (EATLBO), Bat and Bee algorithms. (ii) Apply the GABONST in language identification (LID) through integrating the GABONST with extreme learning machine (ELM) and named (GABONST-ELM). The ELM is considered as one of the most useful learning models for carrying out classifications and regression analysis. The generation of results is carried out grounded upon the LID dataset, which is derived from eight separate languages. The GABONST algorithm has the capability of producing good quality solutions and it also has better control of the exploitation and exploration as compared to the conventional GA, EATLBO, Bat, and Bee algorithms in terms of the statistical assessment. Additionally, the obtained results indicate that (GABONST-ELM)-LID has an effective performance with accuracy reaching up to 99.38%.

Keywords: metaheuristic; optimization; genetic algorithm based on natural selection theory

1. Introduction

The past few decades have witnessed an increasing interest in using nature-inspired algorithms to solve numerous optimization problems including timetabling problems [1–4]; data mining [5–7]; breast cancer diagnosis [8]; load balancing of tasks in cloud computing [9]; language identification [10,11]; and vehicle routing problems [12–14]. The observation of processes found in nature became the basis for nature-inspired algorithms of which the main objective is to seek the global optimal solutions for certain problems [15]. There are two common key factors in nature-inspired algorithms, namely diversification (exploration) and intensification (exploitation). Exploration entails the search for global optima via the random exploration of new solution spaces; meanwhile, exploitation entails the search for local optima in solution spaces that have been previously

explored [15]. Optimal solution is not found via intense exploration, while algorithm is trapped in local optima due to deep exploitation [15]. Hence, a balance between both factors is pertinent for any nature-inspired algorithm [15,16].

Numerous nature-inspired algorithms have been suggested in literature such as particle swarm optimization [17]; genetic algorithm [18]; bat algorithm [19]; harmony search [20]; and kidney-inspired algorithm [21]. Physical/biological activities found in the natural world form the basis for nature-inspired algorithms in the quest to find solutions for various optimization problems [21], which then render these algorithms to be effective optimization algorithms. Nevertheless, shortcomings including finding the balance between the key factors of exploration and exploitation could still be prevalent, which affects the algorithm's efficacy [15,21].

Among these metaheuristics, the genetic algorithm (GA) is considered as one of the most popular metaheuristic algorithms that has been used in practically all optimization, design and application areas [11]. GA is actually one of the earliest proposed population-based stochastic algorithms. GA consists of the selection, crossover, and mutation operations, similar to other evolutionary algorithms (EAs). Darwin's theory of evolution [22–24], i.e., the survival of the fittest, makes up the basis of GA whereby the fittest genes are simulated. Its algorithm is population-based, and each solution matches a chromosome whilst each parameter denotes a gene. A fitness (objective) function is used to evaluate the fitness of each individual in the population. For improving poor solutions, the best solutions are chosen randomly with a selection (e.g., random, k-tournament or roulette wheel) mechanism. This operator is more likely to choose the best solutions since the probability is proportional to the fitness (objective value). What increases local optima avoidance is the probability of choosing poor solutions. This means that if good solutions are trapped in a local solution, they can be pulled out with other solutions.

Despite its proven capacity in resolving numerous search and optimization problems [25], GA still has several weaknesses. For example, in GA, the mechanism of both 1: the new population is generated from the existing parents and 2: the randomness involvement that happened in the population at the time of solution initialization may both lead to making the GA fail to create exploration in the solution search space. In addition, the mutation is applied to the new chromosomes (offspring) that are obtained from the crossover operation and then moved to the next generation (using same pool) that might lead to not providing a good enough exploration and solution diversity. In other words, the crossover followed by mutation may prevent the mutation and crossover operations from guaranteeing an optimal solution [26–28]. Hence, this current study proposes a new GA based on the theory of natural selection to improve the GA performance. The proposed genetic algorithm based on natural selection theory (GABONST) has been evaluated based on the following: (**i**) fifteen renowned benchmark test functions (**ii**) apply the GABONST in language identification (LID) through integrating the GABONST with extreme learning machine (ELM) (GABONST-ELM) and compare the proposed GABONST-ELM with enhanced self-adjusting extreme learning machine (ESA-ELM) algorithm. The ESA-ELM is based on enhanced ameliorated teaching learning-based optimization (EATLBO) [29], where EATLBO still suffers from some downsides in terms of the selection criteria and the ability to create good fresh solutions. The explanation of the new proposed GABONST is provided in Section 2.2.

A reminder that this paper is organized as such: Section 2 presents the proposed method. Section 3 details the experiments carried out and their respective findings, and finally Section 4 presents the general conclusions and suggestions for future research.

2. Materials and Methods

2.1. Genetic Algorithm

The concept of GA is aimed to imitate the natural changes that occur in living ecosystems, which is social systems, evaluate the psychological consequences, and model the variable methods. Holland [30] pointed out that Goldberg [22] had significantly contributed to the widespread usage of

GA by demonstrating that a large number of problems can basically be solved by utilizing the GA methods. According to [31,32], GA is a widely popular search and optimization method for resolving highlyintricate problems. The success of its methods has been proven in areas involving machine learning approaches. A complete description of the actual coded GA is provided in this section. Figure 1 provides the flowchart of the standard GA.

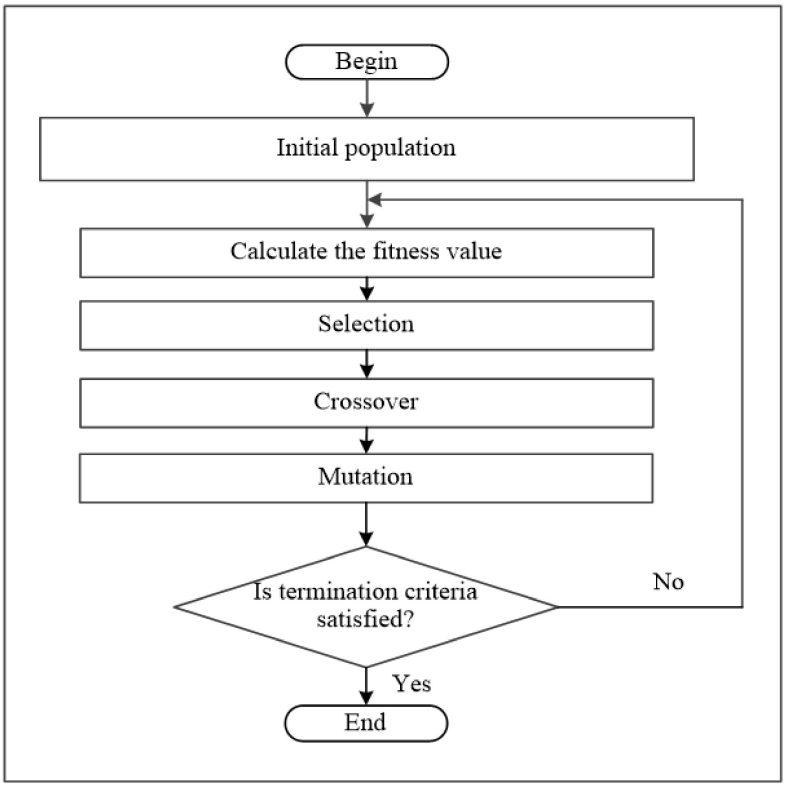

Figure 1. Flowchart of the standard genetic algorithm (GA) [33].

Below is the description of the standard GA procedure [33]:

Initial population. This entails the possible solution for set P, i.e., a series of random generations of real values, $P = \{p_1, p_2, \ldots, p_s\}$.

Evaluation (calculate the fitness value). The fitness function must be delineated in order to evaluate each chromosome in the population, characterized as fitness = g(P).

Selection. Following the fitness value calculation, the chromosomes are arranged by their fitness values. The selection of parents is then conducted entailing two parents for the crossover and the mutation.

Genetic operators. Once the selection process is complete, the parents' new chromosomes or the offspring (C_1, C_2) are created by utilizing the genetic operators. The new chromosomes (C_1, C_2) are then saved into children population C. This process involves the crossover and mutation operations [34]. The crossover operation is applied to exchange information between two parents, which were selected earlier. Several methods of crossover operators are available such as single-point, two-point, k-point crossover, arithmetical crossover... etc. While in the mutation operation, the genes of the crossed offspring's chromosomes are changed. Likewise, several methods are available for the mutation operator.

Upon completion of the selection, crossover and mutation operations, children population C is completely generated and will be transferred to the subsequent population (P). P is then utilized in the next iteration, whereby the whole process is run again. The iterations will stop if there is convergence of results, or if the number of iterations goes beyond the maximum threshold.

Recently, GA has been broadly used in machine learning, adaptive control, combinatorial optimization, and signal processing areas. GA has a good capability of global search and also it is considered as one of the essential technologies that are associated with the modern intelligent calculation [35]. Additionally, the GA has been implemented in many applications such as face recognition, where GA has been applied to optimize the feature search process [36]. In addition, GA has used for task scheduling to solve the problem of task scheduling for phased array radar (PAR) [37]. In addition, GA has been used in image encryption for analyzing the image encryption security [38]. The GA has also been used in healthcare facilities for formulating an efficient prediction model of stochastic deterioration that combines the latest observed condition in the forecasting process for overcoming uncertainties and the subjectivity, which are related to the currently used methods [39]. Additionally, the GA has been applied in the fuzzy logic in order to find the association rules of the fuzzy logic [40]. The authors in [41] have combined GA with hesitant intuitionistic fuzzy sets to obtain the optimal solution for the decision making. The work in [42] presents an efficient GA-based content distribution scheme to reduce the transmission delay and also to improve the throughput of fog radio access network (F-RAN). The GA was used in [43] to enhance the performance of ELM algorithm by selecting the optimal input weights and applying it in breast cancer detection. Moreover, there are many optimizations and improvements have been done on the GA. For example, in [44–46] they present hybrid GAs, [47,48] propose enhancements on the GA in terms of operations (i.e., selection, mutation, and crossover), and the study in [11] worked on separating the population pools (crossover and mutation pools).

2.2. Genetic Algorithm Based on Natural Selection Theory (GABONST)

The GABONST was created based on the concept of natural selection theory. Natural selection is a biological theory was first proposed by Charles Darwin [49]. The natural selection theory entails the idea that genes adjust and survive throughout generations with the help of several factors. In other words, the organism with high ability is qualified to survive in the current environment and generates new organisms into the new generation. Whilst the organism with a low ability has two chances to survive in the current environment and avoid extinction: 1: the first chance is getting married to a well-qualified organism (an organism with high ability), which may lead to generating a new high ability offspring into the new generation, and 2: the second chance is the genetic mutation, which might lead to making the organism stronger and able to survive in the current environment. In case the organism, which is obtained from one of the two chances, does not satisfy the environment's requirements, the organism may become extinct over the time. However, it is a mutual impact where the environment affects the organisms and at the same time, the organisms affect the environment. Therefore, over time both the organisms and the environment will obtain changes [50]. Thus, applying the idea of the natural selection theory into GA promises to improve the exploration, exploitation and the solution's diversity of the conventional GA by controlling the search space based on both the organisms and the environment.

This study is simulating the idea of natural selection theory and integrating it into the genetic algorithm. The new proposed algorithm is named as a genetic algorithm based on natural selection theory (GABONST). The procedure of the GABONST is presented in the following steps:

1. Beginning of the algorithm.
2. Set number of population n and number of iteration NumIter.
3. Generate the population (chromosomes (S)) randomly; where $S = \{s_1, s_2, \ldots, s_n\}$.
4. Calculate the fitness value of each chromosome in the population g(S).
5. Calculate the mean of the fitness values using Equation (1).

$$\text{Mean} = \frac{\sum_{i=1}^{n} g(s_i)}{n} \quad (1)$$

In the GABONST, (1) the mean of the fitness values simulates the environment in biological theory. (2) The solution is simulating the organism in biological theory while the fitness value of the solution ($g(s_i)$) is simulating the ability of that organism to survive or not in that environment.

6. Compare the fitness value of each chromosome $g(s_i)$ with the mean:

   a. If $g(s_i)$ is less or equal to the mean then implement the mutation operation on the s_i and move to the next generation. This represents the right side of the GABONST flowchart (see Figure 2), where the right side simulates the well-qualified organisms (chromosomes) to survive the current environment.

   b. Otherwise, the chromosome s_i will get two chances to be improved, this represents the left side of the GABONST flowchart (see Figure 2), where the left side simulates the idea of giving the unqualified organisms (chromosomes) two chances to adjust their genes and be qualified to survive the current environment:

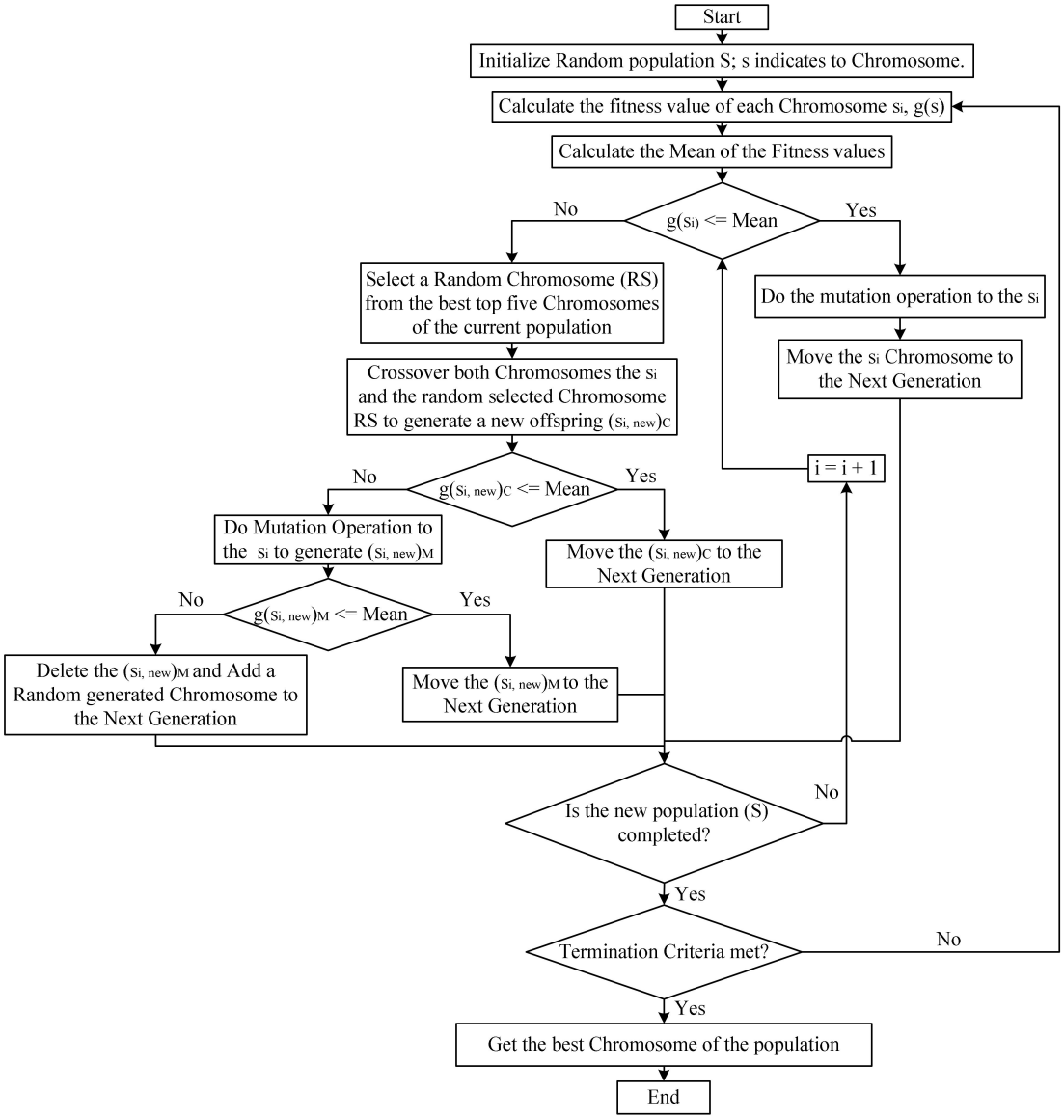

Figure 2. Flowchart of the genetic algorithm based on natural selection theory (GABONST).

i. The first chance is through getting married to a well-qualified organism (crossover the weak chromosome (s_i) with a well-qualified chromosome (RS)). If the new chromosome $(s_{i,\ new})_C$, which is obtained by crossover s_i and RS, qualifies to survive the current environment ($g(s_{i,\ new})_C$ less or equal to the mean) then the $(s_{i,\ new})_C$ move to the next generation. Otherwise, go to the second chance, step (ii).

The crossover operation is subject to the boundaries (upper bounds and lower bounds). In case the value of the gene has gone beyond the max (upper bound), then we make it equal to the max (upper bound). While in the case that the value of the gene has gone lower than the min (lower bound), then we make it equal to the min (lower bound).

ii. The second chance is through the genetic mutation (implement the mutation operation to the weak chromosome (s_i)). If the new chromosome $(s_{i,\ new})_M$, which is obtained by applying the mutation operation on s_i, qualifies to survive the current environment ($g(s_{i,\ new})_M$ less or equal to the mean) then the $(s_{i,\ new})_M$ move to the next generation. Otherwise, in the case that the organism (chromosome (s_i)) has missed both of the chances to be qualified to survive in the current environment then that organism will die (that chromosome (s_i) will be deleted) and a new one comes to life (add a random generated chromosome to the next generation). Figure 3 provides an example of the arithmetic crossover and uniform mutation operations that have been applied in GABONST.

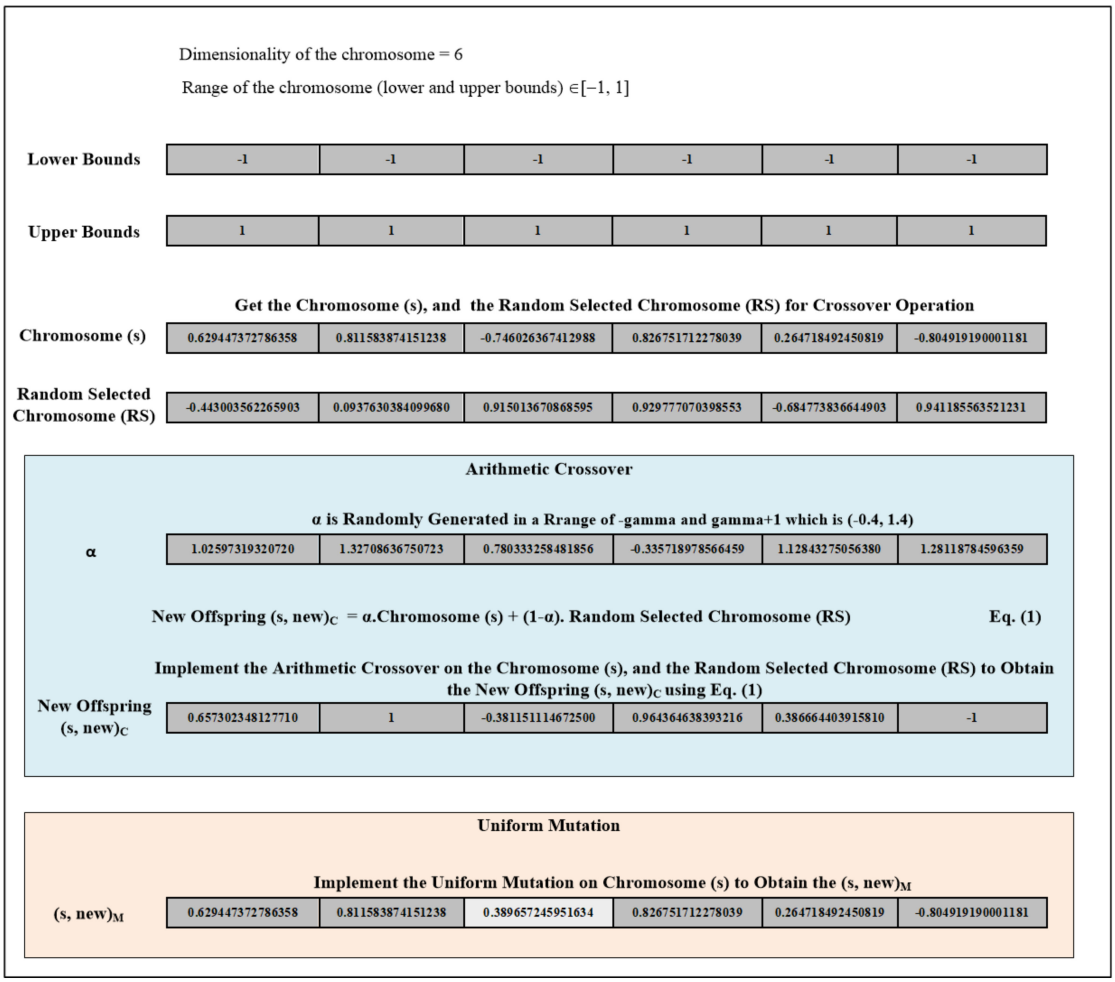

Figure 3. Diagram of the arithmetic crossover and uniform mutation operations example.

Following that, the generation of the new population (S) will be obtained. S is then utilized in the ensuing iteration, whereby the whole process is run again. The process iterations will stop if there is convergence of results or if the number of iterations goes beyond the maximum threshold. The GABONST procedure is illustrated in Algorithm 1.

Algorithm 1 GABONST

Begin
Set number of population n and number of iteration NumIter.
Initial population. The initial population is a possible chromosome set S, which is a set of real values generated randomly, $S = \{s_1, s_2, \ldots, s_n\}$.
Evaluation. A fitness function should be defined to evaluate each chromosome in the population and can be written as fitness = g(S).
Calculate the Mean of the fitness values based on Equation (1).
Iter = 1;
While (Iter < NumIter)
i = 1;
While (i < n)
Compare the g(s_i) with Mean:
If (g(s_i) <= Mean)
  Implement mutation operation to s_i and move it to the next generation.
Else
  Select a random chromosome from the top five chromosomes (RS) of the current population and implement the crossover operation on both s_i and RS to generate $(s_{i,\ new})_C$; g$(s_{i,\ new})_C$.
    If (g$(s_{i,\ new})_C$ <= Mean)
      Move the $(s_{i,\ new})_C$ to the next generation.
    Else
      Implement mutation operation to s_i and generate $(s_{i,\ new})_M$; g$(s_{i,\ new})_M$.
        If (g$(s_{i,\ new})_M$ <= Mean)
          Move the $(s_{i,\ new})_M$ to the next generation.
        Else
          Delete s_i and add a random generated chromosome to the next generation.
        End if
    End if
End if
i = i + 1;
End while
Calculate the fitness value of each chromosome in the population g(S).
Calculate the Mean of the fitness values based on Equation (1).
Iter = Iter + 1;
End while
End

3. Results

3.1. Experimental Test One

The measures for evaluating the GABONST is discussed in this section, which compares the GABONST with EATLBO, conventional GA, Bat and Bee algorithms, in terms of certain standard mathematical functions associated with the optimization surface. These algorithms underwent fifteen experiments that applied fifteen distinct objective functions, with 100 iterations and a population size of 50. The most common fifteen objective functions [51] were used to assess the performance of the

optimal solution value selection in all the iterations for these algorithms. The optimal solution and dimension for each of the mathematical objective functions (F1–F15) are presented in Table 1 below. Figure 4 depicts the graphical representation of mathematical objective functions.

Table 1. Details of the utilized mathematical objective functions.

Objective Function	Dim	Range	Optimal Solution
$f_1(x) = -\frac{1}{d}\sum_{i=1}^{d} \sin^6(5\pi x_i)$	10	[−1, 1]	−1
$f_2(x) = -\sum_{i=1}^{d} \sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$	2	[0, π]	−1.8013
$f_3(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	2	[−10, 10]	0
$f_4(x) = \frac{1}{2}\sum_{i=1}^{d}\left(x_i^4 - 16x_i^2 + 5x_i\right)$	10	[−5, 5]	−391.6599
$f_5(x) = \prod_{i=1}^{d} \sqrt{x_i}\sin(x_i)$	2	[0, 10]	−6.1295
$f_6(x) = \sum_{i=1}^{n} x_i^2$	256	[−5.12, 5.12]	0
$f_7(x) = \sum_{i=1}^{n-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	30	[−30, 30]	0
$f_8(x) = \sum_{i=1}^{n} ix_i^4 + \text{random}(0,1)$	30	[−1.28, 1.28]	0
$f_9(x) = \sum_{i=1}^{n}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30	[−5.12, 5.12]	0
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	128	[−32.768, 32.768]	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[−600, 600]	0
$f_{12}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)]$	2	[−2, 2]	3
$f_{13}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1\left(b_i^2 + b_ix_2\right)}{b_i^2 + b_ix_3 + x_4}\right]^2$	4	[−5, 5]	0.00030
$f_{14}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[−5, 5]	−1.0316
$f_{15}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[−5, 5]	0.398

Table 2 presents the statistical results of the fifteen mathematical objective functions of GABONST, GA, EATLBO, Bat and Bee algorithms following the programme's 50 runs. The most common three statistical evaluation measures were used in this study: root mean square error (RMSE), mean, and standard deviation (STD) [21,52]. The 50 results of each objective function were used to calculate the RMSE, mean, and STD. In Table 2, the values of the GABONST-RMSE and GABONST-STD are lower, which proves the effectiveness of the GABONST in terms of achieving the optimal solution. Meanwhile, the mean is close to the optimal solution (see Tables 1 and 2), which means that the GABONST had generally attained an optimal solution in the fifteen mathematical objective functions throughout the 50 rounds, indicating that the GABONST had performed better in comparison to EATLBO, GA, Bat and Bee algorithms in terms of effectiveness and efficiency. In Table 2, the best results are shown in bold.

Based on the results in Table 2 comparing the GABONST, conventional GA, EATLBO, Bat and Bee algorithms; the GABONST have outperformed the conventional GA, EATLBO, Bat and Bee algorithms on most of the test objective functions. Except F11 and F14, where in F11 both GABONST and EATLBO have achieved the optimal solution while in F14 the conventional GA was slightly better than the GABONST (see Tables 1 and 2). This means the GABONST is concluded to have a better performance than conventional GA, EATLBO, Bat and Bee algorithms. GABONST in this comparison is based on

the idea of natural selection theory, which aims to enhance the exploration, exploitation and improve the diversity of the solutions.

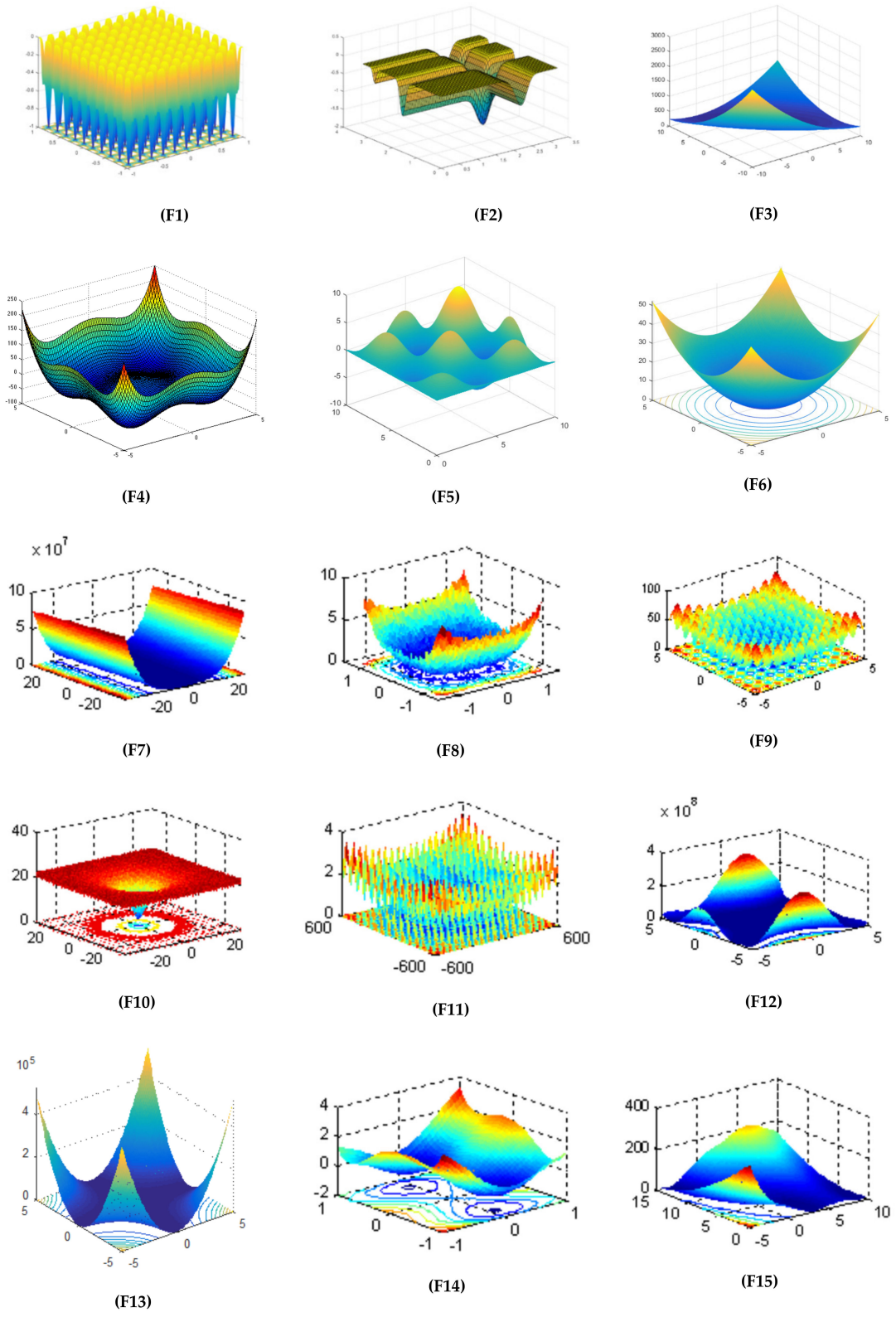

Figure 4. Graphical representation of mathematical objective functions (mathematical objective functions **F1**–**F15** in Table 1).

Table 2. Statistical results of the mathematical objective functions for the optimization approaches (GABONST, GA, enhanced ameliorated teaching learning-based optimization (EATLBO), Bat and Bee)).

	F1	**F2**	**F3**	**F4**	**F5**
GA–RMSE	0.408266	0.66005	0.457172	69.84426	2.734866
GABONST–RMSE	**0.008912**	**0.1498**	**0**	**15.50201**	**0**
EATLBO–RMSE	0.241362	0.803974	0.379706	148.6135	0.715572
Bat-RMSE	0.3820	0.4033	5.7316	153.3258	0.2902
Bee-RMSE	1.0000	0.5693	9.2615×10^{-10}	186.7217	0.2444
GA–Mean	−0.60383	−1.20717	18.4889	−326.997	−3.7482
GABONST–Mean	**−0.99688**	**−1.9511**	**0**	**−380.897**	**−6.1295**
EATLBO–Mean	−0.77554	−0.99733	0.27178	−245.971	−5.67757
Bat-Mean	−0.6215	−1.5889	3.3044	−240.5929	−5.9602
Bee-Mean	-9.4481×10^{-11}	−1.3024	6.1739×10^{-10}	−207.6444	−6.0014
GA–STD	0.099654	0.290461	0.271562	26.66188	1.358616
GABONST–STD	0.008434	$\mathbf{2.24299 \times 10^{-15}}$	**0**	**11.26751**	$\mathbf{4.48598 \times 10^{-15}}$
EATLBO–STD	0.089637	0.00257	0.267856	29.62479	0.56043
Bat-STD	0.0526	0.3463	4.7307	26.4876	0.2381
Bee-STD	$\mathbf{3.1940 \times 10^{-11}}$	0.2769	6.9736×10^{-10}	31.9960	0.2103
	F6	**F7**	**F8**	**F9**	**F10**
GA–RMSE	115.0308	8.1381×10^{3}	0.1612	42.7116	10.9405
GABONST–RMSE	**0**	**0**	$\mathbf{2.2336 \times 10^{-4}}$	**0**	**0**
EATLBO–RMSE	4.4872×10^{-56}	28.9495	$\mathbf{2.7609 \times 10^{-4}}$	205.0804	2.8037
Bat-RMSE	1.2039×10^{3}	9.0087×10^{7}	74.3799	364.5571	20.1364
Bee-RMSE	1.7538×10^{3}	2.3143×10^{7}	14.6593	141.4336	19.6219
GA–Mean	113.9390	6.7387×10^{3}	0.1464	41.6742	10.9243
GABONST–Mean	**0**	**0**	$\mathbf{1.5895 \times 10^{-4}}$	**0**	**0**

Table 2. *Cont.*

EATLBO–Mean	3.3624×10^{-56}	28.9495	2.0063×10^{-4}	202.5195	2.6162
Bat-Mean	1.1761×10^{3}	8.1520×10^{7}	67.4795	362.3866	20.1321
Bee-Mean	1.7533×10^{3}	2.1898×10^{7}	14.3940	140.4960	19.6218
GA–STD	15.9710	4.6090×10^{3}	0.0682	9.4516	39.3751
GABONST–STD	**0**	**0**	$\mathbf{1.5852 \times 10^{-4}}$	**0**	**0**
EATLBO–STD	3.0016×10^{-56}	0.0175	1.9160×10^{-4}	32.6362	1.0182
Bat-STD	259.8600	3.8731×10^{7}	31.6047	40.1246	0.4206
Bee-STD	39.3751	7.5653×10^{6}	2.8049	16.4241	0.0496
	F11	**F12**	**F13**	**F14**	**F15**
GA–RMSE	2.2342	6.4588×10^{-6}	0.0018	$\mathbf{2.8284 \times 10^{-5}}$	1.1264×10^{-4}
GABONST–RMSE	**0**	$\mathbf{7.6591 \times 10^{-14}}$	$\mathbf{1.4195 \times 10^{-4}}$	2.8453×10^{-5}	$\mathbf{1.1264 \times 10^{-4}}$
EATLBO–RMSE	**0**	9.9516	0.0067	0.0372	0.1170
Bat-RMSE	348.4865	21.6007	0.0645	0.8243	0.5704
Bee-RMSE	221.4966	2.5635×10^{-9}	2.7909×10^{-4}	2.8453×10^{-5}	1.1264×10^{-4}
GA–Mean	2.1932	3.000000925712561	0.0017	**−1.031628252987515**	0.3978873583048
GABONST–Mean	**0**	**2.999999999999923**	$\mathbf{4.0025 \times 10^{-4}}$	−1.031628453489878	**0.3978873577297**
EATLBO–Mean	**0**	9.9257	0.0042	−1.0078	0.4496
Bat-Mean	338.7847	18.6428	0.0464	−0.4807	0.7070
Bee-Mean	219.4071	3.000000001676081	5.1854×10^{-4}	−1.031628453353341	0.3979
GA–STD	0.4302	6.4570×10^{-6}	0.0011	1.3312×10^{-6}	2.3933×10^{-9}
GABONST–STD	**0**	$\mathbf{1.8266 \times 10^{-15}}$	$\mathbf{1.0152 \times 10^{-4}}$	$\mathbf{5.4942 \times 10^{-16}}$	$\mathbf{3.3645 \times 10^{-16}}$
EATLBO–STD	**0**	7.2188	0.0055	0.0288	0.1061
Bat-STD	82.4855	15.0473	0.0455	0.6194	0.4843
Bee-STD	30.6602	1.9593×10^{-9}	1.7534×10^{-4}	2.0841×10^{-10}	1.0905×10^{-10}

Comparatively, most of the objective functions' experimental results clearly prove that the GABONST has faster convergence (see Figure 5). This is a result of the good exploitation and exploration offered by the idea of natural selection theory. Figure 5 depicts the comparison results that were obtained from the fifteen objective functions, which compare the GABONST against the conventional GA, EATLBO, Bat and Bee algorithms in a single run. In addition, Figure 5 proves that the reaching ability of the GABONST to the optimal solution is faster and with fewer iterations. Thus, the GABONST will be integrated into the ELM instead of the EATLBO for the purpose of adjusting the input and hidden layer weights.

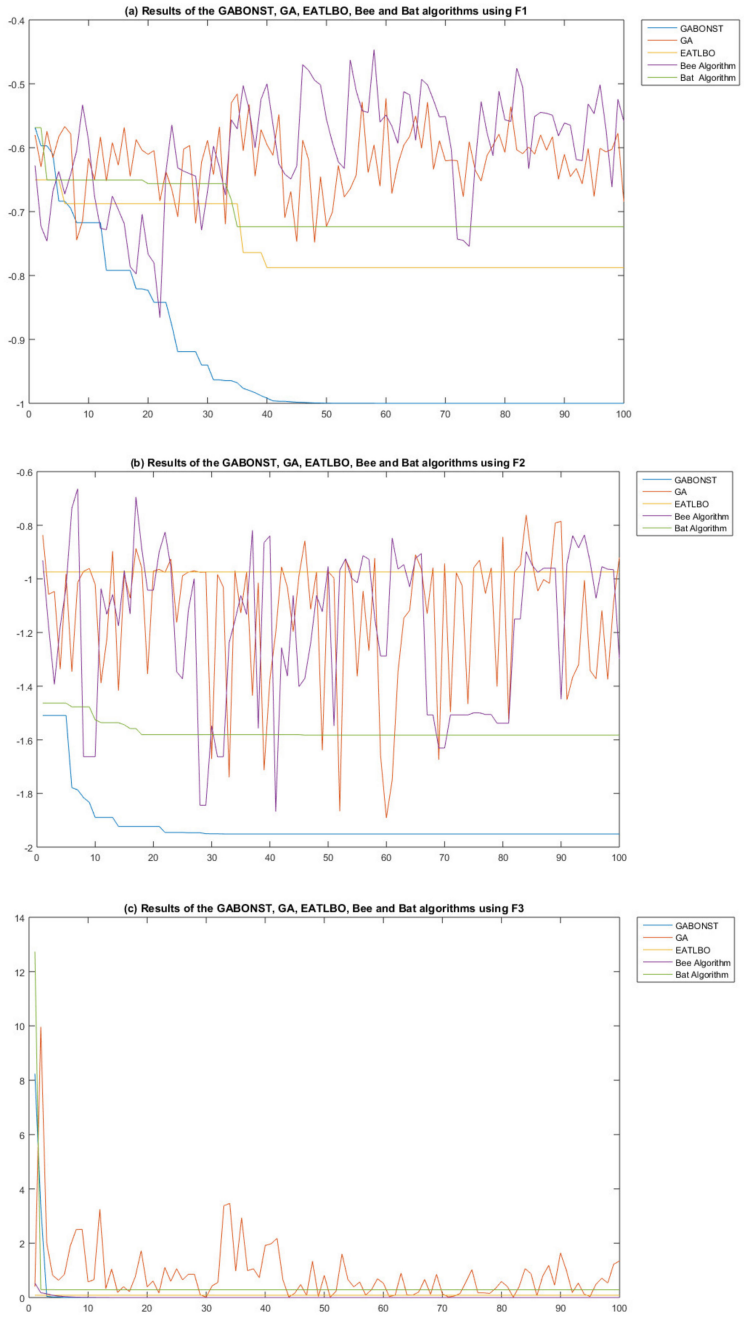

Figure 5. *Cont*.

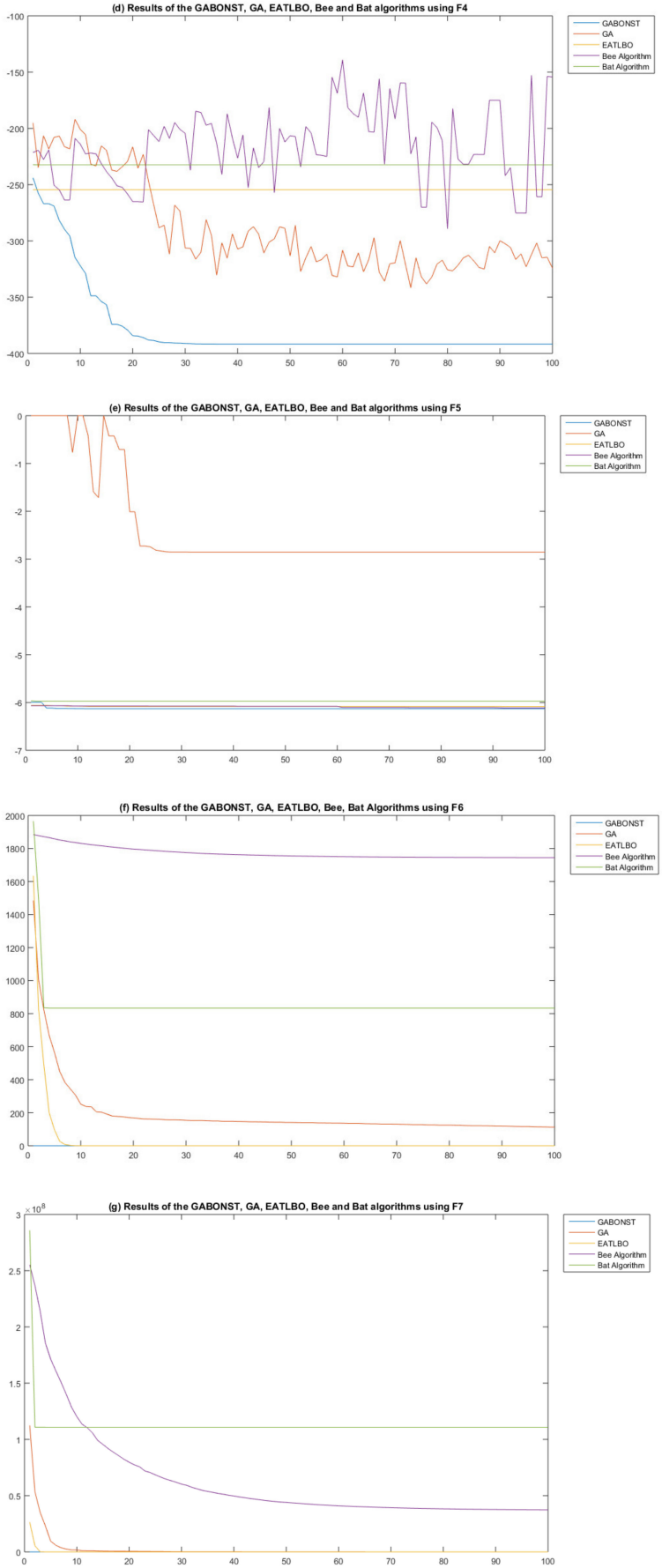

Figure 5. *Cont*.

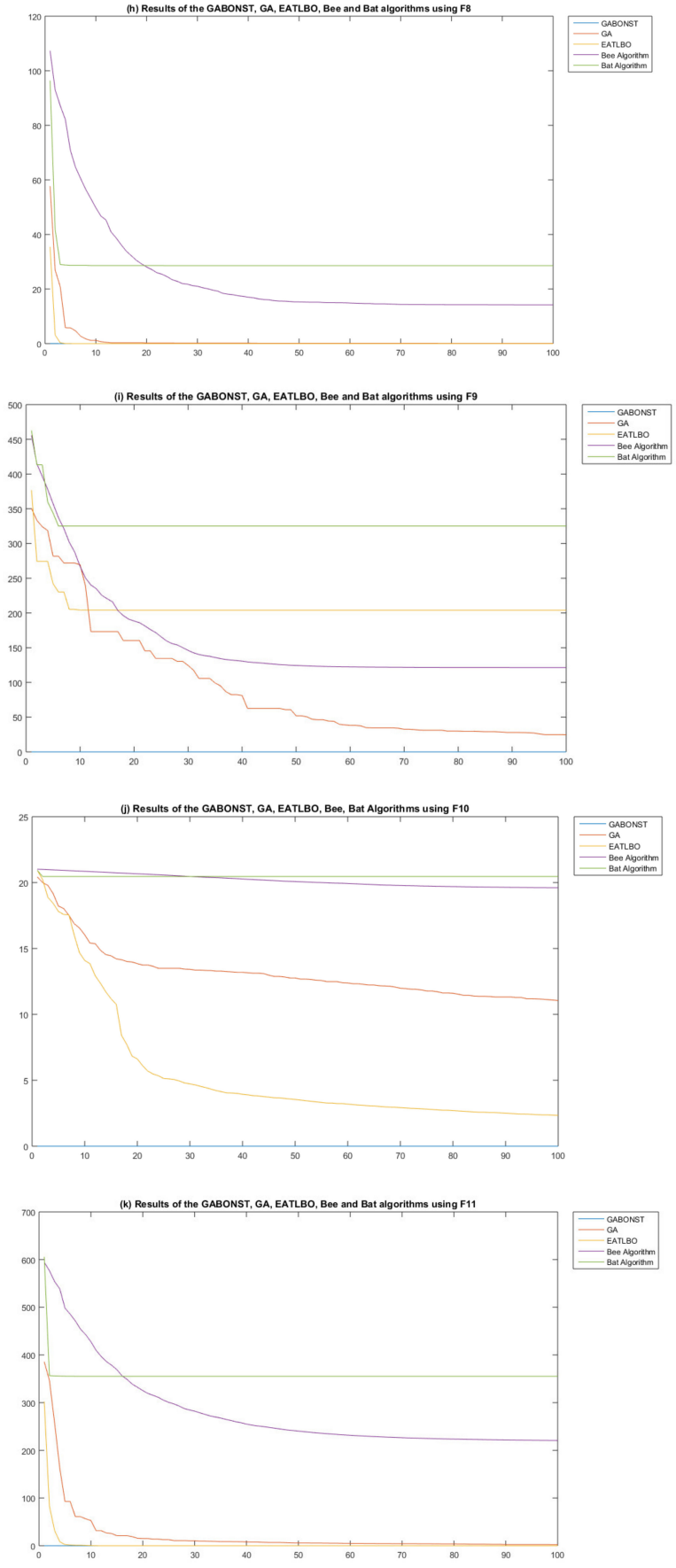

Figure 5. *Cont.*

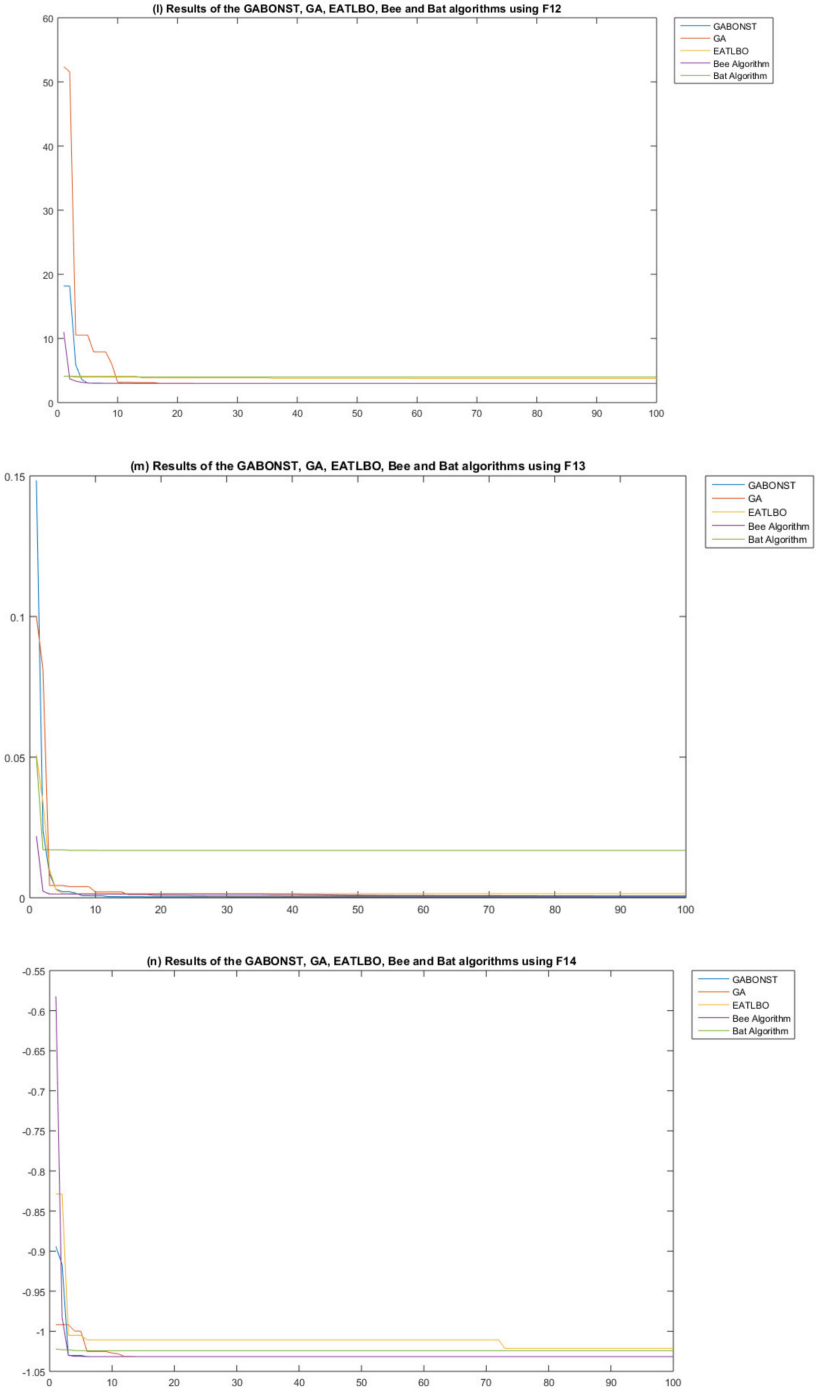

Figure 5. *Cont*.

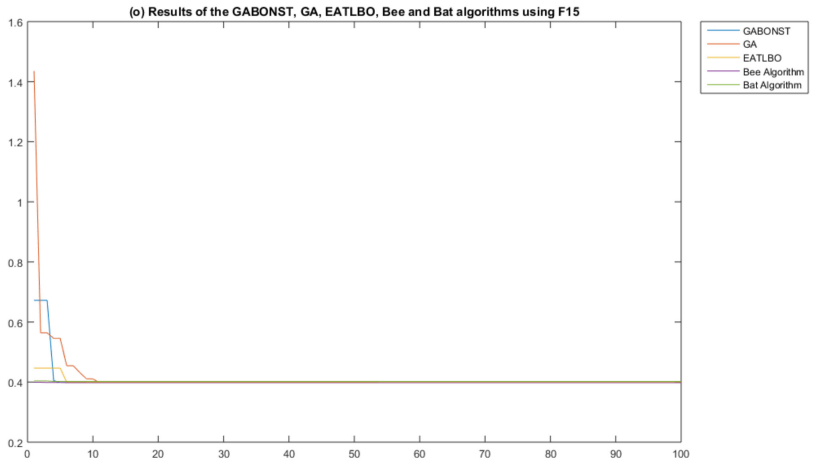

Figure 5. The comparison results of GABONST, GA, EATLBO, Bee and Bat algorithms using the fifteen objective functions (**F1**–**F15**).

The achieved results are shown in Figure 5a–o, which are based on the best solution in each iteration obtained from GABONST, GA, EATLBO, Bee and Bat algorithms using F1–F15 during a single run. The optimal solutions of the F1–F15 are provided in Table 1. The results clearly show the superiority of the GABONST over the traditional GA, EATLBO, Bee and Bat algorithms. The GABONST has achieved and reached the optimal solutions faster and with fewer iterations in comparison to the traditional GA, EATLBO, Bee and Bat algorithms.

3.2. Experimental Test Two

Additionally, this study also aims to evaluate the impact of the proposed GABONST in an application. Thus, this section will implement and evaluate the GABONST in spoken language identification (LID) by integrating the GABONST into the ELM. According to [53], ELM is a single-hidden layer feedforward neural network (SLFN), which entails the random generation of its hidden layer thresholds and input weights. Due to the fact that its output weights are computed using the least squares method, the ELM exhibits speedy training and testing functions. However, its training goal achievement and global minimum requirement are not guaranteed by the randomly generated input weights and hidden layer thresholds, indicating that both are not the best parameters for usage. Many studies have proven that the SLFN's weight optimization as trained by the ELM using various methods is problematical. Several studies have attempted to carry out weights optimization using metaheuristic searching methods [54–57], one of which is the enhanced self-adjusting extreme learning machine (ESA-ELM) [29], which utilizes the teaching and learning phases under the framework of the enhanced ameliorated teaching learning-based optimization (EATLBO). However, the approach of optimization EATLBO is still suffering from several disadvantages such as criteria of selection and the capability of generating good fresh solutions. This could result in an incomplete optimization or a slow rate of convergence, which could not often assure achieving the optimum solution. Therefore, the aim of this study is to enhance the ELM algorithm by integrating the new proposed GABONST into the ELM instead of EATLBO and then implemented into the spoken language identification (LID). Finally, this study aims to prove the capacity of the newly offered GABONST optimization algorithm in enhancing ELM's efficiency and effectiveness as a classifier model for LID.

3.2.1. Basic ELM

One study [53] proposed the training of SLFN by utilizing the initial ELM algorithm. The ELM's main concepts are made up of the random bias generation and the hidden layer weights. The calculation of the output weights was done using the least squares solution as delineated by the hidden layer and

the targets' outputs. Figure 6 details the main idea of the ELM structure. The next subsection briefly explains about the ELM. Table 3 shows the ELM's description along with its notations.

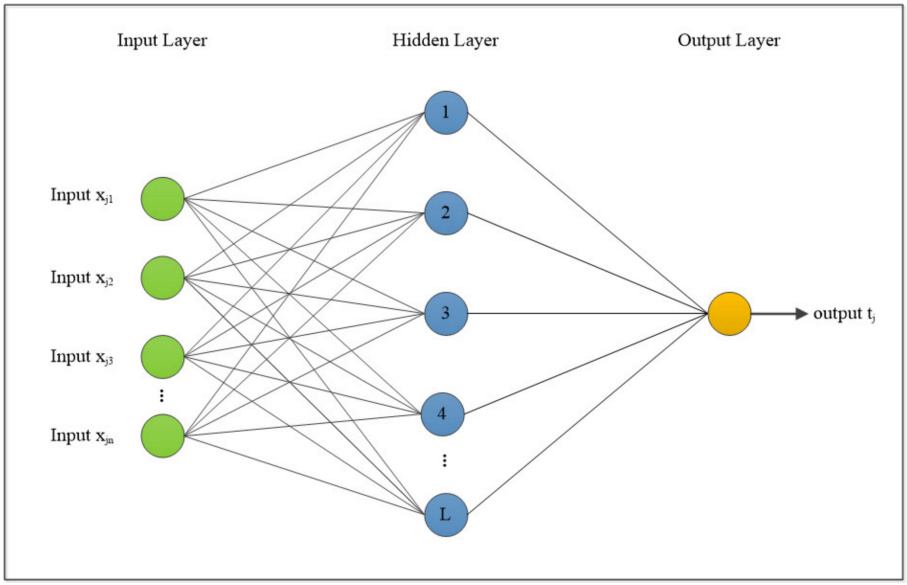

Figure 6. Diagram of the ELM [58].

Table 3. Extreme learning machine's (ELM's) notation table [29].

Notations	Implications
N	distinct samples set (X_i, t_i), where: $X_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T \in Rn$ $t_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^T \in Rm$
L	hidden neurons number
g(x)	activation function, described in Equation (2) [53].
$W_i = [W_{i1}, W_{i2}, \ldots, W_{in}]^T$	input weights that connect the ith input neurons and the hidden neurons
$\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}]^T$	output weight that connect the ith output neurons and the hidden neurons
b_i	threshold of the ith hidden neurons

$$\sum_{i=1}^{L} \beta_i g_i(X_j) = \sum_{i=1}^{L} \beta_i g_i(W_i \cdot X_j + b_i) = o_j. \quad (2)$$

J = 1, . . . , N.

L = number of hidden neurons; and g(x) = activation function of the standard of SLFNs can be N samples without error.

Thus, $\sum_{j=1}^{L} \|o_j - t_j\| = 0$, that is, β_i, W_i and bi exist, such that in Equation (3) [53].

$$\sum_{i=1}^{L} \beta_i g_i(W_i \cdot X_j + b_i) = t_j, \; j = 1, \ldots., N. \quad (3)$$

The equation below is attainable by using the abovementioned equations for N [53]:

$$H\beta = T, \tag{4}$$

where:

$$H\begin{pmatrix} W_1 & \dots & W_L, & b_1 & \dots & b_L, & X_1 & \dots & X_N \end{pmatrix} = \begin{bmatrix} g(W_1.X_1 + b_1) & \cdots & g(W_L.X_1 + b_L) \\ \vdots & \dots & \vdots \\ g(W_1.X_N + b_1) & \cdots & g(W_L.X_N + b_L) \end{bmatrix}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \beta_L^T \end{bmatrix}_{L*m} \text{ and } T = \begin{bmatrix} t_1^T \\ t_N^T \end{bmatrix}_{N*m}$$

Based on [53], H entails the neural network (NN) hidden layer's output matrix; in H, the ith column represents the ith hidden layer nodes on the input nodes. If the preferred number of hidden nodes is $L \leq N$, then the activation function g can substantially be differentiated. Equation (4) then develops into a linear system. The output weights β can be systematically determined by recognizing the least squares solution as below:

$$\beta = H^{\dagger}T \tag{5}$$

where $H^{\dagger}$ is the Moore–Penrose generalized inverse of H. Hence, the calculation of the output weights is based on a mathematical conversion minus a prolonged training phase. Iterative adjustments are conducted on the network parameters with several appropriate learning parameters such as learning rate and iterations.

Without an explicit approach for determining the input-hidden layer weights, the ELM is subjected to local minima, i.e., no method can ensure the usability of the trained ELM in performing the classification. This weakness can be overcome by integrating the ELM with an optimized approach in which the optimal weights are identifiable thus leading to the attainment of the ELM's best performance. The next subsection presents the genetic algorithm based on natural selection theory-extreme learning algorithm (GABONST-ELM) after adopting the GABONST as an optimization approach into the ELM.

3.2.2. GABONST–ELM

The GABONST-ELM is based on the GABONST, which we have described in Section 2.2. GABONST-ELM uses the idea of the natural selection theory along with the GA whereby the processes of selection, crossover and mutation are used to adjust input weight values and hidden nodes bias. Table 4 summarizes the ELM and GABONST parameter values used in the experiments of this study, along with the GABONST-ELM description.

Table 4. Summary of ELM and GABONST parameter settings.

ELM		**GABONST**	
Parameters	**Values**	**Parameters**	**Values**
C	Bias and input weight assemble	Iteration numbers	100
β	Output weight matrix	Population size (PS)	50
Input–weights	−1 to 1	Crossover operation	Arithmetical crossover
Bias values	0–1	Mutation operation	Uniform mutation
Number of input nodes	Input attributes	Selection operation	Select a random solution from the top five solutions of the current population
Number of hidden nodes	650–900, with increment or step of 25	Mean	Mean $= \frac{\sum_i^{PS} f(C_i)}{PS}$
Output neurones	Class values	Gamma	0.4
Activation function	Sigmoid		

Random definitions of the input weights values and hidden nodes are carried out at the onset of the GABONST-ELM and of which were regarded as chromosomes.

$$\text{Vis C} = \left\{ w_{11},\ w_{12},\ \ldots\ w_{1n},\ w_{21},\ w_{22},\ \ldots\ w_{2n},\ w_{m1},\ w_{m2},\ \ldots\ w_{mn},\ b_1,\ \ldots\ b_m \right\} \quad (6)$$

where:

$w_{ij} \in [-1, 1]$, is the input weight value that connect ith hidden node and jth input node

$b_i \in [0, 1]$ = ith hidden node bias

n = input node numbers

m = hidden node numbers

$m \times (n + 1)$ represents the chromosome's dimension, hence requiring parameter optimizations. Therefore, the fitness function in the GABONST–ELM set is calculated utilizing Equation (6).

$$f(C) = \sqrt{\frac{\sum_j^N \left\| \sum_k^m \beta_k g(w_k x_j + b_k) - y_j \right\|_2^2}{N}} \quad (7)$$

where

β: output weight matrix

y_j: true value

N: number of training samples

The procedure of the GABONST-ELM is explained in the following steps:

Firstly, the target function fitness value is calculated for each chromosome C in the population. The fitness value $f(C_i)$ of each C is calculated in order to evaluate C against the mean.

Secondly, the mean of the fitness values is calculated: Mean $= \frac{\sum_i^{PS} f(C_i)}{PS}$. The mean of the fitness values is calculated in order to simulate the environment in biological theory.

Thirdly, compare each chromosome's fitness value with the mean value. If that chromosome's fitness value is equal to or less than the mean then the uniform mutation operation is implemented on that chromosome and moves it into the new generation. This simulates the well-qualified organisms' (chromosomes') survival of the current environment. If that chromosome's fitness value greater than the mean then that chromosome will obtain two chances to be improved. This simulates the idea of giving the unqualified organisms (chromosomes) two chances to adjust their genes and become qualified to survive the current environment:

A. The arithmetical crossover operation is used for exchanging information between that chromosome and a randomly selected chromosome from the top five chromosomes of the current population. The new offspring will be compared to the mean: If it is equal to or less than the mean then move the new offspring into the new generation. If it is greater than the mean then implement step B.

B. The uniform mutation operation is applied to change the genes of that chromosome and generate a new chromosome. The new chromosome will be compared to the mean: if it is equal to or less than the mean then move it into the new generation. If it is greater than the mean then delete that chromosome and add a randomly generated chromosome.

Upon the generation of the new population, the subsequent iteration resumes using this new population, and the whole procedure is reiterated. This iterative process can be stopped when the number of iterations exceeds the maximum limit. The GABONST optimization results are utilized as the input weights and the hidden layer biases of ELM, computing the hidden layers' output matrix H using the activation function g(x). Additionally, the output weights β are calculated using Equation (5) whilst the predicting ELM model is saved for testing. Figure 7 depicts the flowchart of the GABONST-ELM.

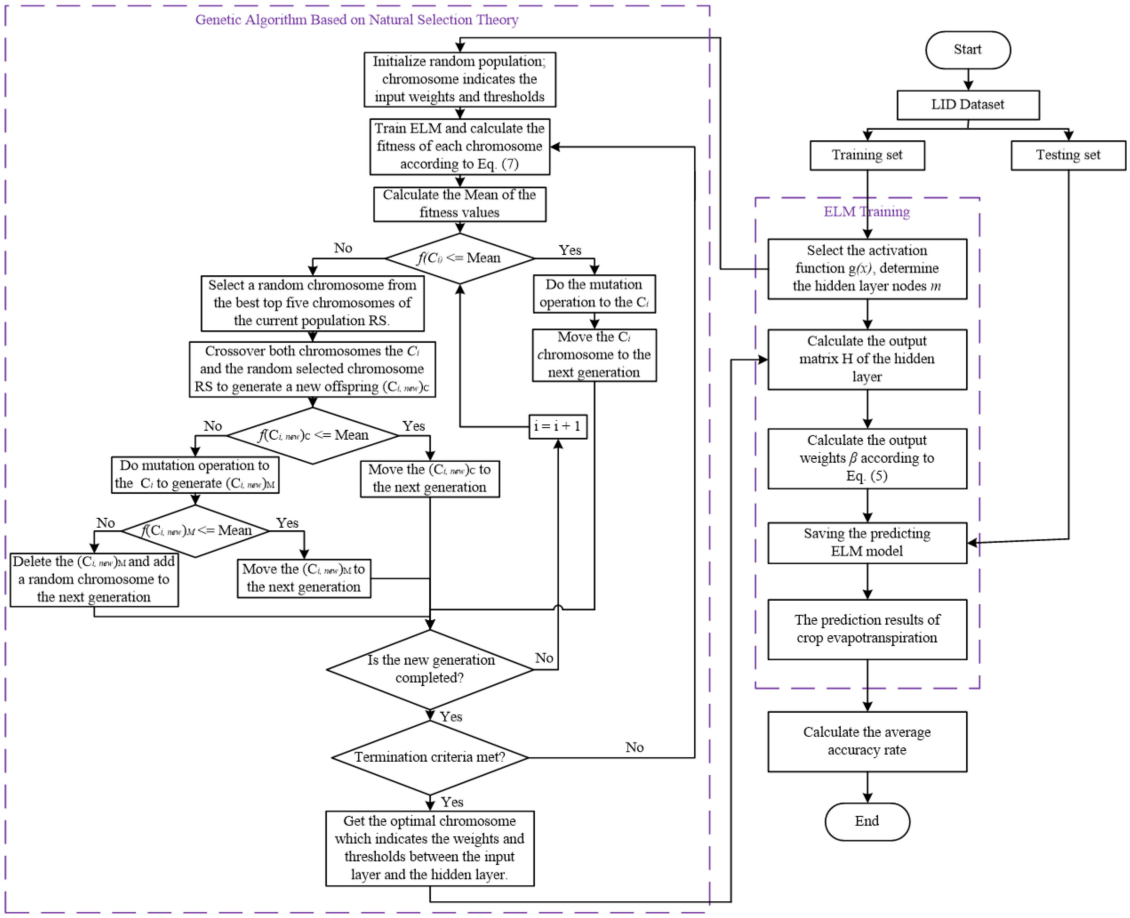

Figure 7. Flowchart of the genetic algorithm based on natural selection theory-extreme learning algorithm (GABONST-ELM).

3.2.3. LID Dataset

This study has used the exact same dataset of the benchmark [29]. Eight spoken languages namely, (1) English, (2) Malay, (3) Arabic, (4) Urdu, (5) German, (6) Spanish, (7) French and (8) Persian were chosen and verified for the purpose of recognition. Audio files were recorded for each language from its respective country's broadcasting media channel as Table 5 provides.

Table 5. List of the media channels [29].

No	Chanel	Language
1	Syrian TV	Arabic
2	British Broadcasting Corporation	English
3	TV9, TV3, and TV2	Malay
4	TF1 HD	French
5	La1, La2, and Real Madrid TV HD	Spanish
6	Zweites Deutsches Fernsehen	German
7	Islamic Republic of Iran News Network	Persian
8	GEO Kahani	Urdu

A total of 15 utterances were recorded for each language, with each utterance lasting 30 s. Training utilized about 67% of the total dataset, which is equal to 80 utterances, whilst testing utilized

the remaining 33%, which is equal to 40 utterances [29]. The recording of the audio files was taken from the channels listed above, whereby each one of the dataset represents one language to determine the algorithm's robustness.

The recording of the utterances was carried out using mp3 together with a dual channel. MATLAB was used as an array entailing two columns that are much alike, despite only one being used. The uttered term corresponded to one vector of the data sampled from the audio file. The length of each utterance was 30 s and each one needed to be sampled and quantified:

1. The sampling rate is 44,100 Hz, based on the Nyquist frequency the highest frequency was 22,050 Hz. The length of 30 s utterance was approximately 1,323,000 (44,100 ∗ 30) samples.
2. Quantization: represents real-valued numbers as integers of a 16-bit range (values from −32,768 to 32,767). The following is a depiction of the utilized dataset:

    a. Name and extension of the dataset: iVectors.mat;
    b. Dimension of the dataset is depicted in Table 6;

Table 6. Dataset dimension [29].

Total Utterance Number	Total Class Number	i-Vector Features Dimension of One Utterance
120	8	600

    c. Depiction of the class is shown in Table 7;

Table 7. Depiction of the class [29].

No	Class Name	Utterance Number
1	Arabic	15
2	English	15
3	Malay	15
4	French	15
5	Spanish	15
6	German	15
7	Persian	15
8	Urdu	15

    d. Feature depiction (as depicted in Table 8);

Table 8. Feature depiction [29].

No	Features Name	Features Type
1→600	i-vector values	Single

    e. The label of the class: last column (column number 601).

3.2.4. Evaluation of the Different Learning Model Parameters

This study used [59] as the basis for the evaluation where numerous measures were applied. [59] handled the classifier evaluation issue and offered effective measures to resolve it. The supervised machine learning offers several evaluation methods for the performance assessment of the learning algorithms and classifiers. Hence, measures concerning classification quality were created

in this study based on a confusion matrix that records recognized examples for each class based on their correction rate. The confusion matrix is one of the most common performance measurement techniques for machine learning classification. Each row of the confusion matrix represents the instances in a predicted class, while each column represents the instances in an actual class [59].

The formulated datasets underwent several classification experiments entailing both the ESA-ELM [29] benchmark and the GABONST-ELM, with a varied amount of hidden neurones ranging from 650–900 and a 25-step increment (following the benchmark scenario [29]). Consequently, there were a total of 11 experiments for the ESA-ELM benchmark and the GABONST-ELM, with 100 iterations for each of the tests.

The ESA-ELM (benchmark) and the GABONST-ELM were hence evaluated using several measures that are based on the ground truth, i.e., utilizing the model for predicting the outcome on the evaluation dataset or held-out data, and comparing that prediction with the real outcome. The evaluation measures were also implemented in the comparison of the benchmark with the GABONST-ELM to determine the false positive, true positive, false negative, true negative, accuracy, recall, precision, G-mean and F-measure. Equations (8)–(12) [29] present the evaluation measures used in this study.

$$\text{accuracy} = \frac{\text{tp} + \text{tn}}{\text{tp} + \text{tn} + \text{fn} + \text{fp}} \tag{8}$$

$$\text{precision} = \frac{\text{tp}}{\text{tp} + \text{fp}} \tag{9}$$

$$\text{recall} = \frac{\text{tp}}{\text{tp} + \text{fn}} \tag{10}$$

$$\text{F} - \text{Measure} = \frac{(2 \times \text{precision} \times \text{recall})}{(\text{precision} + \text{recall})} \tag{11}$$

$$\text{G} - \text{Mean} = \sqrt[2]{\frac{\text{tp}}{\text{p}} \times \frac{\text{tn}}{\text{n}}} \tag{12}$$

where fp = false positive, tp = true positive, fn = false negative, and tn = true negative.

The evaluation of both approaches, ESA-ELM and GABONST-ELM, was based on the same dataset and features the extraction approach with the benchmark [29]. The results for all the experiments carried out between the ESA-ELM and the GABONST-ELM are shown in the figures below. The GABONST-ELM displayed hidden neurones accuracy in the range of 650–900, which was higher than that recorded by the ESA-ELM benchmark, indicating that the performance results of GABONST-ELM in all the iterations are more superior to that of the ESA-ELM benchmark. The comparison of results between both methods in terms of accuracy, precision, recall, F-measure and G-mean are presented in Figures 8–12. The GABONST-ELM achieved the highest accuracy with 725, 800–875 neurons whilst the ESA-ELM achieved an accuracy of 875 neurons (see Figure 8). The GABONST-ELM achieved 99.38% accuracy whilst the ESA-ELM achieved a slightly lower accuracy at 96.25%. The outcomes for ESA-ELM for the other measures are precision 85.00%, recall 85.00%, G-mean 73.41%, and F-measure 85.00%. Meanwhile, GABONST-ELM recorded higher results for all the other measures i.e., recall 97.50%, precision 97.50%, F-measure 97.50% and G-mean 95.06%. The following Tables 9 and 10 present all the results of the evaluation measures for both the ESA-ELM and GABONST-ELM:

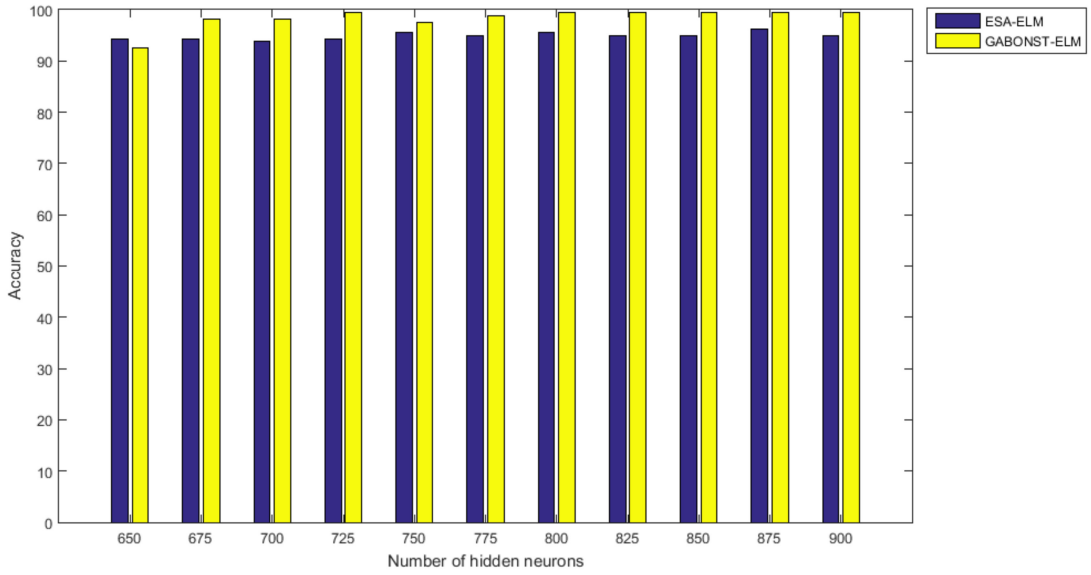

Figure 8. Accuracy results of the GABONST-ELM and enhanced self-adjusting extreme learning machine (ESA-ELM).

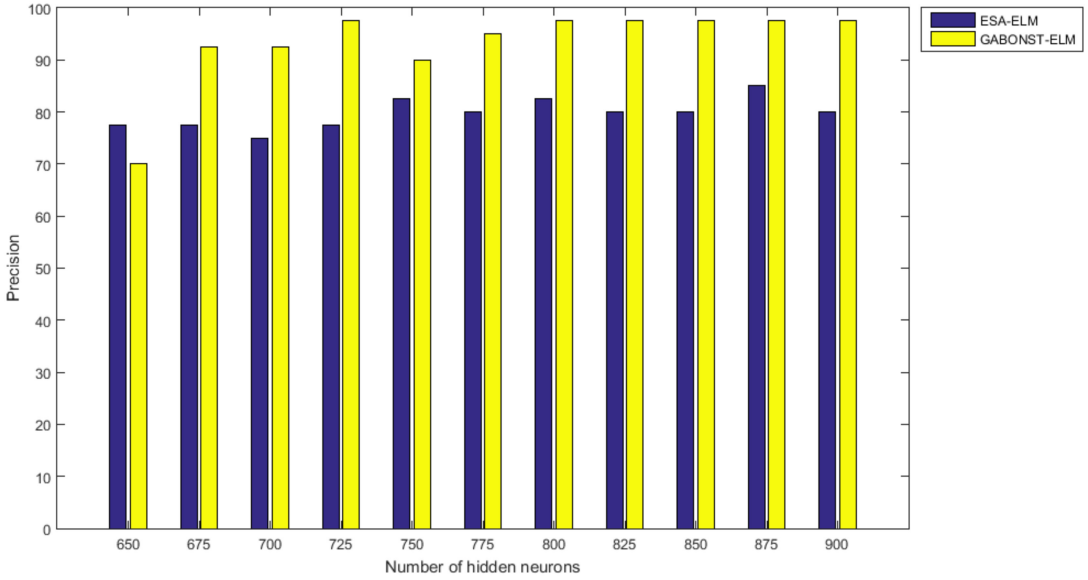

Figure 9. Precision results of the GABONST-ELM and ESA-ELM.

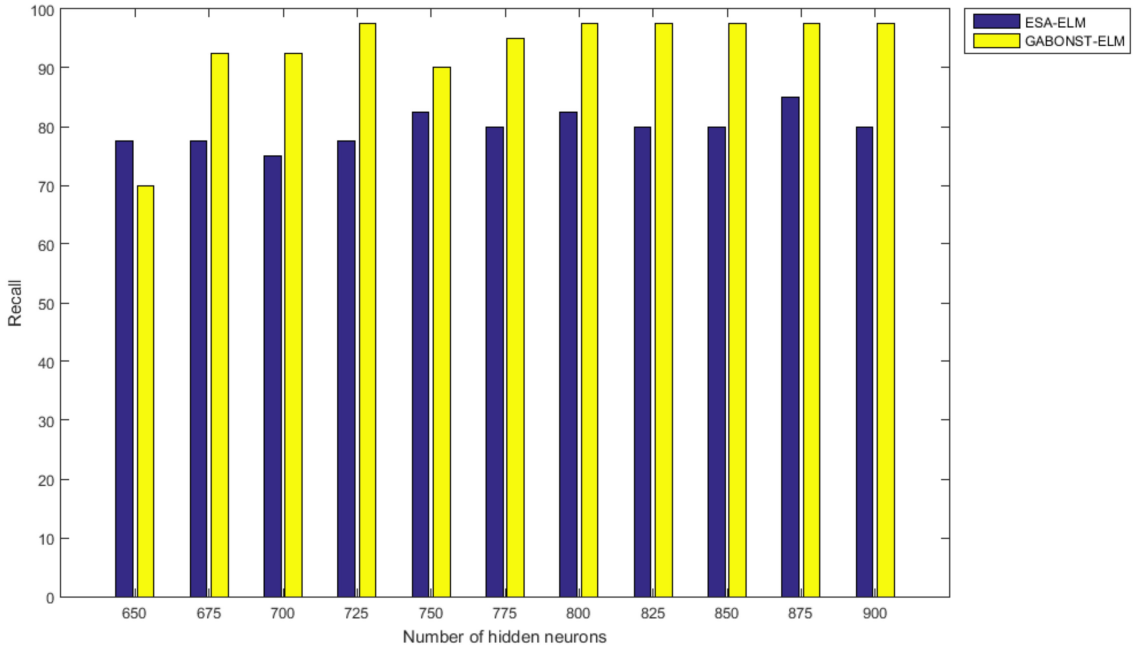

Figure 10. Recall results of the GABONST-ELM and ESA-ELM.

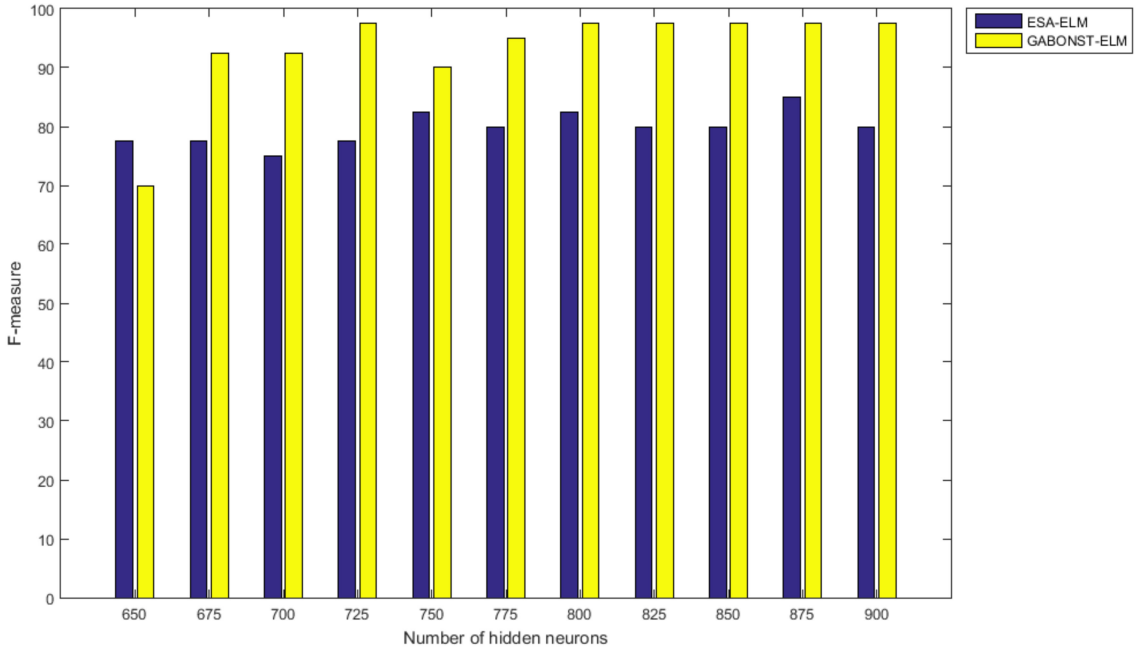

Figure 11. F-measure results of the GABONST-ELM and ESA-ELM.

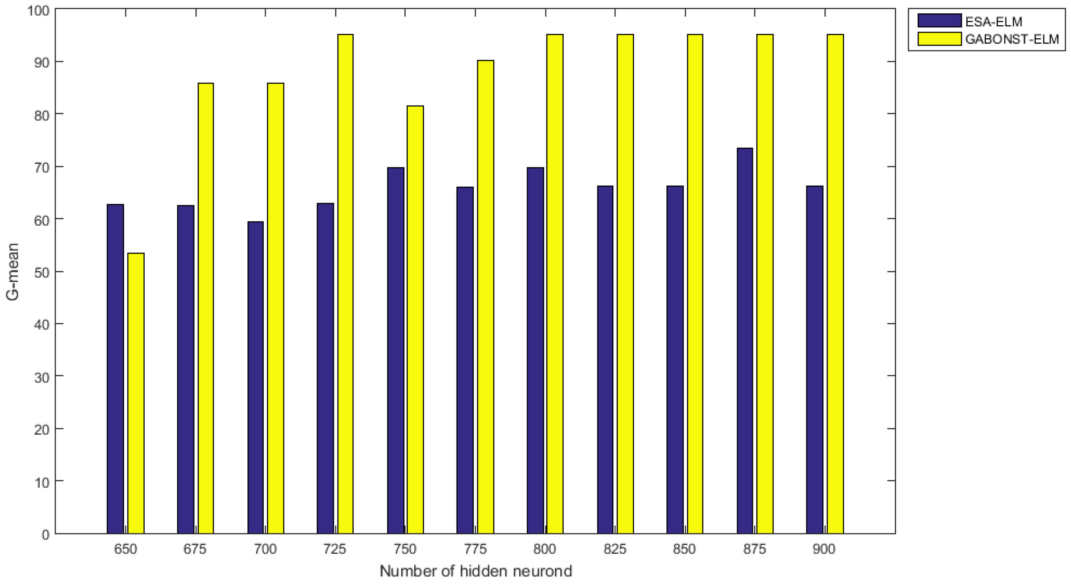

Figure 12. G-mean results of the GABONST-ELM and ESA-LEM.

Table 9. ESA-ELM evaluation results during all the experiments.

Hidden Neuron Numbers	Accuracy	Precision	Recall	F-Measure	G-Mean
650	94.37	77.50	77.50	77.50	62.76
675	94.37	77.50	77.50	77.50	62.50
700	93.75	75.00	75.00	75.00	59.38
725	94.37	77.50	77.50	77.50	62.81
750	95.63	82.50	82.50	82.50	69.64
775	95.00	80.00	80.00	80.00	66.11
800	95.63	82.50	82.50	82.50	69.64
825	95.00	80.00	80.00	80.00	66.16
850	95.00	80.00	80.00	80.00	66.16
875	96.25	85.00	85.00	85.00	73.41
900	95.00	80.00	80.00	80.00	66.20

Table 10. GABONST-ELM evaluation results during all the experiments.

Hidden Neuron Numbers	Accuracy	Precision	Recall	F-Measure	G-Mean
650	92.50	70.00	70.00	70.00	53.40
675	98.12	92.50	92.50	92.50	85.85
700	98.12	92.50	92.50	92.50	85.85
725	99.38	97.50	97.50	97.50	95.06
750	97.50	90.00	90.00	90.00	81.56
775	98.75	95.00	95.00	95.00	90.25
800	99.38	97.50	97.50	97.50	95.06
825	99.38	97.50	97.50	97.50	95.06
850	99.38	97.50	97.50	97.50	95.06
875	99.38	97.50	97.50	97.50	95.06
900	98.12	92.50	92.50	92.50	85.85

Moreover, other experiments were conducted utilizing the i-vector features and the neural network (NN) classifier. "Adam" optimizer and rectified linear unit (ReLU) activation function have been used in the NN. The NN was implemented in LID based on the exact same benchmark's dataset (see Section 3.2.3) with variation of the hidden neuron numbers in the range of 650–900 with an increment step of 25. Table 11 provides all results of the NN during all experiments.

Table 11. Neural network (NN) evaluation results during all the experiments.

Hidden Neuron Numbers	Accuracy	Precision	Recall	F-Measure	G-Mean
650	89.44	55.00	55.00	55.00	31.49
675	90.02	57.50	57.50	57.50	39.85
700	90.55	52.50	52.50	52.50	27.67
725	88.88	47.50	47.50	47.50	27.00
750	89.44	42.50	42.50	42.50	20.34
775	89.16	45.00	45.00	45.00	22.80
800	90.00	55.00	55.00	55.00	36.13
825	89.72	55.00	55.00	55.00	30.03
850	88.88	50.00	50.00	50.00	25.95
875	90.55	55.00	55.00	55.00	30.36
900	90.55	52.50	52.50	52.50	29.51

Additionally, several experiments were performed based on the benchmark's dataset (see Section 3.2.3) for the basic ELM and fast learning network (FLN) with varying numbers of hidden neurons within the range of 650–900 with an increment of 25. Tables 12 and 13 provide the experiment results of the basic ELM and FLN. The highest performance of the basic ELM was achieved with 875 neurons, and the achieved accuracy was 89.38%. The results of other evaluation measures were 57.50%, 57.50%, 57.50% and 40.53% for F-measure, precision, recall, and G-mean, respectively. The highest performance of the FLN was achieved with 725 neurons, and the achieved accuracy was 92.50%. The results of other evaluation measures were 70.00%, 70.00%, 70.00%, and 53.44% for F-measure, precision, recall, and G-mean, respectively.

Table 12. Basic ELM evaluation results during all the experiments.

Hidden Neuron Numbers	Accuracy	Precision	Recall	F-Measure	G-Mean
650	84.38	37.50	37.50	37.50	25.36
675	87.50	50.00	50.00	50.00	34.19
700	85.00	40.00	40.00	40.00	27.13
725	87.50	50.00	50.00	50.00	33.69
750	86.88	47.50	47.50	47.50	32.21
775	88.75	55.00	55.00	55.00	38.43
800	86.25	45.00	45.00	45.00	30.34
825	88.75	55.00	55.00	55.00	38.34
850	86.88	47.50	47.50	47.50	31.86
875	89.38	57.50	57.50	57.50	40.53
900	86.88	47.50	47.50	47.50	31.86

Table 13. Fast learning network (FLN) evaluation results during all the experiments.

Hidden Neuron Numbers	Accuracy	Precision	Recall	F-Measure	G-Mean
650	88.12	52.50	52.50	52.50	36.00
675	89.38	57.50	57.50	57.50	40.63
700	88.75	55.00	55.00	55.00	38.40
725	92.50	70.00	70.00	70.00	53.44
750	90.00	60.00	60.00	60.00	42.88
775	88.75	55.00	55.00	55.00	38.16
800	90.63	62.50	62.50	62.50	45.26
825	89.38	57.50	57.50	57.50	40.49
850	88.75	55.00	55.00	55.00	38.09
875	88.12	52.50	52.50	52.50	36.24
900	90.00	60.00	60.00	60.00	42.65

In addition, several experiments were conducted based on the benchmark's dataset (see Section 3.2.3) for the genetic algorithm-extreme learning machine (GA-ELM), bat-extreme learning machine (Bat-ELM), and bee-extreme learning machine (Bee-ELM) with varying numbers of hidden neurons within the range of 650–900 and an increment of 25. Tables 14–16 provide the experiment results of the GA-ELM, Bat-ELM, and Bee-ELM.

As the results show in Tables 9–16, the performance of GABONST-ELM outperformed the basic ELM, FLN, NN, GA-ELM, Bat-ELM, Bee-ELM and the benchmark ESA-ELM in all experiments. This finding confirms that generating suitable biases and weights for the ELM with single hidden layer reduces classification errors. Avoiding unsuitable biases and weights prevents the ELM from becoming stuck in the local maxima of biases and weights. Therefore, the performance of GABONST-ELM is very impressive, with an accuracy of 99.38%.

Table 14. GA-ELM evaluation results during all the experiments.

Number of Hidden Neurons	Accuracy	Precision	Recall	F-Measure	G-Mean
650	96.88	87.50	87.50	87.50	77.43
675	96.25	85.00	85.00	85.00	73.19
700	95.00	80.00	80.00	80.00	66.16
725	97.50	90.00	90.00	90.00	81.61
750	93.75	75.00	75.00	75.00	59.11
775	97.50	90.00	90.00	90.00	81.56
800	96.88	87.50	87.50	87.50	77.43
825	92.50	70.00	70.00	70.00	53.40
850	95.63	82.50	82.50	82.50	69.74
875	96.88	87.50	87.50	87.50	77.27
900	95.00	80.00	80.00	80.00	66.16

Table 15. Bat-ELM evaluation results during all the experiments.

Number of Hidden Neurons	Accuracy	Precision	Recall	F-Measure	G-Mean
650	93.13	72.50	72.50	72.50	56.20
675	94.37	77.50	77.50	77.50	62.58
700	93.13	72.50	72.50	72.50	56.16
725	93.75	75.00	75.00	75.00	59.46
750	91.87	67.50	67.50	67.50	50.44
775	92.50	70.00	70.00	70.00	53.12
800	93.13	72.50	72.50	72.50	56.37
825	93.75	75.00	75.00	75.00	59.38
850	93.13	72.50	72.50	72.50	56.20
875	93.75	75.00	75.00	75.00	59.33
900	92.50	70.00	70.00	70.00	53.09

Table 16. Bee-ELM evaluation results during all the experiments.

Number of Hidden Neurons	Accuracy	Precision	Recall	F-Measure	G-Mean
650	93.75	75.00	75.00	75.00	59.54
675	93.13	72.50	72.50	72.50	56.37
700	92.50	70.00	70.00	70.00	53.56
725	93.75	75.00	75.00	75.00	59.32
750	91.87	67.50	67.50	67.50	50.24
775	93.75	75.00	75.00	75.00	59.50
800	93.13	72.50	72.50	72.50	56.16
825	93.75	75.00	75.00	75.00	59.37
850	93.13	72.50	72.50	72.50	56.21
875	95.00	80.00	80.00	80.00	66.25
900	92.50	70.00	70.00	70.00	53.29

4. Conclusions

In this study, we proposed the new GABONST based on the existing genetic algorithm (GA) for the optimization problem. GABONST has the same concept as the conventional GA, which imitated the biological structure of the natural world based on Darwin's principles that are made up of three operations, i.e., selection, crossover, and mutation. The GABONST enhanced the conventional GA based on the idea of natural selection theory. It is worth mentioning that all the experiments have been implemented in MATLAB programming language. Based on the algorithm implementation and its results in fifteen different standard test objective functions, the algorithm has shown to be more effective than the conventional GA. This algorithm is primarily advantageous due to its focus on the search space's better area, which resulted from a good exploration–exploitation balance. The good exploration results from the idea of (i) giving the chromosomes that do not satisfy the mean two chances to be improved via the crossover and mutation operations. (ii) By deleting the chromosomes that obtained the two chances and still do not satisfy the mean and adding a randomly generated chromosome. Whilst the good exploitation results from using the idea of the mean that intensifies the search space in the best region. Such an advantage allows for the algorithm's achievement of better convergence. The GABONST is proven to have a better performance than the conventional GA and EATLBO based on the statistical analysis. Additionally, the GABONST-ELM outperformed the

ESA-ELM in LID through adopting the GABONST into the ELM instead of EATLBO. Following this study, the plan is to investigate new alternative selection criteria and integrate them into the concept of choosing a chromosome for the crossover operation instead of random selection criteria and apply it on several possible applications.

Author Contributions: Conceptualization, methodology, writing—original draft, software, writing—review and editing: M.A.A.; supervision, funding acquisition, project administration: S.T.; supervision: M.A.; writing—review and editing, investigation: F.A.-D. All authors have read and agreed to the published version of the manuscript.

Funding: This project was under the funding of the Malaysian government with the research code: GUP-2020-063.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alzaqebah, M.; Abdullah, S. An adaptive artificial bee colony and late-acceptance hill-climbing algorithm for examination timetabling. *J. Sched.* **2013**, *17*, 249–262. [CrossRef]
2. Alzaqebah, M.; Abdullah, S. Hybrid bee colony optimization for examination timetabling problems. *Comput. Oper. Res.* **2015**, *54*, 142–154. [CrossRef]
3. Aziz, R.A.; Ayob, M.; Othman, Z.; Ahmad, Z.; Sabar, N.R. An adaptive guided variable neighborhood search based on honey-bee mating optimization algorithm for the course timetabling problem. *Soft Comput.* **2016**, *21*, 6755–6765. [CrossRef]
4. Sabar, N.R.; Ayob, M.; Kendall, G.; Qu, R. A honey-bee mating optimization algorithm for educational timetabling problems. *Eur. J. Oper. Res.* **2012**, *216*, 533–543. [CrossRef]
5. Jaddi, N.S.; Abdullah, S.; Hamdan, A.R. Multi-population cooperative bat algorithm-based optimization of artificial neural network model. *Inf. Sci.* **2015**, *294*, 628–644. [CrossRef]
6. Jaddi, N.S.; Abdullah, S.; Hamdan, A.R. A solution representation of genetic algorithm for neural network weights and structure. *Inf. Process. Lett.* **2016**, *116*, 22–25. [CrossRef]
7. Serapião, A.B.; Corrêa, G.S.; Gonçalves, F.B.; Carvalho, V.O. Combining K-Means and K-Harmonic with Fish School Search Algorithm for data clustering task on graphics processing units. *Appl. Soft Comput.* **2016**, *41*, 290–304. [CrossRef]
8. Hassanien, A.E.; Moftah, H.M.; Azar, A.T.; Shoman, M. MRI breast cancer diagnosis hybrid approach using adaptive ant-based segmentation and multilayer perceptron neural networks classifier. *Appl. Soft Comput.* **2014**, *14*, 62–71. [CrossRef]
9. Krishna, P.V. Honey bee behavior inspired load balancing of tasks in cloud computing environments. *Appl. Soft Comput.* **2013**, *13*, 2292–2303.
10. Albadr, M.A.A.; Tiun, S. Spoken Language Identification Based on Particle Swarm Optimisation–Extreme Learning Machine Approach. *Circuits Syst. Signal. Process.* **2020**, 1–27. [CrossRef]
11. Albadr, M.A.A.; Tiun, S.; Ayob, M.; Al-Dhief, F.T. Spoken language identification based on optimised genetic algorithm–extreme learning machine approach. *Int. J. Speech Technol.* **2019**, *22*, 711–727. [CrossRef]
12. Yassen, E.T.; Ayob, M.; Nazri, M.Z.A.; Sabar, N.R. A Hybrid Meta-Heuristic Algorithm for Vehicle Routing Problem with Time Windows. *Int. J. Artif. Intell. Tools* **2015**, *24*, 1550021. [CrossRef]
13. Yassen, E.T.; Ayob, M.; Nazri, A.; Zakree, M. The Effect of Hybridizing Local Search Algorithms with Harmony Search for the Vehicle Routing Problem with Time Windows. *J. Theor. Appl. Inf. Technol.* **2015**, *73*, 43–58.
14. Yassen, E.T.; Ayob, M.; Nazri, M.Z.A.; Sabar, N.R. Meta-harmony search algorithm for the vehicle routing problem with time windows. *Inf. Sci.* **2015**, *325*, 140–158. [CrossRef]
15. Agarwal, P.; Mehta, S. Nature-Inspired Algorithms: State-of-Art, Problems and Prospects. *Int. J. Comput. Appl.* **2014**, *100*, 14–21. [CrossRef]
16. Jaddi, N.S.; Abdullah, S. Optimization of neural network using kidney-inspired algorithm with control of filtration rate and chaotic map for real-world rainfall forecasting. *Eng. Appl. Artif. Intell.* **2018**, *67*, 246–259. [CrossRef]
17. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]
18. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; MIT press: Cambridge, MA, USA, 1992.

19. Yang, X.-S. A new metaheuristic bat-inspired algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74.
20. Geem, Z.W.; Kim, J.H.; Loganathan, G. A New Heuristic Optimization Algorithm: Harmony Search. *Simulation* **2001**, *76*, 60–68. [CrossRef]
21. Jaddi, N.S.; Alvankarian, J.; Abdullah, S. Kidney-inspired algorithm for optimization problems. *Commun. Nonlinear Sci. Numer. Simul.* **2017**, *42*, 358–369. [CrossRef]
22. Goldberg, D.E.; Holland, J.H. Genetic algorithms and machine learning. *Mach. Learn.* **1988**, *3*, 95–99. [CrossRef]
23. Holland, J.H. Genetic algorithms. *Sci. Am.* **2012**, *7*, 1482. [CrossRef]
24. Mirjalili, S. Genetic algorithm. In *Evolutionary Algorithms and Neural Networks*; Springer: New York, NY, USA, 2019; pp. 43–55. [CrossRef]
25. Contreras-Bolton, C.; Parada, V. Automatic Combination of Operators in a Genetic Algorithm to Solve the Traveling Salesman Problem. *PLoS ONE* **2015**, *10*, e0137724. [CrossRef] [PubMed]
26. Anam, S. *Parameters Estimation of Enzymatic Reaction Model for Biodiesel Synthesis by Using Real Coded Genetic Algorithm with Some Crossover Operations*; IOP Publishing: Bristol, UK, 2019; Volume 546, p. 052006.
27. Malik, A. A Study of Genetic Algorithm and Crossover Techniques. *Int. J. Comput. Sci. Mob. Comput.* **2019**, *8*, 335–344.
28. Mankad, K.B. *A Genetic Fuzzy Approach to Measure Multiple Intelligence*; Sardar Patel University: Gujarat, India, 2013.
29. Albadr, M.A.A.; Tiun, S.; Al-Dhief, F.T.; Sammour, M.A.M. Spoken language identification based on the enhanced self-adjusting extreme learning machine approach. *PLoS ONE* **2018**, *13*, e0194770. [CrossRef]
30. Holland, J.H. *Adaption in Natural and Artificial Systems. An Introductory Analysis with Application to Biology, Control and Artificial Intelligence*, 1st ed.; The University of Michigan: Ann Arbor, MI, USA, 1975.
31. Bi, C. Deterministic local alignment methods improved by a simple genetic algorithm. *Neurocomputing* **2010**, *73*, 2394–2406. [CrossRef]
32. Mohamed, M.H. Rules extraction from constructively trained neural networks based on genetic algorithms. *Neurocomputing* **2011**, *74*, 3180–3192. [CrossRef]
33. Höschel, K.; Lakshminarayanan, V. Genetic algorithms for lens design: A review. *J. Opt.* **2018**, *48*, 134–144. [CrossRef]
34. Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs. *Math. Intell.* **1996**, *18*, 71. [CrossRef]
35. Yu, F.; Fu, X.; Li, H.; Dong, G. Improved Roulette Wheel Selection-Based Genetic Algorithm for TSP. In Proceedings of the 2016 International Conference on Network and Information Systems for Computers (ICNISC), Wuhan, China, 15–17 April 2016; pp. 151–154.
36. Zhi, H.; Liu, S. Face recognition based on genetic algorithm. *J. Vis. Commun. Image Represent.* **2019**, *58*, 495–502. [CrossRef]
37. Zhang, H.; Xie, J.; Ge, J.; Zhang, Z.; Zong, B. A hybrid adaptively genetic algorithm for task scheduling problem in the phased array radar. *Eur. J. Oper. Res.* **2019**, *272*, 868–878. [CrossRef]
38. Wong, K.-W.; Yap, W.-S.; Wong, D.C.-K.; Phan, R.C.-W.; Goi, B.-M. Cryptanalysis of genetic algorithm-based encryption scheme. *Multimedia Tools Appl.* **2020**, *79*, 25259–25276. [CrossRef]
39. Ahmed, R.; Zayed, T.; Nasiri, F. A Hybrid Genetic Algorithm-Based Fuzzy Markovian Model for the Deterioration Modeling of Healthcare Facilities. *Algorithms* **2020**, *13*, 210. [CrossRef]
40. Kar, S.; Kabir, M.M.J. Comparative Analysis of Mining Fuzzy Association Rule using Genetic Algorithm. In Proceedings of the 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox'sBazar, Bangladesh, 7–9 February 2019; pp. 1–5.
41. Tan, X.; Wu, J.; Mao, T.; Tan, Y. Multi-attribute intelligent decision-making method based on triangular fuzzy number hesitant intuitionistic fuzzy sets. *Syst. Eng. Electron.* **2017**, *39*, 829–836.
42. Li, X.; Wang, Z.; Sun, Y.; Zhou, S.; Xu, Y.; Tan, G. Genetic algorithm-based content distribution strategy for F- RAN architectures. *ETRI J.* **2019**, *41*, 348–357. [CrossRef]
43. Serbanescu, M.-S. Genetic algorithm/extreme learning machine paradigm for cancer detection. *Ann. Univ. Craiova Math. Comput. Sci. Ser.* **2019**, *46*, 372–380.

44. Choudhary, A.; Kumar, M.; Gupta, M.K.; Unune, D.K.; Mia, M. Mathematical modeling and intelligent optimization of submerged arc welding process parameters using hybrid PSO-GA evolutionary algorithms. *Neural Comput. Appl.* **2019**, 1–14. [CrossRef]
45. Jamali, B.; Rasekh, M.; Jamadi, F.; Gandomkar, R.; Makiabadi, F. Using PSO-GA algorithm for training artificial neural network to forecast solar space heating system parameters. *Appl. Therm. Eng.* **2019**, *147*, 647–660. [CrossRef]
46. Lipare, A.; Edla, D.R.; Cheruku, R.; Tripathi, D. GWO-GA Based Load Balanced and Energy Efficient Clustering Approach for WSN. In *Smart Trends in Computing and Communications*; Springer: Singapore, 2020; pp. 287–295. [CrossRef]
47. Beg, A.H.; Islam, Z. Novel crossover and mutation operation in genetic algorithm for clustering. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 2114–2121. [CrossRef]
48. Kora, P.; Yadlapalli, P. Crossover Operators in Genetic Algorithms: A Review. *Int. J. Comput. Appl.* **2017**, *162*, 34–36. [CrossRef]
49. Darwin, C.; Wallace, A.R. *Evolution by Natural Selection*; Cambridge University Press: Cambridge, UK, 1958.
50. Livezey, R.L.; Darwin, C. On the Origin of Species by Means of Natural Selection. *Am. Midl. Nat.* **1953**, *49*, 937. [CrossRef]
51. Jamil, M.; Yang, X.S. A literature survey of benchmark functions for global optimisation problems. *Int. J. Math. Model. Numer. Optim.* **2013**, *4*, 150–194. [CrossRef]
52. Jain, M.; Singh, V.; Rani, A. A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm Evol. Comput.* **2019**, *44*, 148–175. [CrossRef]
53. Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501. [CrossRef]
54. Alexander, V.; Annamalai, P. An Elitist Genetic Algorithm Based Extreme Learning Machine. *Softw. Eng. Intell. Syst.* **2015**, 301–309. [CrossRef]
55. Nayak, P.K.; Mishra, S.; Dash, P.K.; Bisoi, R. Comparison of modified teaching–learning-based optimization and extreme learning machine for classification of multiple power signal disturbances. *Neural Comput. Appl.* **2015**, *27*, 2107–2122. [CrossRef]
56. Niu, P.; Ma, Y.; Li, M.; Yan, S.; Li, G. A Kind of Parameters Self-adjusting Extreme Learning Machine. *Neural Process. Lett.* **2016**, *44*, 813–830. [CrossRef]
57. Yang, Z.; Zhang, T.; Zhang, D. A novel algorithm with differential evolution and coral reef optimization for extreme learning machine training. *Cogn. Neurodyn.* **2015**, *10*, 73–83. [CrossRef]
58. Albadra, M.A.A.; Tiuna, S. Extreme learning machine: A review. *Int. J. Appl. Eng. Res.* **2017**, *12*, 4610–4623.
59. Sokolova, M.; Japkowicz, N.; Szpakowicz, S. *Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1015–1021.