Soft Computing
https://doi.org/10.1007/s00500-022-07115-7

FOCUS

ST-AL: a hybridized search based metaheuristic computational algorithm towards optimization of high dimensional industrial datasets

Reham R. Mostafa[1] · Noha E. El-Attar[2] · Sahar F. Sabbeh[2,3] · Ankit Vidyarthi[4] · Fatma A. Hashim[5]

Accepted: 23 March 2022

Abstract
The rapid growth of data generated by several applications like engineering, biotechnology, energy, and others has become a crucial challenge in the high dimensional data mining. The large amounts of data, especially those with high dimensions, may contain many irrelevant, redundant, or noisy features, which may negatively affect the accuracy and efficiency of the industrial data mining process. Recently, several meta-heuristic optimization algorithms have been utilized to evolve feature selection techniques for dealing with the vast dimensionality problem. Despite optimization algorithms' ability to find the near-optimal feature subset of the search space, they still face some global optimization challenges. This paper proposes an improved version of the sooty tern optimization (ST) algorithm, namely the ST-AL method, to improve the search performance for high-dimensional industrial optimization problems. ST-AL method is developed by boosting the performance of STOA by applying four strategies. The first strategy is the use of a control randomization parameters that ensure the balance between the exploration–exploitation stages during the search process; moreover, it avoids falling into local optimums. The second strategy entails the creation of a new exploration phase based on the Ant lion (AL) algorithm. The third strategy is improving the STOA exploitation phase by modifying the main equation of position updating. Finally, the greedy selection is used to ignore the poor generated population and keeps it from diverging from the existing promising regions. To evaluate the performance of the proposed ST-AL algorithm, it has been employed as a global optimization method to discover the optimal value of ten CEC2020 benchmark functions. Also, it has been applied as a feature selection approach on 16 benchmark datasets in the UCI repository and compared with seven well-known optimization feature selection methods. The experimental results reveal the superiority of the proposed algorithm in avoiding local minima and increasing the convergence rate. The experimental result are compared with state-of-the-art algorithms, i.e., ALO, STOA, PSO, GWO, HHO, MFO, and MPA and found that the mean accuracy achieved is in range 0.94–1.00.

Keywords Sooty tern optimization · Ant lion optimization · Feature optimization · Metaheuristic algorithm · High dimensional search space

1 Introduction

In the past decades, optimization issues have attracted extensive attention in several fields, to name a few: computer science, engineering, operational research, energy, and business (Oliva and Elaziz 2020). In general, optimization techniques aim to identify the best solutions from a set of available alternatives in the problem search space. Optimization problems can be categorized into binary or continuous, static or dynamic, single-objective or multi-objective, and constrained or unconstrained (Hussien and Amin 2021). In sophisticated optimization problems, it is imperative to investigate the search space adequately based on the problem type (Anand and Arora 2020). Consequently, due to the growing complexity in optimization problems and the variety in their types, the conventional mathematical techniques (e.g., Newton and gradient descent) have become worthless due to their substantial time-consuming and probability of falling in local optima problem (Hussien and Amin 2021).

Meta-heuristic techniques have been successfully developed to handle a lot of tough optimization problems

Communicated by Priti Bansal.

Extended author information available on the last page of the article

Published online: 09 May 2022

effectively. They have the ability to exploit significant information from the search space and determine the optimal solution rapidly and efficiently (Anand and Arora 2020). Almost all meta-heuristic algorithms have been inspired by nature, like the behavior of animals, birds, insects, and even humans (Hussien and Amin 2021). Genetic algorithm (GA) (Goldberg and Holland 1988), particle swarm optimization (PSO) (Eberhart and Kennedy 1995), differential evolution (DE) (Storn and Price 1997), firefly algorithm (Yang 2010), flower pollination algorithm (FPA) (Yang 2012), artificial bee colony (ABC) (Karaboga and Basturk 2007), and grey wolf optimization algorithm (GWO) (Mirjalili et al. 2014) are examples of the original and prominent meta-heuristic algorithms. Recently, there are several nature-inspired meta-heuristic techniques have been innovated, to name a few, Grasshopper optimization algorithm (GOA) (Mirjalili et al. 2018), selfish herd optimizer (SHO) (Fausto et al. 2017), honey badger algorithm (HBA) (Hashim et al. 2022), butterfly optimization algorithm (BOA) (Arora and Singh 2019), Sine Cosine Algorithm (SCA) (Mirjalili 2016), Salp Swarm Algorithm (SSA) (Mirjalili et al. 2017), and Snake Optimizer (SO) (Hashim and Hussien 2022).

Primarily, the meta-heuristic algorithm contains two fundamental stages: exploration and exploitation. The exploration phase is commonly based on randomization methods used to search effectively in the search space. At the same time, the exploitation phase concerns finding the most promising region of the search space. On the other hand, working on knowledge discovery over high-dimensional datasets is crucial. It needs to prepare the data through a pre-processing data stage (Anand and Arora 2020). This pre-processing step is used mainly to reduce the dimensionality of high dimensional data by neglecting and stripping the irrelevant, redundant, missing, and noisy features from the data set (Sayed et al. 2018). In general, the feature selection process is considered a vital data pre-processing method for coping with the dimensionality curse. Feature selection strategies aim to pick a subset of features based on a set of criteria while maintaining the physical meanings of the original features (Huang et al. 2020). The feature selection process can boost learning model comprehension and perception by reducing the search space size to increase learning efficiency (i.e., training time and classifier complexity are reduced, and prediction performance or classification accuracy is improved) (Zhang et al. 2014).

Commonly, feature selection approaches are divided into three categories based on the methods used to evaluate feature subsets: filter, wrapper, and embedding methods (Neggaz et al. 2020). The intrinsic properties of the data are used to select features for a filter method (Teng et al. 2017). Filter methods are called classifier-independent since they evaluate important information for classification regardless of the machine learning technique (Rani and Rajalaxmi 2015). Filter approaches are quick since they don't use a learning algorithm to analyze attributes, but they don't provide enough information to categorize samples. The Fast Correlation-based Filter (FCBF) and the minimal-redundancy-maximal-relevance (mRMR) are two filter types. Wrapper and embedded models, on the other hand, are dependent on the classifier. The wrapper model investigates the space of potential solutions using a machine learning technique (Emary et al. 2016). To evaluate the selected subset, the validation accuracy of a certain classifier is used. Embedded-based approaches discover, as the classification model is being built, which features have the greatest impact on its accuracy. A wrapper method typically outperforms a filter method since the proposed subset of features is evaluated for accuracy using feedback from the learning algorithm. However, computationally, they are more expensive, and in terms of performance, they depend on the applied learning method.

Accordingly, the most critical aspect of the feature selection algorithm is searching for an optimal or nearly optimal subset of features that increase the classifier's accuracy and reduce the computational complexity. Exhaustive search methods like breadth and depth searches are considered infeasible for discovering a subset of features, especially in massive datasets. A dataset containing M features requires the production of 2M feature subsets. The quality of these feature subsets needs to be evaluated (Zhang et al. 2014), which is computationally intensive, especially in wrapper-based approaches, where the learning algorithm must be implemented for each subset. The best way is to treat feature selection as an NP-hard optimization problem. The objective function minimizes the number of selected features while preserving the highest classification accuracy. This means that feature selection problems could benefit from metaheuristics, which have shown extraordinary performance in tackling various optimization problems (Motoda and Liu 2002). Metaheuristic algorithms have the ability to address complex optimization problems because of their dynamic search behaviors and global search capability. Indeed, several meta-heuristic algorithms have been utilized to improve the performance of feature selection process, to name a few, genetic algorithms (Oh et al. 2004), particle swarm optimization (Gu et al. 2018), ant colony optimization (ACO) algorithm (Aghdam et al. 2009), artificial bee colony (ABC) algorithm (Uzer et al. 2013), binary gravitational search algorithm (BGSA) (Papa et al. 2011), scatter search algorithm (SSA) (Wang et al. 2012), archimedes optimization algorithm (AOA) (Desuky et al. 2021), backtracking search algorithm (BSA) (Ghanem and Layeb 2021), and moth-flame optimization (MFO) algorithm (Soliman et al. 2018).

Most of the originally introduced optimization techniques often suffer from some performance shortcomings, especially when implemented in large-scale datasets. These shortcomings are due to the imbalance between the exploration and exploitation stages, leading to falling into local optima or not converging properly. In this case, most of the feature selection literature has recently tended to modify existing metaheuristics algorithms to improve their performance or hybridize between different metaheuristics algorithms to take advantage of one technique to improve the search efficiency of the other. For instance, the hybridization between Harris hawks optimization (HHO) algorithm with simulated annealing (SA) (Abdel-Basset et al. 2021), arithmetic optimization algorithm (AOA) with genetic algorithm (GA) (Ewees et al. 2021), salp swarm algorithm (SSA) with sine cosine algorithm (SCA) (Neggaz et al. 2020), and the combination of seagull optimization algorithm (SOA) and Lévy flight and mutation operator (Ewees et al. 2022).

However, these methodologies have some restrictions that impact the ultimate solution's quality. Based on the No Free Lunch Theorem (NFL) (Wolpert and Macready 1997), it is concluded that no algorithm is better than all others with all classes of feature selection problems. Therefore, a new algorithm or an improved version of an existing one must be devised to deal with feature selection challenges more effectively. This is the primary motivation for us to propose a new feature selection approach based on enhancing the performance of a novel metaheuristic algorithm, known as the Sooty Tern Optimization Algorithm (STOA) Dhiman and Kaur (2019). This improvement is made by using the Ant lion optimization (ALO) (Mirjalili 2015a) algorithm to enhance the exploration of STOA due to ALO's capacity to locate the feasible regions that contain the optimal solution.

The STOA algorithm is a new population-based metaheuristic algorithm developed by Dhiman and Kaur, through simulating the migration and attacking behaviors of sea bird sooty tern in nature (Dhiman and Kaur 2019). It has gotten a lot of attention in the last few decades and has been used in a variety of applications (Ali et al. 2021; Zheng et al. 2021; Kader and Zamli 2022). Despite eminent applications, STOA is still needed more improvement to overcome its limitations. For example, the STOA exploration phase is based on the best solution only which prevents it to explore the search space properly in order to find the prominent region that contains the optimal solution. On the other hand, ALO is popular metaheuristic algorithm proposed by Mirjalili (2015a), and it is inspired by the hunting mechanism of antlions. It is characterized by good exploration and exploitation phases, avoidance of falling into the local optimum level, and rapid convergence of the optimal solution.

In this study, a novel hybridization technique was proposed based on boosting the performance of STOA through the use of the ALO algorithm. This hybridization is called the ST-AL method. The performance of the proposed ST-AL method was assessed using two experiments; (1) solving global optimization problems and (2) solving feature selection challenges. The main contributions of this paper can be summarized as follows:

1. Developed a novel hybrid method based on Sooty Tern Optimization Algorithm (ST) and Ant Lion Optimization (AL). The proposed method is called ST-AL.
2. Tested ST-AL on CEC'2020 test suite.
3. Employed ST-AL as a wrapper feature selection algorithm for large and small benchmark datasets
4. Comparing the performance of ST-AL with established swarm intelligence algorithms such as PSO, GWO, HHO, MFO, MPA and conventional ST and AL algorithms
5. Demonstrated the effectiveness and superiority of the proposed ST-AL in both global optimization and feature selection problems.

The rest of the paper is organized as follows: Sect. 2 presented the detailed overview on the related work. To understand the methodology, a preliminary study about the algorithms is presented in Sect. 3. The detailed overview on the proposed methodology is presented in Sect. 4. The performance evaluation of the proposed algorithm is given in Sect. 5. At the last, the work is concluded with future scope in Sect. 6.

2 Related works

Recently, meta-heuristic algorithms have attracted attention as an efficient technique to find the optimal solutions and enhance the feature selection process, especially with the massive increase of the data volume and in the level of its complexity. To enhance the optimization process, several studies have developed robust current meta-heuristic optimization algorithms to overcome the local optima problem in the ample solutions space. For instance, some researchers have used chaotic search to enhance the search process and solve local optima problems and low convergence rates, such as Arora et al. (2020). In this study, the authors have presented a novel Chaotic Interior Search Algorithm (CISA) based on integrating the Interior Search Algorithm and the chaos theory to solve the entrapment of both local optima and slow convergence speed. To evaluate the proposed algorithm, it has been tested on 13 global benchmark functions. Also, Sayed et al. have adopted chaos theory to enhance the performance of the Salp Swarm Algorithm (SSA) and proposed Chaotic Salp

Table 1 Recent approaches of hybrid optimization techniques

References	Utilized algorithms	Year
Zhang et al. (2016)	Firefly algorithm and differential evolution	2016
Sayed et al. (2018)	Salp swarm algorithm and chaotic maps	2018
Jia et al. (2019)	Seagull optimization algorithm with thermal exchange op-timization	2019
Oliva and Elaziz (2020)	Brainstorm optimization algorithm, chaotic maps, and opposition-based learning	2020
Khamees and Al-Baset (2020)	Sine cosine algorithm and cuckoo search	2020
Khaleel and Mitras (2020)	Whale optimization algorithm with modified conjugate gra-dient algorithm	2020
Anand and Arora (2020)	Chaotic search and Selfish Herd Optimizer	2020
Arora et al. (2020)	Interior search algorithm and chaos theory	2020
Hussien and Amin (2021)	Harris hawks optimization, opposition-based learning, chaotic local search, and self-adaptive technique	2021
Wang et al. (2021)	Harris hawks optimization with Aquila optimizer	2021
Long et al. (2021)	Butterfly optimization algorithm and Pinhole-imaging-based learning	2021
EL-Hasnony et al. (2021)	Butterfly algorithm with PSO	2021
Assiri (2021)	Butterfly algorithm, chaotic local search, and opposition-based learning	2021
Che and He (2021)	Whale optimization with Seagull optimization algorithm	2021
Adamu et al. (2021)	Chaotic crow search and PSO	2021

Swarm Algorithm. This paper has employed ten different chaotic maps to improve the convergence rate and resulting accuracy (Sayed et al. 2018). Chaotic search has also boosted the search process of selfish herd optimizers (SHO) in Anand and Arora (2020). Anand and Arora have proposed a Chaotic Selfish Herd Optimizer (CSHO) algorithm with various chaotic maps to substitute the value of each searching agent's survival parameter, which helped in controlling both exploration and exploitation processes. Likewise, in Oliva and Elaziz (2020) have applied chaotic maps and opposition-based learning (OBL) to enhance the Brainstorm optimization algorithm (BSO) performance. The proposed algorithm was called opposition chaotic BSO with disruption (OCBSOD). The idea of this algorithm can be summarized in the following steps: first, the chaotic map was applied to compute the initial solutions; after that, the opposition-based learning produced the opposite positions in the search space, then, the best particles were identified and applied in the iterative process. The role of the disruption operator was to update the position of the instance in the population. Finally, the OBL was applied to enhance the exploration process of the search domain.

Harris hawks optimization (HHO) is another recent meta-heuristic algorithm inspired by Harris's cooperative manner and chasing behavior. The performance of HHO has been improved by integrating it with various optimization techniques like opposition-based learning, Chaotic Local Search, and a self-adaptive technique in Hussien and Amin (2021). Wang et al. (2021) also have tried to enhance the HHO searching performance for global optimization by developing a hybrid algorithm that combines HHO with Aquila Optimizer (AO).

In the same context, Long et al. have developed a modified version of the Butterfly optimization algorithm BOA with adaptive gbest-guided search strategy and pinhole-imaging-based learning to overcome the problem of local optimum, which may occur when solving high dimensional optimization problems (Long et al. 2021). This proposed algorithm (PIL-BOA) has been investigated on 23 classical benchmark test functions, 30 complex benchmark functions of IEEE CEC2014, 30 latest benchmarks from CEC 2017, and 21 feature selection problems. Also, EL-Hasnony et al. (2021) have modified the butterfly algorithm by combining it with the PSO algorithm to boost its global optimization performance. In this study, the authors investigated the performance of the proposed algorithm on the COVID-19 dataset. Chaotic Local Search and Opposition-based have also been integrated with to butterfly optimization algorithm to gain the most optimal or near-optimal results in Assiri (2021).

Whale optimization algorithm (WOA) based on simulating Humpback Whales' behavior in their manner in food searching and migration has also been combined with a modified conjugate gradient algorithm in Khaleel and Mitras (2020). This hybrid algorithm is based on deriving a new conjugate coefficient to enhance the efficacy of global optimization problem-solving. In another context, WOA has been used to enhance other optimization algorithms due to its strong global search ability, like in Che and He (2021). This study integrated the WOA with the Seagull optimization algorithm (SOA) and presented a modified

version of SOA called WSOA. Thermal exchange optimization was another optimization algorithm combined with SOA to enhance its exploitation ability and solve feature selection problems Jia et al. (2019). Several other types of research have presented the hybridization between various optimization techniques such as chaotic crow search and particle swarm optimization algorithm in Adamu et al. (2021), sine cosine algorithm and cuckoo search in Khamees and Al-Baset (2020), and Firefly algorithm and differential evolution (Zhang et al. 2016).

According to the various mentioned studies' findings, optimization algorithms still worthwhile need to be developed to enhance the exploitation ability and solve global optimization problems like tardy convergence, low computational accuracy, and falling in local optima. Table 1 displays the recent research that applied the idea of hybridization to enhance metaheuristic optimization algorithms and solve the feature selection problem. This paper presents a new approach to pick the most informative features by boosting the performance of the Sooty Tern Optimization algorithm (STOA) and hybridizing it with the Ant Lion algorithm (ALO).

Table 2 Parameter settings of each experiments

Parameter name	Problem	Value
Population size (N)	CEC2020	30
	Feature selection	30
Max iterations (tmax)	CEC2020	3000
	Feature selection	100
Problem dimension (dim)	CEC2020	10 and 20
	Feature selection	Dataset features
Number of independent runs	CEC2020	30
	Feature selection	30

3 Preliminary study about algorithms

3.1 Ant lion optimization (ALO)

The ant lion optimizer (ALO) (Mirjalili 2015a) is a biologically inspired optimizer that models how antlion bugs behave in nature. Their hunting behavior is very unique and interesting. Antlions build a sharp-edged cope-shaped trap to trap ants. Afterward, they hide and wait for ants/insects to be trapped. The sharp edges prevent the trapped insects from escaping and easily falling to the bottom of the

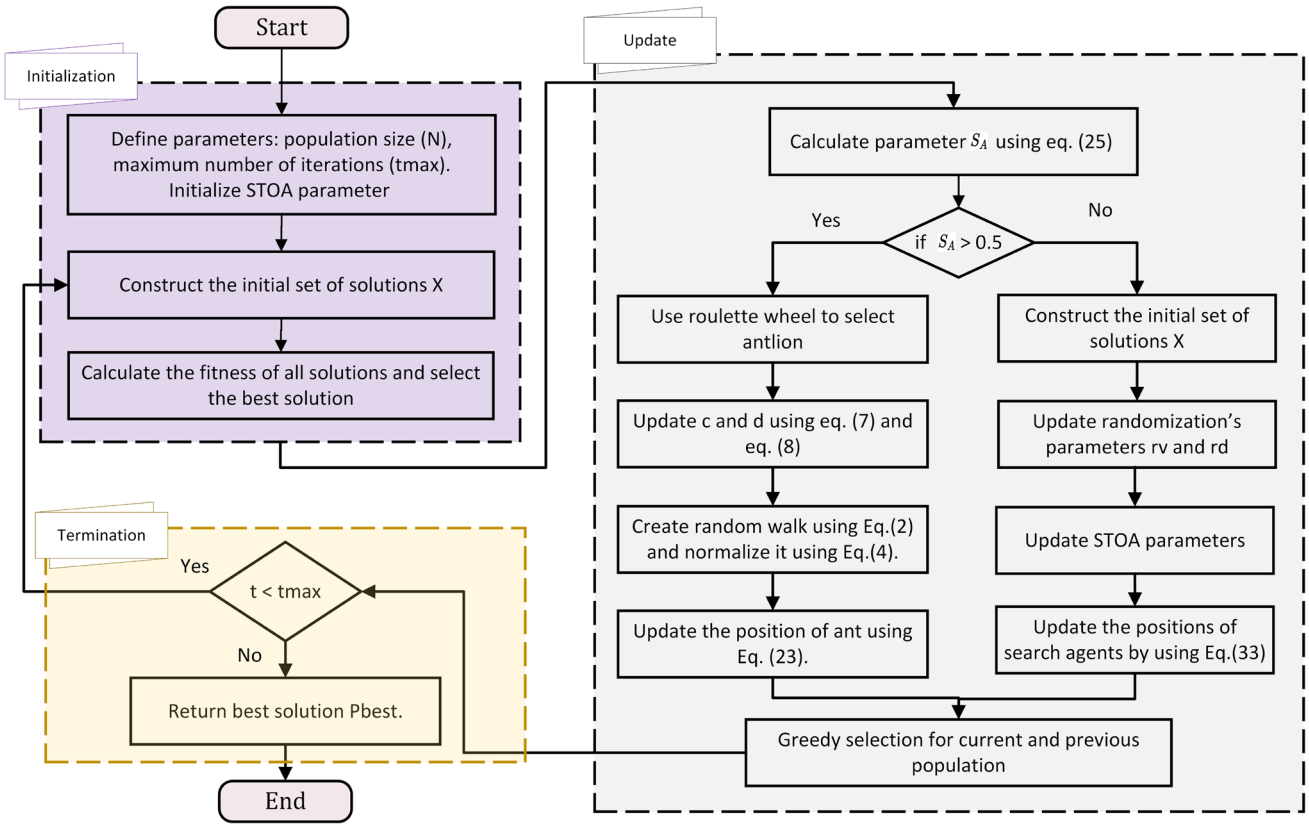


Fig. 1 The flowchart of proposed ST-AL method

Table 3 Parameters setting of competitive algorithms

Algorithms	Parameters setting
PSO	wMax = 0.9, wMin = 0.1 (Default)
GWO	a decreases linearly from 2 to 0
HHO	beta = 1.5 (Default)
MFO	$b = 1$ and a decreases linearly from -1 to -2 (Default)
MPA	FADs = 0.2, $P = 0.5$, $\beta = 1.5$
ALO	–
STOA	$C_f = 2, C_B \in [0, 0.5], u, v = 1$

trap. Finally, the antlion consumes the insect, throws the leftovers, and prepare the trap for the next hunt. It has been noticed that the higher the antlions' hunger, the bigger trap they dig.

The ALO tries to solve optimization problems taking into consideration the random walk of ants to search for food, the trap building process, the ants' entrapment, catching targets, and the traps re-building. These random movements over the search space are modelled using cumulative sum function and a random function applied through different iterations. Such random behavior force to find the global optimization solution. An objective function is employed during optimization to show the model's goal to efficiently maximize the resources' utilization. ALO also assumes that the antlions hide in the search space which is restricted using the min–max algorithm. The ALO algorithm simulates the main five steps of the hunting: (a) random walk of ants, (b) traps' building, (c) ants entrapment, (d) sliding preys towards ant-lions, (e) grasping ants and traps re-building as follows:

Table 4 CEC2020 benchmark functions description with fitness score (Fi*)

No.	Function description	Fi*
Unimodal function		
F1	Shifted and rotated Bent Cigar function	100
Multimodal shifted and rotated functions		
F2	Shifted and rotated Schwefel's function	1100
F3	Shifted and rotated Lunacek bi-Rastrigin function	700
F4	Expanded Rosenbrock's plus Griewangk's function	1900
Hybrid functions		
F5	Hybrid function 1 ($N = 3$)	1700
F6	Hybrid function 2 ($N = 4$)	1600
F7	Hybrid function 3 ($N = 5$)	2100
Composition functions		
F8	Composition function 1 ($N = 3$)	2200
F9	Composition function 2 ($N = 4$)	2400
F10	Composition function 3 ($N = 5$)	2500

1. *Random walk of ants*. At first, population of ants in the search landscape is initialized and their positions are stored in a vectors as follows:

$$Ant_i = \left[A_{i,1}, A_{i,2}, \ldots, A_{i,d}\right] \tag{1}$$

where Ant_i is the ith ant, $A_{i,d}$ is the position of the ith ant in the dth dimension. The position of each ant in each dimension is updated using a random walk at each step of the optimization.

$$x(t) = \left[0, \text{cumsum}(2r(t_1 - 1)), \text{cumsum}(2r(t_2 - 1)), \ldots \right. \\ \left. \ldots, \text{cumsum}(2r(t_n - 1))\right] \tag{2}$$

$$r(t) = \begin{cases} 1 & \text{rand} > 0.5 \\ 0 & \text{rand} \leq 0.5 \end{cases} \tag{3}$$

where cumsum is the cumulative sum, n is the maximum number of iterations, t is a step/iteration of random walk, $r(t)$ is a stochastic function, rand is a random number uniformly distributed between [0, 1]. The random walks of ants are restricted within the boundaries of the finite search space using min–max normalization as follows:

$$x_i^t = \frac{\left(x_i^t - a_i\right) \times \left(d_i - c_i^t\right)}{\left(d_i^t - a_i\right)} + c_i \tag{4}$$

where a_i is the minimum of random walk of variable i, b_i is the maximum of random walk of variable i, c_i^t is the minimum of variable i at iteration t, d_i^t is the maximum of variable i at iteration t.

2. *Ants entrapment* The trap is built using a roulette wheel to select antlions depending on their fitness. ALO assumes that ants are trapped in only one selected antlion trap. Antlions construct larger pits based on their fitness value to catch insects/targets. The impact of antlions on the movement of ants is modelled as follows:

Table 5 Statistical results of ST-AL versus other metaheuristics on CEC2020 benchmark functions $D = 10$

Function	Measures	PSO	GWO	HHO	MFO	MPA	ALO	STOA	ST-AL
F1	Best	100.826	1512.383	97240.3	137.9296	100.6736	102.9929	195,835.9	**100**
	Worst	4936.871	3.28E+08	494,935.9	**1.42E+09**	12,734.87	12,356.59	7.83E+08	**100**
	Mean	1676.327	17,447,639	255,933.6	88,533,559	6574.686	2190.297	1.92E+08	**100**
	Std	1434.075	73,156,598	96,382.65	3.15E+08	4574.322	3082.667	2.49E+08	**0**
F2	Best	1342.317	**1108.298**	1571.98	1204.36	1116.859	1419.401	1568.529	1115.36157
	Worst	2300.557	2228.222	2383.975	2641.262	1837.691	2260.379	2200.763	**1490.702**
	Mean	1747.702	1543.441	1949.897	2078.011	1451.227	1838.022	1902.805	**1273.42804**
	Std	290.0656	258.5856	240.7425	361.8118	187.4426	228.5664	174.2322	**100.063159**
F3	Best	716.9109	718.62	746.467	717.3873	712.7313	719.7333	720.7049	**712.142278**
	Worst	741.807	747.2259	821.3927	753.9382	**721.5218**	758.4034	781.3539	725.842949
	Mean	726.4704	728.6477	783.6794	733.184	716.7477	740.1851	748.8464	**716.590131**
	Std	6.466513	8.060786	20.39673	9.844801	**2.781191**	11.96006	12.72752	2.89892234
F4	Best	1900.574	1900.51	1902.453	1900.8	**1900.203**	1900.506	1900.992	1900.3827
	Worst	1901.93	1903.119	1911.646	**1960.669**	1901.11	1902.07	1906.199	1901.58745
	Mean	1901.09	1901.502	1906.15	1905.063	**1900.583**	1901.16	1903.086	1900.83892
	Std	0.376943	0.775824	2.21147	13.15188	**0.22342**	0.487497	1.351234	0.30449676
F5	Best	2095.018	2870.067	2831.059	4948.331	2093.779	2232.061	4228.143	**1700**
	Worst	10,125.04	346,987.8	101,688.2	154,875.6	12,427.31	13,213.51	36,828.5	**1704.97479**
	Mean	5003.836	39,555.51	38,316.17	28,684.16	6790.615	6622.3	11,604.37	**1700.92772**
	Std	2693.21	105,163.4	39,034.46	34,291.65	3464.369	3417.503	7102.483	**1.08564299**
F6	Best	1719.861	1601.307	1602.998	1601.538	**1600.049**	1601.493	1650.931	1600.90527
	Worst	2057.216	1854.354	2016.48	1970.371	**1601.141**	1955.882	1944.735	1613.98899
	Mean	1817.027	1732.626	1761.277	1809.43	**1600.532**	1739.135	1749.094	1602.74265
	Std	86.93557	89.95497	90.97596	122.1287	**0.334115**	93.09908	51.94684	3.66477493
F7	Best	2101.296	2493.738	2648.412	2810.218	2166.543	2723.26	2873.355	**2100.04555**
	Worst	2817.753	16,101.66	28,462.2	47,264.84	2566.775	23,473.42	17,294.41	**2100.86278**
	Mean	2317.767	8872.406	9094.916	12,352.6	2316.229	9380.929	7568.5	**2100.43022**
	Std	168.9784	4527.194	8610.609	11,798.61	106.2277	7138.869	5117.499	**0.27717999**
F8	Best	2219.911	2301.407	2305.98	2300.795	2220.291	2221.889	2223.742	**2215.83467**
	Worst	2303.192	2320.279	2327.855	2550.84	**2300.842**	3238.041	4007.94	2301.17164
	Mean	2297.5	2307.001	2315.111	2325.678	**2293.224**	2341.933	2865.334	2296.25875
	Std	18.27148	**5.52889**	5.701036	55.53402	21.8177	212.3494	641.6428	18.9342359
F9	Best	**2500**	2722.003	2500.918	2749.388	2500.002	**2500**	2737.275	**2500**
	Worst	2780.04	2764.887	2923.949	2792.254	2748.407	2779.331	2776.437	**2600**
	Mean	2715.859	2740.05	2809.683	2767.348	2515.143	2728.255	2751.611	**2505**
	Std	93.77332	10.24654	84.81685	12.14551	23.94911	78.75515	**9.611748**	22.3606798
F10	Best	2897.836	2898.292	**2717.372**	2898.384	2897.94	2897.757	2898.754	2897.74287
	Worst	2949.504	3024.415	3024.521	2978.48	2949.906	2951.041	3024.674	**2897.74287**
	Mean	2935.808	2933.8	2924.869	2939.017	2927.937	2928.899	2933.125	**2897.74287**
	Std	19.37901	27.88147	59.08303	27.01963	23.94911	23.03498	26.03043	**9.3312E−13**

$$c_i^t = \text{Antlion}_j^t + c^t \tag{5}$$

$$d_i^t = \text{Antlion}_j^t + d^t \tag{6}$$

where c^t is the minimum of all variables at iteration t, d^t is the vector of maximum of all variables at iteration t, c_i^t is the minimum of all variables for ant i, $Antlion_j^t$ determines the location of antlion j at iteration t.

3. *Building Trap* The pit/entrap is built using a roulette wheel to choose antlions based on their fitness. ALO assumes that ants are entrapped in only one particular

Table 6 Statistical results of ST-AL versus other metaheuristics on CEC2020 benchmark functions $D = 20$

Function	Measures	PSO	GWO	HHO	MFO	MPA	ALO	STOA	ST-AL
F1	Best	149.0675	9169.883	1498054	9938.403	429.5645	**121.0709**	1.18E+09	137.039303
	Worst	6076.59	2.71E+09	4004630	8.22E+09	11415.45	**4758.852**	5.14E+09	11797.7427
	Mean	1910.775	6.04E+08	2773719	2.31E+09	4802.742	**1412.314**	3.22E+09	5067.13731
	Std	1920.911	7.43E+08	716443.9	2.38E+09	3628.375	**1215.155**	1.4E+09	4540.32487
F2	Best	1468.85	1677.007	1854.2	2173.439	1798.769	2788.219	2516.219	**1244.57253**
	Worst	3631.735	3485.399	3391.141	4795.42	3548.621	4006.389	3716.063	**1929.16945**
	Mean	2684.459	2428.321	2459.846	3076.589	2535.579	3427.677	3146.28	**1603.27468**
	Std	680.6819	488.4162	452.0179	795.4393	568.8346	415.6953	364.3663	**184.673132**
F3	Best	749.1345	753.8559	812.0693	751.3783	728.6476	792.3075	835.9468	**728.414091**
	Worst	805.7087	810.3985	936.0164	1074.894	**749.5001**	896.3071	939.8215	761.631671
	Mean	772.9227	770.7944	899.7641	837.214	**737.09**	834.092	873.2247	742.661136
	Std	17.23145	15.32764	36.46678	101.9656	**6.773366**	35.01951	28.97488	9.85814681
F4	Best	1901.425	1902.745	1914.023	1905.823	**1901.422**	1903.54	1919.641	1902.28299
	Worst	1905.133	1950.59	1931.732	22,523.83	**1902.599**	1907.686	2671.891	1904.99868
	Mean	1902.894	1916.953	1921.658	7707.941	**1902.007**	1904.849	2098.6	1903.37977
	Std	1.012403	15.32228	6.07432	7939.735	**0.36723**	1.562298	236.5469	0.80847691
F5	Best	4576.084	45,510.86	43,245.39	4897.889	1734.605	12,376.85	33,292.3	**1718.88259**
	Worst	151,338.3	1,563,541	492,134.1	5,162,088	2203.713	253,429	486,787.9	**1982.4812**
	Mean	58,998.96	693,320.8	237,767.3	888,330	1919.761	111,336.6	269,894.8	**1849.60918**
	Std	42,310.16	550,428.2	132,439.6	1,429,444	113.9693	70,004.62	165,169.1	**73.0268531**
F6	Best	1602.271	1654.922	1894.872	1764.33	1602.207	1668.604	1822.076	**1602.05363**
	Worst	2313.125	1958.078	2312.84	2338.96	1720.192	2674.598	2489.339	**1613.58199**
	Mean	1935.191	1864.35	2090.694	2031.482	1612.784	2242.032	2065.039	**1605.42989**
	Std	177.8016	81.43428	123.0286	167.3889	23.71376	301.072	208.3496	**3.73490572**
F7	Best	3893.95	32,677.23	12,044.76	24,277.9	2102.306	4249.516	14,138.56	**2101.59872**
	Worst	161,212.1	249,374.2	455,508	1,198,037	2280.503	316,732.5	228,195.7	**2234.34465**
	Mean	28,323	135,952.6	115,152.6	299,970.3	2165.784	73,687.46	90,663.87	**2136.33553**
	Std	44,782.82	69,725.22	122,523.3	391,212.4	58.78263	106,741.1	73,679.27	**45.0973721**
F8	Best	**2300**	2310.62	2311.782	2301.171	2300.004	**2300**	2524.337	2300.01914
	Worst	5613.677	4339.147	6005.18	5964.614	**2313.576**	4763.698	6296.554	5138.35431
	Mean	3016.317	2819.914	3159.088	4041.776	**2303.359**	2692.017	5322.392	3145.49128
	Std	1304.161	753.7906	1536.643	1621.218	**3.892559**	915.7107	981.0725	1255.01527
F9	Best	2852.118	2821.487	2965.218	2837.652	2810.925	2852.385	2847.739	**2810.61914**
	Worst	3007.874	2916.136	3353.093	2945.812	**2835.372**	2927.431	2906.623	2841.71589
	Mean	2901.121	2857.805	3172.954	2885.383	2823.501	2887.074	2868.628	**2821.48244**
	Std	48.29315	29.57164	113.3423	25.77292	9.00791	25.40788	18.77983	**8.18603102**
F10	Best	2910.509	2924.751	2925.859	2910.67	**2910.198**	2914.069	2956.969	2910.22865
	Worst	3000.437	3181.11	3002.534	3169.652	2914.002	2999.653	3181.348	**2913.82748**
	Mean	2949.443	3027.891	2975.48	2961.343	2913.231	2970.505	3024.626	**2913.14968**
	Std	33.8739	81.12437	21.64809	75.05806	1.412285	23.04355	57.18155	**1.3075439**

The bold values highlight the largest, or the highest value received per row of the data. It signifies that which algorithm is producing best result under same external conditions on a specific dataset

antlion trap. Antlions construct larger pits based on their fitness value to catch insects/targets.

4. *Sliding ants towards antlion* When an ant trapped, antlion starts to throw sands towards the center of the pit so the prey slides down into the trap. In this step, the hyper-sphere radius of the random walk is adaptively minimized using Eqs. (7) and (8).

Table 7 ST-AL vs other meta-heuristics algorithms for CEC2020 ($D = 10$) in terms of P values of the Wilcoxon ranksum test

ST-AL vs.	PSO	GWO	HHO	MFO	MPA	ALO	STOA
F1	8.007E−09	8.00655E−09	8.00655E−09	7.7176E−09	8.00655E−09	8.00655E−09	8.00655E−09
F2	2.563E−07	0.000247061	6.79562E−08	7.94795E−07	0.000686822	9.17277E−08	6.79562E−08
F3	1.576E−06	3.41558E−07	6.79562E−08	3.93881E−07	0.797197419	1.23464E−07	7.89803E−08
F4	0.0179386	0.003638826	6.79562E−08	4.54008E−06	0.00604033	0.033717669	2.95975E−07
F5	6.796E−08	6.79562E−08	6.79562E−08	6.79562E−08	6.79562E−08	6.79562E−08	6.79562E−08
F6	6.796E−08	1.57567E−06	9.17277E−08	6.0148E−07	1.43085E−07	4.53897E−07	6.79562E−08
F7	6.796E−08	6.79562E−08	6.79562E−08	6.79562E−08	6.79562E−08	6.79562E−08	6.79562E−08
F8	9.748E−06	6.79562E−08	6.79562E−08	1.65708E−07	0.010581211	1.25052E−05	0.000115901
F9	1.103E−07	3.37272E−08	3.94662E−08	3.37272E−08	3.37272E−08	4.61473E−08	3.37272E−08
F10	8.007E−09	8.00655E−09	2.10246E−07	8.00655E−09	8.00655E−09	8.00655E−09	8.00655E−09

Table 8 ST-AL vs other meta-heuristics algorithms for CEC2020 ($D = 20$) in terms of P values of the Wilcoxon ranksum test

ST-AL vs.	PSO	GWO	HHO	MFO	MPA	ALO	STOA
F1	0.0970911	9.0734E−06	3.39182E−06	6.13704E−06	0.966914777	0.042110617	3.39182E−06
F2	0.5067205	0.839859973	0.750831884	0.053097957	6.00576E−05	0.001353941	0.008615558
F3	0.0002462	0.000123346	3.65846E−05	0.000123346	0.193930852	3.65846E−05	3.65846E−05
F4	0.1939309	0.000384202	3.65846E−05	3.65846E−05	9.73457E−05	0.010193105	3.65846E−05
F5	0.0120228	0.035089116	0.544370146	0.068964333	3.65846E−05	0.174853307	0.370844333
F6	0.000592	3.65846E−05	3.65846E−05	3.65846E−05	3.65846E−05	3.65846E−05	3.65846E−05
F7	0.0086156	0.112351198	0.707453968	0.140955219	3.65846E−05	0.260236203	0.839859973
F8	9.75E−03	0.193930852	0.126022122	0.014137969	0.795012172	0.088533772	0.000155796
F9	3.658E−05	0.000592042	3.65846E−05	4.69487E−05	0.623604884	3.65846E−05	3.65846E−05
F10	0.0035498	3.65846E−05	3.65846E−05	0.000384202	0.019373319	3.65846E−05	3.65846E−05

$$c^t = \frac{c^t}{I} \tag{7}$$

$$d^t = \frac{d^t}{I} \tag{8}$$

where I is a ratio, c^t is the minimum of all variables at iteration t, d^t is the vector including the maximum of all variables at iteration t.

5. *Catching prey and rebuilding the pit*. In this step, the caught ant is assumed to be fitter than the associated antlion. Afterwards, antlion updates its position to the latest position of the caught prey to increase its chance of catching a new one. This step is mathematically modeled using Eq. (9).

$$\text{Antlion}_j^t = \text{Ant}_i^t \quad \text{if} \quad f(\text{Ant}_i^t) > f(\text{Antlion}_j^t) \tag{9}$$

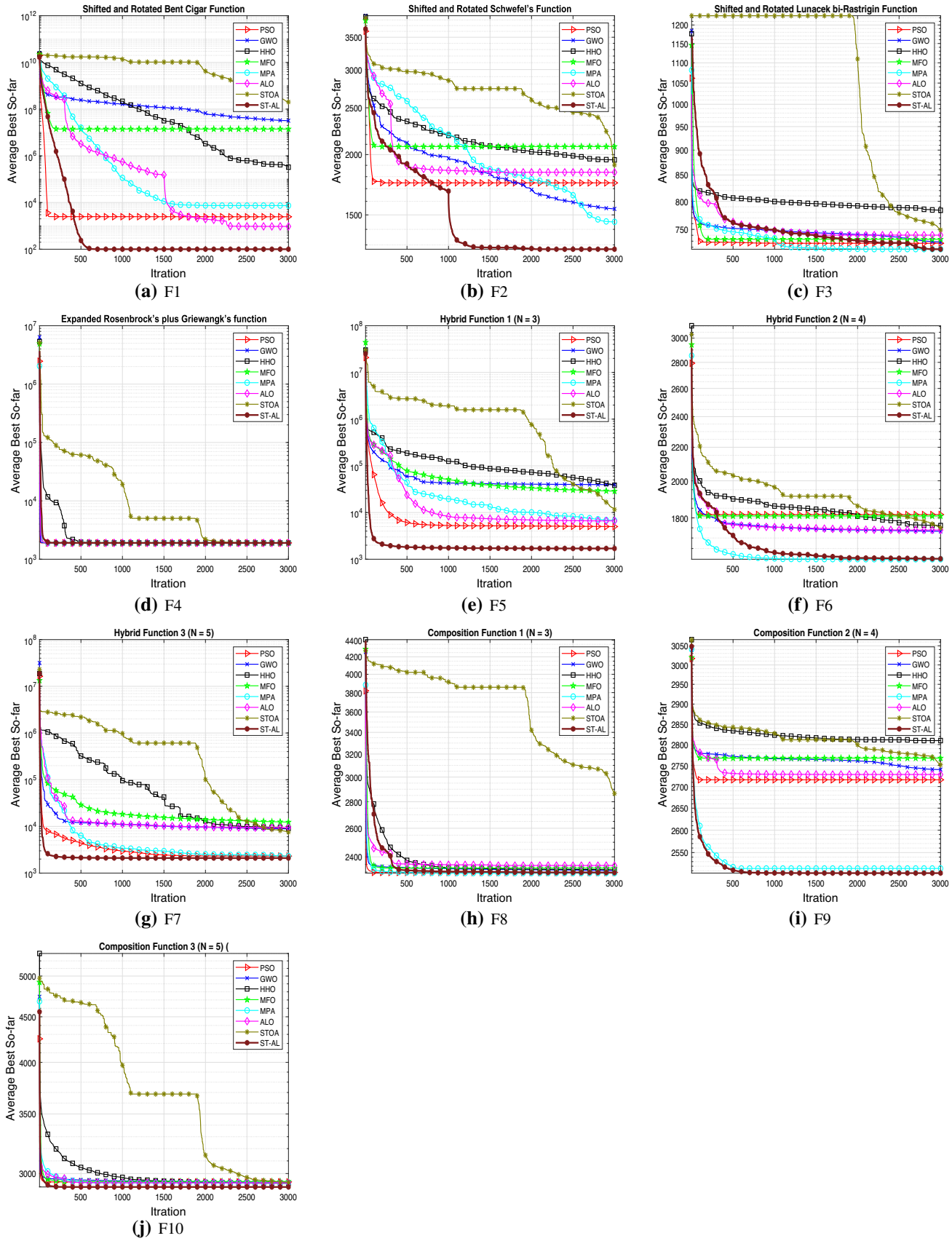
where t is the present iteration, Antlion_j^t is the location of antlion j at iteration t. Ant_i^t is the location of ant i at iteration t.

6. *Elitism* ALO maintains the best obtained solution throughout the optimization process. The fittest obtained antlion in each iteration is considered as an elite. The elite antlion is able to affect the movements of all ants during iterations. Thus, every ant randomly walks around a selected antlion by the roulette wheel and the elite as follows:

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{10}$$

where R_A^t is the random walk around the selected antlion by roulette wheel at iteration t, R_E^t is the random walk around the elite at iteration t, Ant_i^t is the position of ant i at iteration t.

The pseudocode of ALO algorithm is given in Algorithm 1.



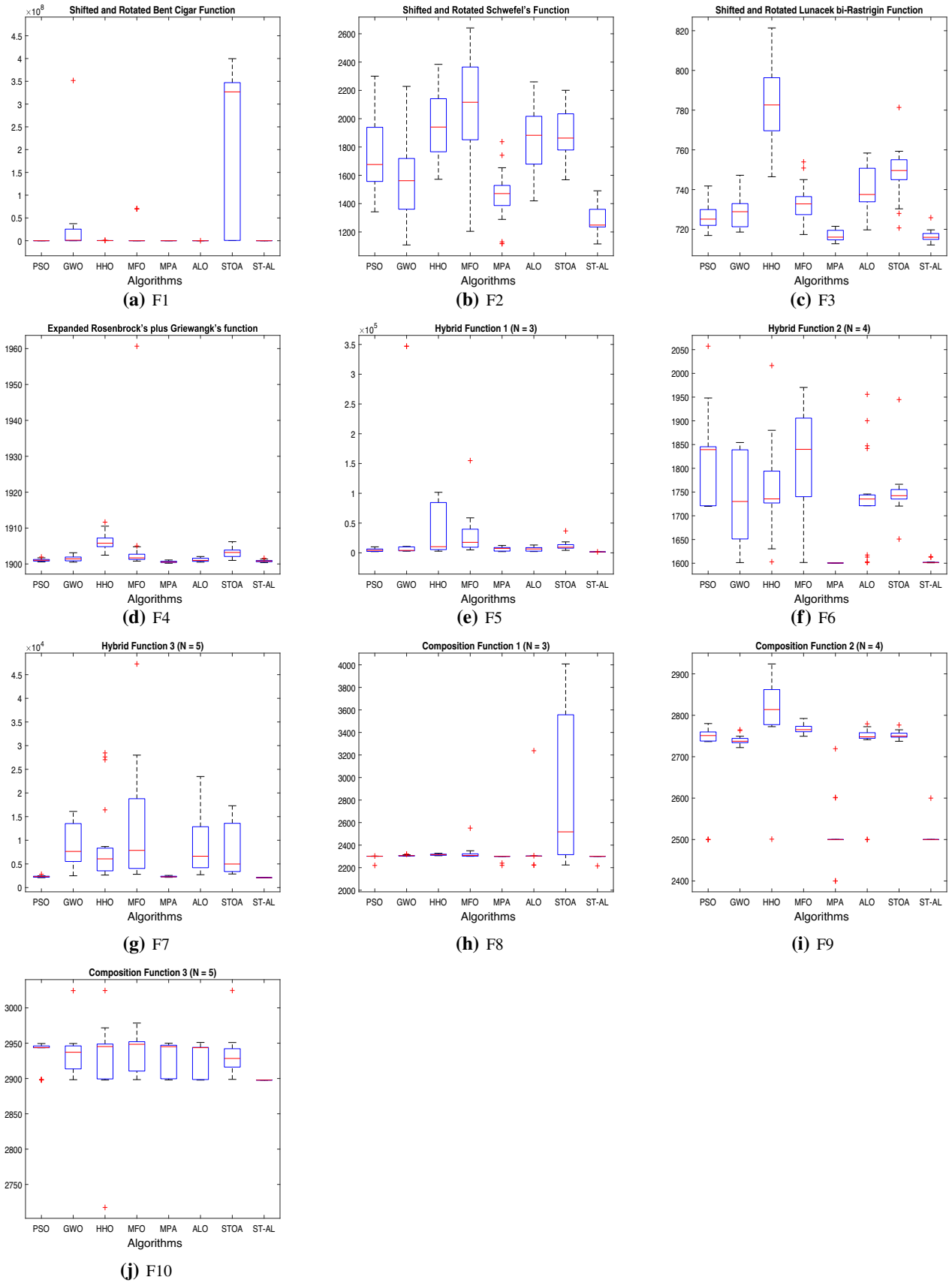
(a) F1 **(b)** F2 **(c)** F3

(d) F4 **(e)** F5 **(f)** F6

(g) F7 **(h)** F8 **(i)** F9

(j) F10

Fig. 2 Convergence curve for ST-AL against other competitors—CEC2020 of $D = 10$



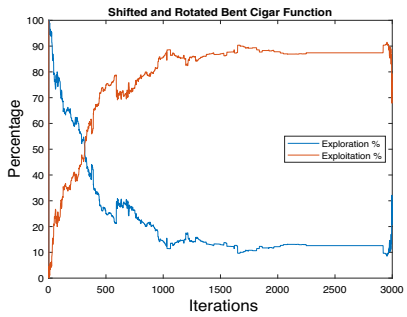
(a) F1 **(b)** F2 **(c)** F3

(d) F4 **(e)** F5 **(f)** F6

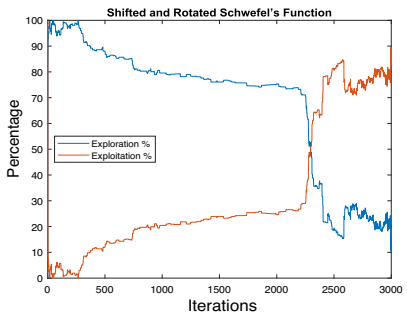
(g) F7 **(h)** F8 **(i)** F9

(j) F10

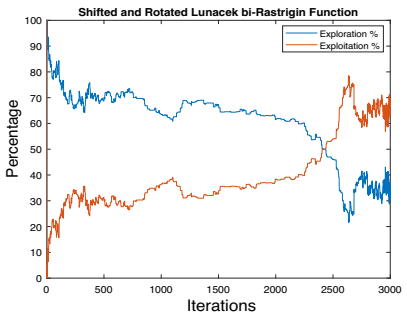
Fig. 3 Boxplot for ST-AL against other competitors—CEC2020 of $D = 10$



(a) F1

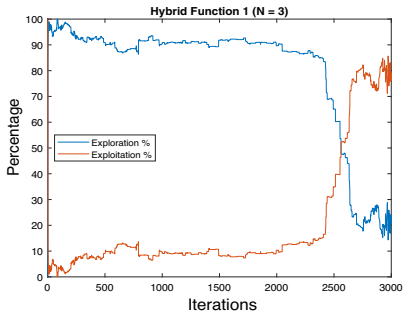


(b) F2

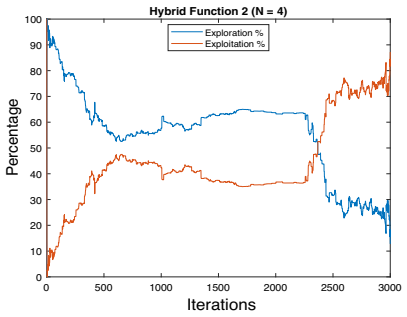


(c) F3

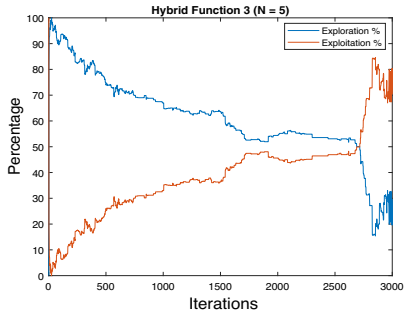
(d) F4



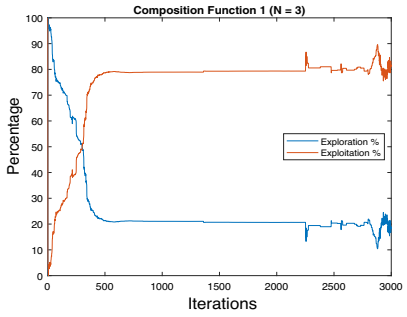
(e) F5



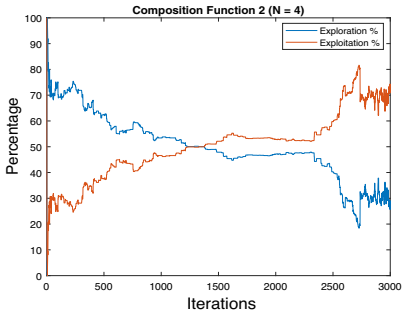
(f) F6



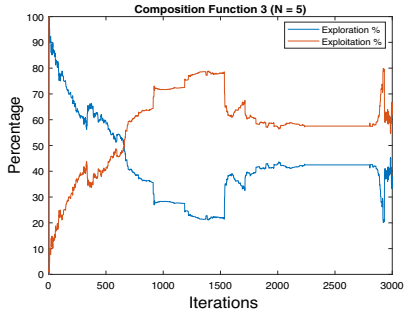
(g) F7



(h) F8



(i) F9



(j) F10

Fig. 4 Exploration and exploitation curves of ST-AL method—CEC2020

Algorithm 1 Pseudo-code of ALO algorithm

Input: Total number of ants and antlions, fitness function, maximum number of iterations
Output: the best antlion and fitness value
Initialize the random positions of ants and antlions within boundaries
Calculate the fitness value for the antlions
Sort fitness values and select the best antlion (elite)
while (end criteria is not satisfied) **do**
  for $i = 1 : N$ **do**
    Use roulette wheel to select antlion A and build trap
    Update c and d using using Eq.(7) and Eq.(8).
    Create random walk of Ant_i using Eq.(2) and normalize it using Eq.(4).
    Update the position of ant using the equation Using Eq. (10).
  end for
  Evaluate the fitness value of all ants
  Replace antlion with its corresponding ant $\text{Antlion}_j^t = \text{Ant}_i^t$ if $f(\text{Ant}_i^t) > f(\text{Antlion}_j^t)$
  Update elite if an antlion becomes fitter than the selected elite
end while
Return elite.

3.2 Sooty tern optimization algorithm (STOA)

The STOA algorithm is inspired of the sooty sea tern sea birds' attacking/exploitation and migration/exploration behavior (Dhiman and Kaur 2019). Sooty terns eat earthworms, insects, fish, reptiles, etc. They live in groups with a unique migration and attacking behavior. During the migration (exploration), sooty terns migrate in groups to search and locate the richest. During their attacks, sooty terns fly to locate their targets. They usually preserve distance between every two birds to avoid collision. Within the group, birds travel in the direction of the fittest/best survival bird and update their positions accordingly. STOA models the mathematical notation of both exploitation and exploration in a search space as follows:

Table 9 UCI benchmark datasets

Datasets	Features	Samples	Classes	Category
Low dimensional datasets				
Exactly	13	1000	2	Biology
Exactly2	13	1000	2	Biology
Lymphography	18	148	2	Biology
SpectEW	22	267	2	Biology
CongressEW	16	435	2	Politics
IonosphereEW	34	351	2	Electromagnetic
Vote	16	300	2	Politics
WineEW	13	178	3	Chemistry
BreastEW	30	569	2	Biology
PenglungEW	325	73	2	Biology
SonarEW	208	60	2	Biology
HeartEW	13	270	2	Biology
M-of-n	13	1000	2	Biology
Zoo	16	101	6	Artificial
High dimensional datasets				
base_Brain_T21	10,367	50	4	Biology
base_leuk1	11,225	72	3	Biology

1. *Migration/exploration behavior* tries to find the distance between search agents which satisfy three conditions:
   (a) *The Collision avoidance* between each agent and its neighbors

$$\overrightarrow{C_{st}} = S_A \times \overrightarrow{P_{st}}(z) \tag{11}$$

   where $\overrightarrow{C_{st}}$ is the position of agent that ensures it does not collide with its neighbors, $\overrightarrow{P_{st}}(z)$ is the position of search agent (st) in iteration z, S_A is the search agent's movement in search space.

$$S_A = C_f \times \left(Z \times \left(\frac{C_f}{\text{Max\_iterations}}\right)\right) \tag{12}$$

   where $Z = 1, 2, \ldots,$ Max_iterations, C_f is a controlling variable to adapt the S_A that is decreased linearly from C_f to 0. C_f is initialized to 2.
   (b) *Converge towards the best neighbor's direction* After avoiding collision, search agents move towards the fittest neighbor's direction.

$$\overrightarrow{M_{st}} = C_B \times \left(\overrightarrow{P_{bst}}(z) - \overrightarrow{P_{st}}(z)\right) \tag{13}$$

   where $\overrightarrow{M_{st}}$ is the locations of agent $\overrightarrow{P_{st}}$ in the direction of the best agent $\overrightarrow{P_{bst}}$, C_B is random variable responsible for better exploration computed as follows:

$$C_B = 0.5 \times R_{\text{and}} \tag{14}$$

Table 10 Mean and standard deviation of fitness values of proposed ST-AL and other competitors

Dataset	Measures	PSO	GWO	HHO	MFO	MPA	ALO	STOA	ST-AL
Low dimensional datasets									
Exactly	Mean	0.07978	0.01844	0.01315	0.00837	**0.0046**	0.21749	0.150503269	**0.004615**
	STD	0.11564	0.061812	0.020916	0.016775	1.78E−18	0.122488	0.149887136	**1.78E−18**
Exactly2	Mean	0.20945	0.20929	0.20441	0.20182	0.2021	0.21128	0.209408269	**0.197752**
	STD	0.008543	0.006572	0.006572	0.006826	0.005229	0.004863	0.007001712	**0.0041013**
Lymphography	Mean	0.06994	0.05381	0.05508	0.05278	0.05174	0.08931	0.070816442	**0.039141**
	STD	0.027083	0.024516	0.018863	0.019668	0.015403	0.034336	0.021362251	**0.00728**
SpectEW	Mean	0.09338	0.08129	0.0789	0.07798	**0.0756**	0.09786	0.09380303	**0.075629**
	STD	0.015687	0.008927	0.006517	0.00568	0.000102	0.018575	0.015655563	**0.0001016**
CongressEW	Mean	0.02545	0.02049	0.01846	0.01757	0.01622	0.0241	0.024602371	**0.014986**
	STD	0.006262	0.006182	0.005452	0.004817	0.00434	0.007149	0.007492906	**0.0033024**
IonosphereEW	Mean	0.03514	0.01966	0.03119	0.02389	**0.0174**	0.04109	0.03392937	0.018379
	STD	0.015475	0.006458	0.0099	0.007166	**0.004996**	0.015815	0.013644524	0.0056419
Vote	Mean	0.00733	0.00331	0.00328	0.00341	0.00328	0.00704	0.00585	**0.003156**
	STD	0.00836	0.000458	0.000344	0.000378	0.000569	0.006098	0.005509949	**0.0001398**
WineEW	Mean	0.00839	0.00318	0.00219	0.00223	**0.0015**	0.00705	0.006317308	0.001692
	STD	0.01481	0.006476	0.000943	0.00093	**4.45E−19**	0.010643	0.010596922	0.0005353
BreastEW	Mean	0.04517	0.03872	0.04261	0.03857	0.03942	0.04565	0.047908772	**0.037335**
	STD	0.004681	0.006352	0.003666	0.003019	0.004017	0.005502	0.005164334	**0.0030016**
PenglungEW	Mean	0.15457	0.12199	0.15683	0.14906	**0.0703**	0.15284	0.075696648	0.146121
	STD	0.016478	0.03174	0.01945	0.007585	0.052681	0.027583	0.0635006	**0.0033567**
SonarEW	Mean	0.04266	**0.0078**	0.049	0.01249	0.01132	0.07172	0.05314881	0.009796
	STD	0.017705	0.012851	0.017735	0.011743	**0.011648**	0.026809	0.024743855	0.0134368
HeartEW	Mean	0.20116	0.19378	0.19485	0.19161	0.19088	0.20746	0.198858974	**0.188513**
	STD	0.011911	0.008557	0.00889	0.00801	0.007765	0.013599	0.008202481	**0.0057219**
M-of-n	Mean	0.00995	0.00977	0.00469	**0.0046**	**0.0046**	0.02921	0.030564423	**0.004615**
	STD	0.014041	0.023072	0.000237	1.78E−18	1.78E−18	0.025765	0.046349329	**1.78E−18**
Zoo	Mean	0.00576	0.00219	0.00197	0.00247	**0.0016**	0.00384	0.0023125	0.002156
	STD	0.011085	0.000555	0.000583	0.000474	**0.00043**	0.001394	0.000577113	0.000516
High dimensional datasets									
base_Brain_T21	Mean	0.367752	0.19934	0.209182	0.351421	0.25074	0.198704	0.198003858	**0.104568**
	STD	0.046541	0.000293	0.015412	0.069951	0.007727	0.000334	0.070133166	**1.364E−06**
base_leuk1	Mean	0.07072	0.00117	0.00039	0.0709	0.00328	0.00013	1.24722E−05	**1.16E−05**
	STD	0.09338	6.87E−05	0.000173	0.093281	0.000229	1.64E−05	1.13389E−05	**1.008E−05**

The bold values highlight the largest, or the highest value received per row of the data. It signifies that which algorithm is producing best result under same external conditions on a specific dataset

where R_{and} is a random number in the range [0, 1]

(c) *Update relevant to the fittest search agent* Eventually, search agent updates its position according to the best agent.

$$\overrightarrow{D_{st}} = \overrightarrow{C_{st}} + \overrightarrow{M_{st}} \tag{15}$$

where $\overrightarrow{D_{st}}$ is the gap between the search agent and fittest agent.

2. *Attacking/exploitation behavior* Sooty terns adjust their velocity and angle during attack. While attacking their targets/preys, they use wings in a flapping way to increase their altitude as follows:

$$x' = R_{\text{adius}} \times \sin(i) \tag{16}$$

$$z' = R_{\text{adius}} \times \cos(i) \tag{17}$$

$$z' = R_{\text{adius}} \times i \tag{18}$$

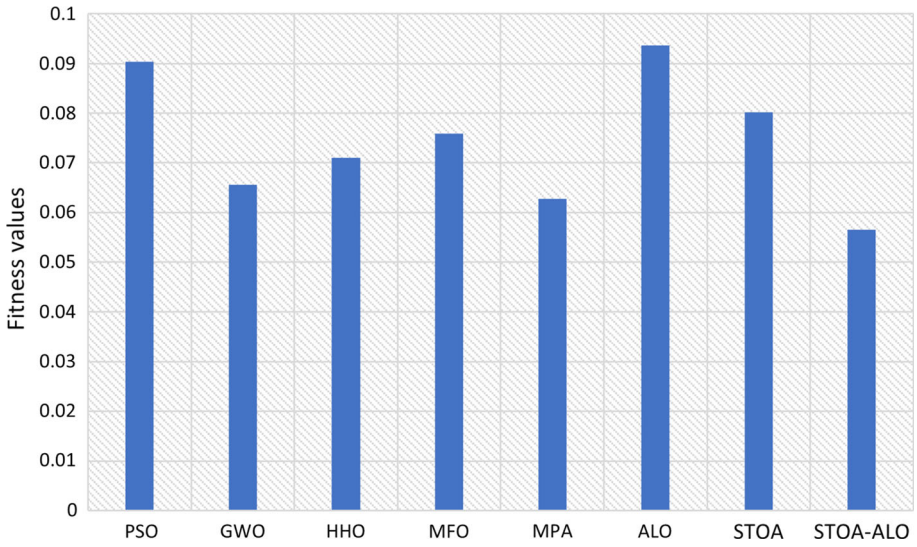


Fig. 5 Average of fitness value over all the tested datasets

$$r = u \times e^{kv} \tag{19}$$

where R_{adius} is the radius of each turn of the spiral, i is a variable within the range of $[0 \leq k \leq 2\pi]$, u and v are constants that identify the shape of the spiral shape, e is the natural logarithm base. Eventually, the updated position of the agent is computed as follows:

$$\overrightarrow{P_{st}}(z) = \left(\overrightarrow{D_{st}} \times (x' + y' + z')\right) + \overrightarrow{P_{bst}}(z) \tag{20}$$

where $\overrightarrow{P_{st}}(z)$ calculates the updated position of other agents and saves the best optimal solution.

The pseudocode of STOA algorithm is shown in Algorithm 2.

4 Proposed hybrid ST-AL optimization algorithm

This section explains the structure of the proposed ST-AL method, which combines both STOA and ALO algorithms. In the proposed ST-AL method, the performance of STOA algorithm is improved using four strategies as follows:

- Control randomization parameters
- New exploration phase based on ALO algorithm
- Enhance STOA exploitation phase
- Greedy selection.

Strategy 1: Control randomization parameters Randomization is a main side of metaheuristic algorithm that plays

Algorithm 2 Pseudo-code of STOA algorithm.

Input: Population $\overrightarrow{P_{st}}$, total number of agents (N), maximum number of iterations (Max_iterations)
Output: Best search agent $\overrightarrow{P_{bst}}$
Determine the initial parameters S_A and C_B.
Evaluate the objective function (fitness) of each search agent
return the best search agent $\overrightarrow{P_{st}}(z)$
while ($z <$ Max_iterations) **do**
  for $i = 1 : N$ **do**
    Update the positions of search agents by using using Eq.(20).
  end for
  Update the parameters S_A and C_B
  Calculate the fitness value of each search agent
  Update $\overrightarrow{P_{bst}}$ if there is a better solution than previous optimal solution
  $z = z + 1$
end while
Return $\overrightarrow{P_{bst}}$.

Table 11 Mean and standard deviation of accuracy of proposed ST-AL and other competitors

Dataset	Measures	PSO	GWO	HHO	MFO	MPA	ALO	STOA	ST-AL
Low dimensional datasets									
Exactly	Mean	0.9245	0.986	0.9915	0.99625	**1**	0.787	0.85225	**1**
	Std	0.116313	0.06261	0.020844	0.016771	**0**	0.122565	0.151818	**0**
Exactly2	Mean	0.79325	0.79275	0.7985	0.8015	0.80075	0.7935	0.79275	**0.806**
	Std	0.009072	0.00734	0.00709	0.007626	0.0052	0.005643	0.007691	**0.00447214**
Lymphography	Mean	0.934015	0.949126	0.947927	0.951092	0.950969	0.914975	0.931302	**0.96436371**
	Std	0.02784	0.025094	0.019986	0.020126	0.016373	0.034311	0.021824	**0.00733628**
SpectEW	Mean	0.909259	0.92037	0.923148	0.924074	**0.925926**	0.905556	0.907407	**0.92592593**
	Std	0.015782	0.008707	0.006784	0.0057	**0**	0.017924	0.015896	**0**
CongressEW	Mean	0.977011	0.981609	0.983908	0.985057	0.986207	0.977586	0.977011	**0.98735632**
	Std	0.006459	0.006876	0.005777	0.005404	0.004717	0.007889	0.008339	**0.00353786**
IonosphereEW	Mean	0.966901	0.98169	0.970423	0.978169	**0.983803**	0.960563	0.966901	0.98309859
	Std	0.015344	0.006622	0.010115	0.007189	**0.00516**	0.015564	0.013917	0.00578016
Vote	Mean	0.996667	**1**	**1**	**1**	**1**	0.9975	0.9975	**1**
	Std	0.008719	**0**	**0**	**0**	**0**	0.006106	0.006106	**0**
WineEW	Mean	0.994405	0.998611	**1**	**1**	**1**	0.995833	0.995833	**1**
	Std	0.014597	0.006211	**0**	**0**	**0**	0.010176	0.010176	**0**
BreastEW	Mean	0.958333	0.963596	0.960526	0.964912	0.963158	0.959211	0.953947	**0.96578947**
	Std	0.004826	0.006536	0.0045	0.002846	0.004589	0.005884	0.005602	**0.00269994**
PenglungEW	Mean	0.847894	0.878223	0.844795	0.853663	**0.930192**	0.848583	0.924227	0.85604396
	Std	0.016622	0.032078	0.019739	0.007723	0.052993	0.02691	0.063877	**0.00338235**
SonarEW	Mean	0.960714	**0.994048**	0.953571	0.990476	0.990476	0.930952	0.947619	0.99285714
	Std	0.017742	0.013098	0.018075	**0.011967**	**0.011967**	0.026648	0.02515	0.01360097
HeartEW	Mean	0.800926	0.807407	0.806481	0.810185	0.810185	0.794444	0.801852	**0.81296296**
	Std	0.011827	0.009308	0.009452	0.008227	0.008227	0.0133	0.008707	**0.00569988**
M-of-n	Mean	0.995	0.99475	**1**	**1**	**1**	0.97675	0.97375	**1**
	Std	0.013669	0.023479	**0**	**0**	**0**	0.024935	0.046901	**0**
Zoo	Mean	0.9975	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	Std	0.01118	**0**	**0**	**0**	**0**	**0**	**0**	**0**
High dimensional datasets									
base_Brain_T21	Mean	0.633333	0.8	0.788889	0.65	0.75	0.8	0.8	**0.89444444**
	Std	0.04714	**0**	0.015713	0.070711	0.070711	**0**	**0**	0.00785674
base_leuk1	Mean	0.933333	**1**	**1**	0.933333	**1**	**1**	**1**	**1**
	Std	0.094281	**0**	**0**	0.094281	**0**	**0**	**0**	**0**

The bold values highlight the largest, or the highest value received per row of the data. It signifies that which algorithm is producing best result under same external conditions on a specific dataset

a vital role in balance between exploration–exploitation phases, so control parameters of randomization must be more accurate to give promising results. In the proposed hybrid ST-AL method, two parameters are presented that integrated together for this task. The first parameter controls the value of randomization, called randomization value (*rv*), is given by:

$$rv = 2 \times \text{rand} - 1 \tag{21}$$

The second proposed parameter in the control randomization, is called, randomization direction (*rd*). The value of *rd* parameter is $+1$ or -1, that gives an opportunity to change the direction of search agents in the given search space and subsequently result in good scan of the interested region. Combination of (*rv*) and (*rd*) leads to excellent scan of a given search space and decrease probability of falling in local optimum and convergence premature.

Strategy 2: New exploration phase based on ALO algorithm The exploration phase is characterized by a large motion step to enable the algorithm to cover the given search space. The STOA's exploration phase doesn't satisfy this side because the process of agent updating position

is based only on the location of the best agent in the swarm, and the agent's current position. Therefore, it fails to move with large steps in different areas in the given search space. In the ST-AL method, the exploration phase is based on the ALO algorithm.

The ALO algorithm has a good exploration strategy that is based on the random selection of antlions (sorted agents) and random walks of ant (agent) around them. Every ant randomly walks around a antlion selected by the roulette wheel and the elite (the fittest antlion) simultaneously as follows:

$$P_{st}(t) = \frac{R_A(t) + R_E(t)}{2} \tag{22}$$

where $P_{st}(t)$ is the position of search agent in yteration t, $R_A(t)$ is the random walk around the agent selected by the roulette wheel, and $R_E(t)$ is the random walk around the best agent.

Accordingly, the exploration phase of the proposed ST-AL method follows the same strategy to get R_A and R_E, and then update the agent's position as follow:

$$P_{st}(t) = P_{st}(t) + S_A \times rv \times rd \times \left| P_{st}(t) - \frac{R_A(t) + R_E(t)}{2} \right| \tag{23}$$

$P_{st}(t)$ is added in the update equation to guide the agents with the current position and not diverse in false positions. The control randomization parameters, and S_A is an absolute value that used to balance between exploration and exploitation and its value change gradually with time.

Strategy 3: Enhance STOA exploitation phase In the proposed hybrid ST-AL method, the exploitation stage is similar to the exploitation stage strategy in the original STOA algorithm, except for position update equation where some settings were made to enhance its efficiency. The exploitation phase of ST-AL can be summarized as follow:

1. Collision avoidance. Here the agents that does not collide are defined by:

$$C_{st} = S_A \times P_{st}(t) \tag{24}$$

where C_{st} is the position of the search agent which does not collide with other search agents. S_A indicates the movement of search agent in a given search space to avoid the collision avoidance between its neighboring search agents, and it is calculated as follow:

$$S_A = C_f - \left(t \times \left(\frac{C_f}{T} \right) \right) \tag{25}$$

where C_f is controlling variable between [0, 2] (= 2 in this study), T is the total time of iterations, and t is the current iteration.

2. Converge towards the direction of best neighbor's. The agents converge to the best agent after collision avoidance that mathematically defined by:

$$M_{st} = C_B \times (P_{bst}(t) - P_{st}(t)) \tag{26}$$

where M_{st} is the different locations of search agent, and C_B is the random number given by:

$$C_B = 0.5 \times R_{\text{and}} \tag{27}$$

where R_{and} is the random number [0, 1]

3. Update corresponding to best search agent. D_{st} is the gap between the current agent and the best agent and is given as follows:

$$D_{st} = C_{st} + M_{st} \tag{28}$$

4. The spiral behavior in the air. After migration, the agents move in spiral motion as follow:

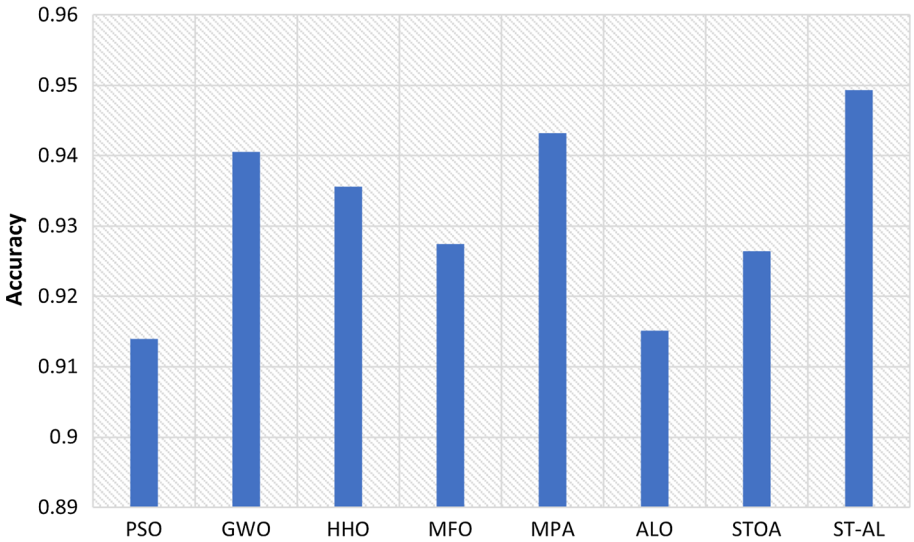


Fig. 6 Average of accuracy measure overall the tested datasets

$$x' = R_{\text{adius}} \times \sin(i) \tag{29}$$

$$y' = R_{\text{adius}} \times \cos(i) \tag{30}$$

$$z' = R_{\text{adius}} \times i \tag{31}$$

$$r = u \times e^{kv} \tag{32}$$

where x', y' and y' represents the spiral behavior in the air, R_{adius} is the radius of each turn of the spiral, i is a variable $[0, 2\pi]$, u and v are constants of the spiral motion shape.

5. Update position of search agent. Finally, the agents update their position as follow:

$$P_{st}(t) = S_A \times rv \times rd \times |D_{st} \times (x' + y' + z')| + P_{best}(t) \tag{33}$$

where $P_{bst}(t)$ is the best agent in swarm. The absolute value is taken to remove ineffective randomization and avoid deviation from global optimum. *rv* and *rd* are the parameters of control randomization given by Eq. (21).

Strategy 4: greedy selection The greedy selection is applied between the generated population and current population to reject the poor generated population and avoid divergence of the algorithm from existing promising regions. The flowchart of the proposed ST-AL method is shown in Fig. 1, and pseudocode is given in Algorithm 3.

5 Performance evaluation of the proposed ST-AL

In this section, the proposed ST-AL algorithm's quality is evaluated by conducting two experiments: (1) employing it as a global optimization method to discover the optimal value of the CEC2020 benchmark functions, and (2) applying the proposed algorithm as an feature selection approach. The parameter settings of each experiment are given in Table 2. Comparisons between ST-AL and popular algorithms such as PSO (Kennedy and Eberhart 1995), GWO (Mirjalili et al. 2014), HHO (Heidari et al. 2019), MFO (Mirjalili 2015b), MPA (Faramarzi et al. 2020), and the original ALO and STOA are made. The settings for each algorithm are specified in Table 3. As demonstrated by authors in Arcuri and Fraser (2013), setting algorithm parameters to their default values is a reasonable and acceptable practice. All findings were calculated using Matlab 2021b on an Intel Corei7 computer with a 2.67G CPU and 8.00G of RAM running 64-bit OS.

5.1 Performance measures

Several statistical measurements are applied to evaluate the performance of the proposed ST-AL method.

1. *Mean*: represents the rate of the optimization algorithm and hence has been applied it *M* times and is defined as follow:

Algorithm 3 Steps of proposed ST-AL method

Input: parameters of the ST_AL: C_f, R_{adius} , u and v, number of population (N), and maximum number of iteration (t_{max})
Output: return best solution P_{bst}
Initialize the parameters such as N solutions,C_f, R_{adius} , u and v
Construct the initial set of solutions X
for $t = 1 : t_{max}$ **do**
    Evaluate the fitness value of $X_i, i = 1, 2, ..., N$
    sort the solutions according to fitness values and return best solution P_{bst}.
    Update S_A using by Eq.(25)
    for $i = 1 : N$ **do**
        if $S_A > 0.5$ **then**
            Use roulette wheel to select antlion
            Update c and d using using Eq.(7) and Eq.(8).
            Create random walk of Ant_i using Eq.(2) and normalize it using Eq.(4).
            Update the position of ant using the equation using Eq. (23).
        else
            Update randomization's parameters rv and rd
            Update STOA parameters C_{st}, M_{st}, C_B,D_{st} using Eqs. 25,27,28,29 respectively
            Update the positions of search agents by using Eq.(33)
        end if
    end for
    Greedy selection for current and previous population
    $t = t + 1$.
end for
Return best solution P_{bst}.

Table 12 Mean and standard deviation of selected features of proposed ST-AL and other competitors

Dataset	Measures	PSO	GWO	HHO	MFO	MPA	ALO	STOA	ST-AL
Low dimensional datasets									
Exactly	Mean	6.55	5.95	6.15	6.05	6	8.6	6	**5.5**
	Std	1.234376	0.223607	0.366348	0.223607	**0**	1.846761	1.051315	**0**
Exactly2	Mean	6.2	**5.35**	6.4	6.9	6.3	8.9	5.5	7.4
	Std	2.214783	1.460894	1.231174	1.333772	1.218282	3.447348	1.468977	**0.99472292**
Lymphography	Mean	8.3	6.2	6.35	7.85	5.75	9.25	**5.05**	6.95
	Std	1.688974	1.542384	2.870448	1.460894	2.336777	2.149051	1.90498	**0.88704121**
SpectEW	Mean	7.8	5.4	6.2	6.2	5.05	9.6	5.05	**4.7**
	Std	2.627787	1.535544	1.576138	1.935812	**0.223607**	3.299123	0.571241	**0.2236068**
CongressEW	Mean	4.3	3.65	4.05	4.45	4.1	3.05	**2.95**	3.95
	Std	1.838191	1.348488	0.887041	1.669384	0.91191	1.190975	1.356272	**0.39403446**
IonosphereEW	Mean	8.05	5.2	6.5	7.75	4.6	6.95	**3.95**	5.6
	Std	1.700619	1.105013	1.90567	2.099499	**0.940325**	2.874113	0.998683	1.18765581
Vote	Mean	6.45	5.3	5.25	5.45	5.25	7.3	5.4	**5.05**
	Std	1.234376	0.732695	0.55012	0.604805	0.910465	1.174286	1.187656	**0.2236068**
WineEW	Mean	3.7	2.35	2.85	2.9	**2**	3.8	2.85	2.2
	Std	1.380313	0.875094	1.225819	1.209611	**0**	1.609184	1.225819	0.69585237
BreastEW	Mean	11.75	8.05	10.6	11.5	8.85	15.8	**6.95**	10.4
	Std	1.860249	**1.761429**	2.85436	2.259483	2.680829	4.490927	1.90498	1.90290636
PenglungEW	Mean	129.6	46.6	103.4	135.95	40.3	95.45	**22.15**	117.15
	Std	10.09116	7.081481	30.62919	**7.067159**	20.67569	50.03207	16.05345	10.3123585
SonarEW	Mean	22.6	11.35	18.2	18.35	11.35	20.2	**7.75**	16.35
	Std	3.424371	2.814904	5.596992	3.572924	2.560325	8.230879	2.468219	**2.0072238**
HeartEW	Mean	5.3	4.05	4.25	4.8	3.85	5.15	**3.5**	4.35
	Std	1.341641	1.145931	1.118034	1.151658	**0.67082**	1.663066	0.82717	0.87509398
M-of-n	Mean	6.5	5.95	6.1	6	6	8.05	6	**5.95**
	Std	0.82717	0.223607	0.307794	0	0	1.538112	0.686333	**0**
Zoo	Mean	5.25	3.5	3.15	3.95	3.45	6.15	3.7	**2.5**
	Std	1.743409	0.888523	0.933302	0.759155	0.825578	2.230766	0.923381	**0.6882472**
High dimensional datasets									
base_Brain_T21	Mean	4926	1389.5	189	5101.5	3359	729.5	**40**	70.5
	Std	132.9361	303.3488	149.9066	54.44722	134.3503	345.7752	**1.414214**	53.0330086
base_leuk1	Mean	5296	1314.5	439	5504	3676.5	149	14	**13**
	Std	46.66905	77.07464	193.7473	63.63961	256.6798	18.38478	12.72792	**11.3137085**

The bold values highlight the largest, or the highest value received per row of the data. It signifies that which algorithm is producing best result under same external conditions on a specific dataset

$$Mean = \frac{1}{M}\sum_{i=1}^{M} g_*^i \tag{34}$$

where g_*^i, indicates to the optimal solution that generated at the i-th operation.

2. *Best* represents the minimum (or best) fitness function value achieved in M independent operations by the optimization algorithm. The calculation of which is given in Eq. (35)

$$Best = \min_{i=1}^{M} g_*^i \tag{35}$$

3. *Worst* is calculated as the maximum (worst value) fitness function value generated in M independent operations by the optimization algorithm and is shown by Eq. (36)

$$Worst = \max_{i=1}^{M} g_*^i \tag{36}$$

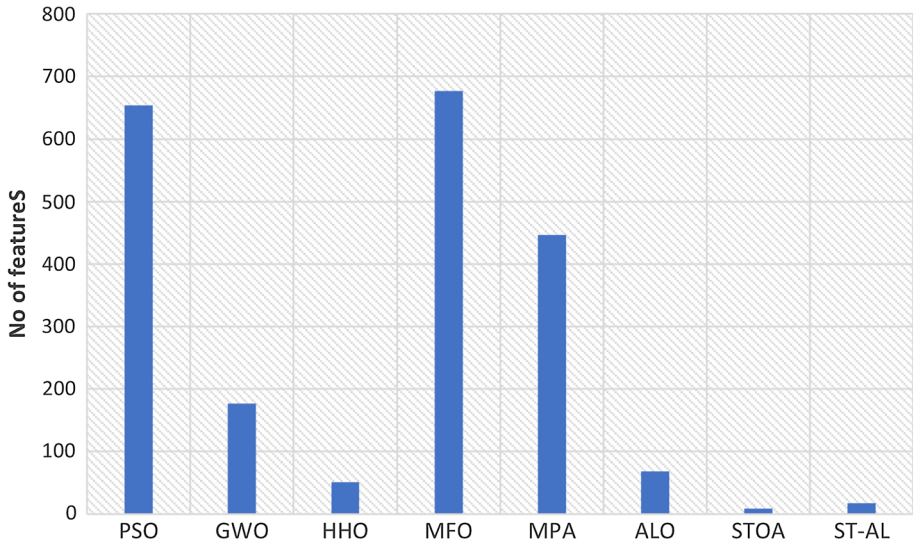


Fig. 7 Average of the selected features for all methods

4. *Standard deviation* (*Std*) defines the optimization algorithm robustness and stability as follow; (1) if Std is small value this mean that the optimization algorithm always converges to the same solution, on the other hand if the Std is large value means the optimization algorithm close to random results, as shown in Eq. (37):

$$Std = \sqrt{\frac{1}{M-1}\sum (g_*^i - Mean)^2} \tag{37}$$

In the evaluation of the feature selection experiment, additional measures were used:

5. *Average classification accuracy* (*Avg_Acc*): The rate at which data is correctly classified is reflected in the accuracy metric. There are 30 runs of each method in this study, hence the *Avg_Acc* metric is determined as follows:

$$Avg\_Acc = \frac{1}{M}\sum_{j=1}^{M}\frac{1}{N}\sum_{i=1}^{N} Match(C_i, L_i) \tag{38}$$

where N indicates the instances number, C_i represents the classifier output label for instance i, L_i is the reference class label for instance i, and *Match* is a function equal 1 when the two input labels are the same and 0 otherwise.

6. *Average selection size* (*Avg_Selec*) represents the average size of the selected features as shown in Eq. (39).

$$Avg\_Selec = \frac{1}{M}\sum_{i=1}^{M}\frac{size(g_*^i)}{N_t} \tag{39}$$

where N_t indicates to entire features number within the original dataset.

7. *Average CPU time* (*Avg_Time*) calculates the average of CPU time (in milliseconds) for each algorithm

$$Avg\_Time = \frac{1}{M}\sum_{k=1}^{M} T_*^k \tag{40}$$

Note that the STD is calculated also for all other measures: accuracy, time and number of selected features. The best value for each measure is highlighted in bold.

5.2 Experimental series 1: CEC'2020 test suite

A standard set of benchmarks listed in IEEE Congress on Evolutionary Computation (CEC2020) (Mohamed et al. 2020a) is utilized to evaluate the proposed ST-AL algorithm's performance. Many metaheuristic algorithms' performance has been studied using these functions (Houssein et al. 2021; Mohamed et al. 2020b). As shown in Table 4, the CEC'2020 benchmark functions include ten test functions that fall into four categories: unimodal, multimodal, hybrid, and composition functions. All algorithms were run 30 times independently to ensure a fair benchmarking comparison and demonstrate the robustness of the proposed ST-AL in comparison to a collection of competing algorithms that run over 3000 iterations with 30 search agents. The maximum number of function evaluations is 90,000 (number of iterations multiplied by the total number of search agents). This study employs a variety of measurements. These metrics include the minimum, maximum, mean, standard deviation (SD) of fitness values, and Wilcoxon rank-sum P values.

Table 13 Mean and standard of computational time

Dataset	Measures	PSO	GWO	HHO	MFO	MPA	ALO	STOA	ST-AL
Low dimensional datasets									
Exactly	Mean	32.8723	31.6798	211.2935	31.7957	61.63897	37.28005	**28.94931**	31.9099857
	Std	3.032551	1.001774	636.6944	1.030127	1.101994	5.634429	1.749094	**0.87864113**
Exactly2	Mean	37.16188	34.18057	79.96094	37.55719	64.96702	45.63201	**31.33725**	33.8151437
	Std	14.05005	11.3701	27.09949	11.79444	9.186924	12.96779	3.334212	**2.3838203**
Lymphography	Mean	3.474046	3.207923	6.842256	3.219168	6.21433	3.508497	**3.03656**	3.61528481
	Std	0.280491	0.42511	0.838324	0.359309	0.569301	**0.207392**	0.260024	0.31703988
SpectEW	Mean	4.226438	3.831835	8.336655	3.985982	7.620279	4.630855	**3.524859**	4.21792923
	Std	0.642847	0.418268	1.093323	0.236051	0.77147	0.459457	**0.153677**	0.24410306
CongressEW	Mean	15.27402	13.40267	30.58325	15.20167	29.79356	12.86989	**12.55053**	15.5711764
	Std	6.697039	**5.776717**	13.69468	6.981718	17.44587	6.296027	6.95009	6.79696608
IonosphereEW	Mean	6.598056	5.487657	11.6526	6.608258	10.96986	6.568173	**4.982914**	7.00948847
	Std	0.555354	**0.244741**	1.009391	0.503626	0.928145	1.030192	0.608827	0.46341451
Vote	Mean	4.738496	4.310376	9.237963	4.295258	8.238886	4.805906	**4.183892**	4.75236379
	Std	0.590074	**0.306373**	0.821992	0.326176	0.671268	0.667188	0.479461	0.52278501
WineEW	Mean	4.750004	4.40867	9.818607	4.769744	9.27589	4.676446	**4.125152**	4.77080706
	Std	3.000175	2.778946	5.910559	3.020723	6.00408	2.759085	**2.542107**	2.82644831
BreastEW	Mean	26.15425	22.20005	58.69249	25.1453	41.39065	31.3347	21.60136	**16.6740296**
	Std	11.97868	10.50037	26.87943	11.33414	16.91998	16.22258	**9.543448**	28.4742736
PenglungEW	Mean	5.605969	4.242601	9.185003	5.693376	7.97668	9.668026	**3.804831**	9.93484848
	Std	0.522903	0.66523	0.926931	**0.373641**	0.518197	0.795727	0.485212	0.614507
SonarEW	Mean	4.883084	3.879995	8.983516	4.737767	7.633989	5.521879	**3.48735**	5.4487059
	Std	0.283649	0.240564	1.091669	0.219798	0.266605	0.625994	0.179738	**0.15488951**
HeartEW	Mean	6.754364	6.358299	13.70715	6.527611	12.62662	6.608504	6.562438	**5.68706242**
	Std	3.773762	3.499975	7.583404	3.559205	7.062762	3.422229	3.781703	**3.35431769**
M-of-n	Mean	47.54074	48.2617	98.49601	43.40045	80.40572	48.055	42.05009	**40.5838176**
	Std	29.45491	29.78828	58.70728	26.45444	42.65164	28.21701	26.41757	**24.6063505**
Zoo	Mean	8.716726	8.318151	18.57037	8.686236	16.98621	9.200277	**8.032926**	8.89933061
	Std	2.852059	**2.762736**	6.198441	2.86613	5.639374	3.017719	2.832504	3.04760754
High dimensional datasets									
base_Brain_T21	Mean	34.80719	12.07017	9.554337	35.16061	471.0573	130.2078	**5.314859**	9.6954273
	Std	0.807416	0.285376	1.04293	**0.229544**	13.4967	0.399301	0.508654	0.75388769
base_leuk1	Mean	69.63425	23.08034	26.75476	68.62912	602.1905	155.3611	**7.911707**	18.8797114
	Std	**0.129373**	0.559691	8.192249	0.71024	1.55028	0.246642	0.195569	1.26942008

The bold values highlight the largest, or the highest value received per row of the data. It signifies that which algorithm is producing best result under same external conditions on a specific dataset

5.2.1 Statistical results analysis

Using ten CEC'2020 benchmark functions with a dimension of solution (Dim = 10), Table 5 illustrates the best, worst, mean, and standard deviation (STD) of the fitness scores achieved by all competing algorithms. It is underlined in bold the best results for each assessment criterion. The results revealed that the proposed ST-AL algorithm outperforms the other metaheuristics in terms of the mean fitness value. As a result of its greater performance on seven test functions (F1, F2, F3, F5, F7, F9, and F10), whereas MPA fared best on only three functions (F4, F6 and F8). Results also show a similar trend in terms of standard deviation, with ST-AL outperforming other algorithms on five benchmark functions while MPA surpassed them on four (F3, F4, F6, and F9). Comparing the ST-AL algorithm to other algorithms using the best and worst fitness metrics, it has showed competitive performance. When the best and worst fitness metrics are taken into consideration, the ST-AL algorithm has demonstrated competitive performance when compared to other algorithms. This shows ST-AL algorithm's search capabilities

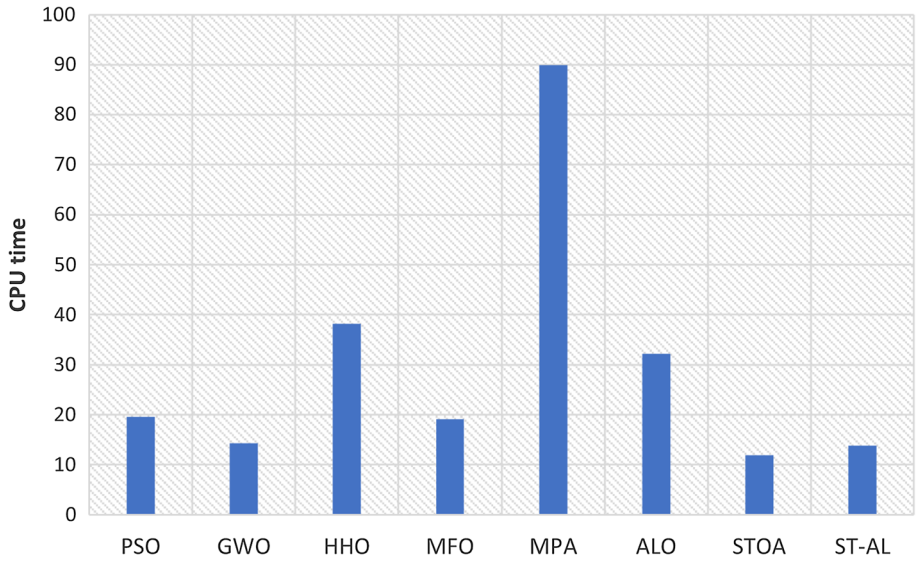


Fig. 8 Average of the computational time for all methods

and stability. Even though other algorithms outperform ST-AL in particular test cases, ST-AL remains the clear winner in terms of overall performance measurements. This experiment's results show that the ST-AL outperforms the other seven metaheuristics in solving the vast majority of these optimization issues.

Additionally, the performance of ST-AL and other metaheuristic algorithms is evaluated on CEC2020 at dimension 20, as described in Table 6. This table shows that the ST-AL outperforms the competitor algorithms in six functions (F2, F5, F6, F7, F9, and F10). On the other hand, MPA outperforms in three functions (F3, F4, and F8). ALO gives better performance at only function F1.

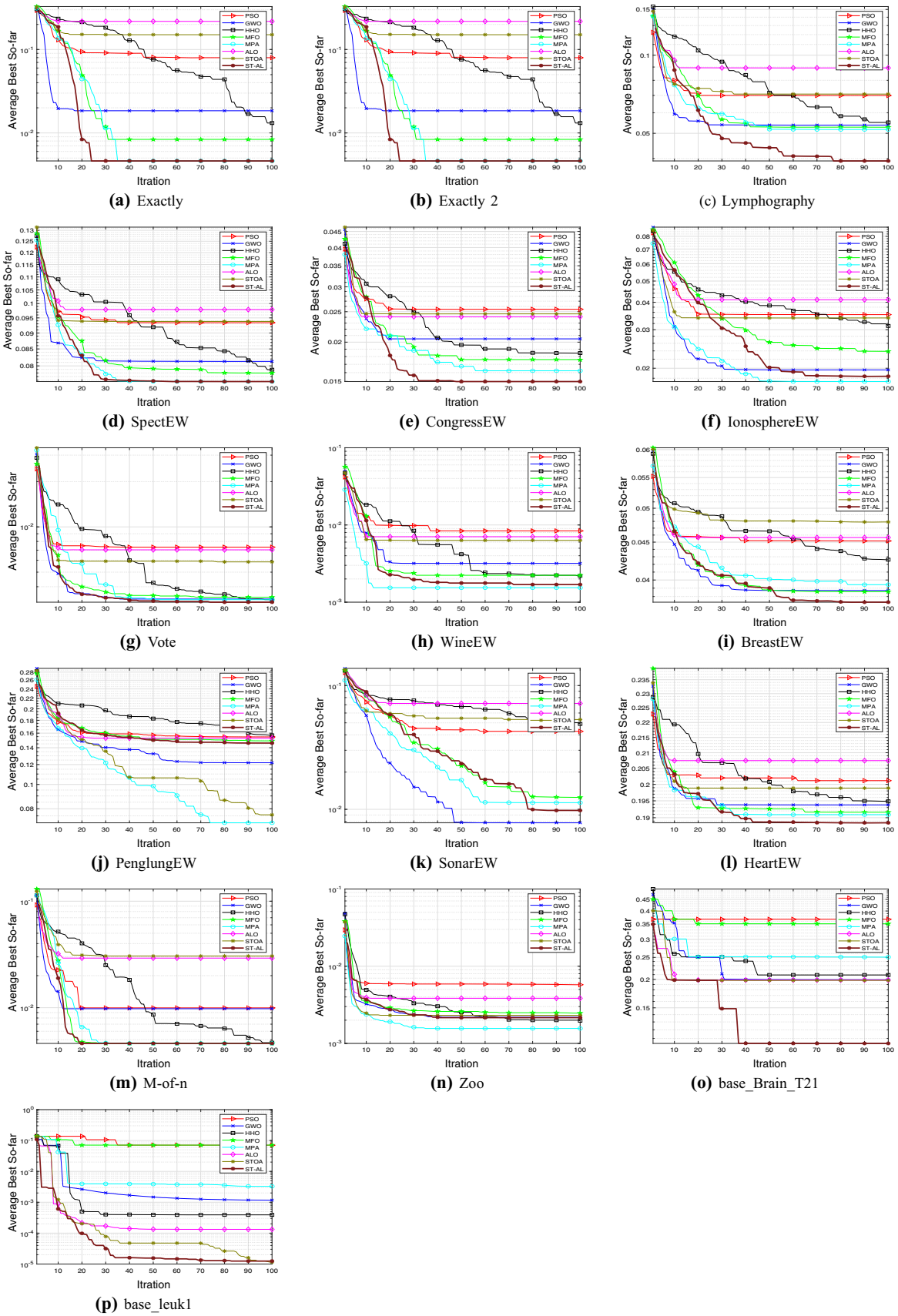
The Wilcoxon sum test is one of the non-parametric tests that may be used to examine the outcomes of paired algorithms. The zero hypothesis indicates that the results of a comparison approach are indistinguishable. Comparative methods may be distinguished by rank, according to this alternative viewpoint. Estimated Wilcoxon rankings for five levels of significance (P) are produced. If $P > 0.05$, the hypothesis is verified as zero, whereas if $P < 0.05$, it is accepted. The P Wilcoxon mean-sum fitness findings are shown in Tables 7 and 8. The proposed ST-AL algorithm is noticeably different from all other algorithms. As a result, the proposed ST-AL algorithm has seen tremendous development.

5.2.2 Convergence behavior analysis

The examination of convergence is a critical step in determining the stability of the optimization algorithms. Therefore, a comparative analysis of the proposed ST-AL and its competitors is conducted. The convergence curves of the proposed ST-AL algorithm and other competitor algorithms for the CEC'2020 functions are shown in Fig. 2. The figure clearly shows that the ST-AL algorithm has reached a stable point for all functions. Over a small number of function evaluations, the proposed ST-AL algorithm achieves the lowest average of global solutions faster than other compared algorithms for all CECs benchmark functions. In applications requiring fast computation, like online optimization problems, this fast convergence of the ST-AL algorithm may be easily characterized as a potential optimization approach.

5.2.3 Boxplot behavior analysis

The boxplot analysis can display the characteristics of the data distribution. This class of functions has multiple local minima; hence to better comprehend the distribution of results. Boxplots are graphical representations of data distributions in three primary quartiles as upper, lower, and middle quartiles. The algorithm's lowest and largest data points represent the minimum and maximum, which constitute the whisker's edges. The ends of the rectangles define the lower and upper quartiles. There is a strong agreement between the data points if the boxplot is narrow. CEC'20 ten functions boxplot for dim = 10 is shown in Fig. 3. The proposed ST-AL algorithm's boxplots in most functions are quite narrow and have the lowest values compared to the distributions of the other algorithms. As a result, the suggested ST-AL algorithm outperforms the other competitor algorithms on the vast majority of the test functions under consideration.



(a) Exactly **(b)** Exactly 2 (c) Lymphography

(d) SpectEW **(e)** CongressEW **(f)** IonosphereEW

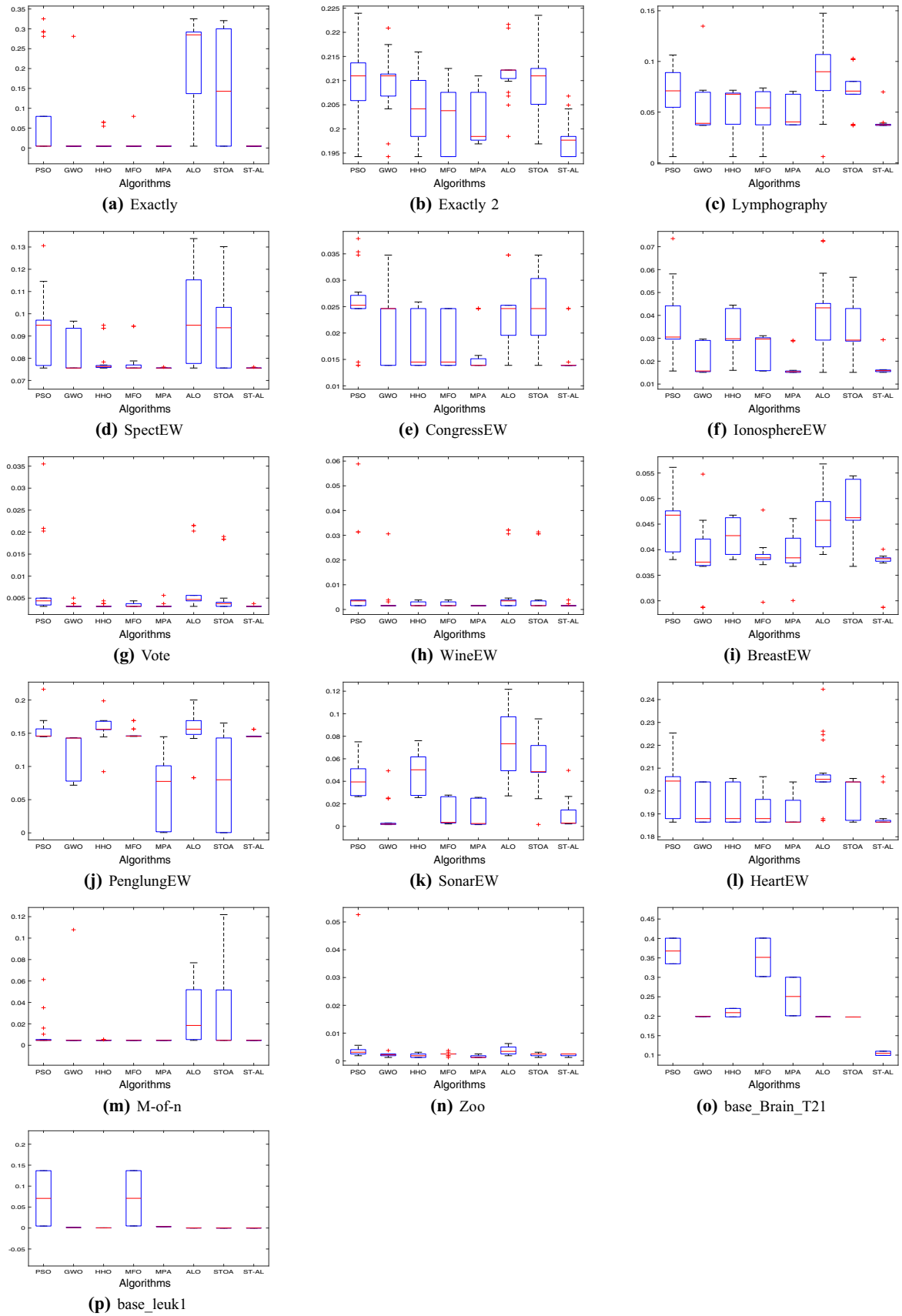
(g) Vote **(h)** WineEW **(i)** BreastEW

(j) PenglungEW **(k)** SonarEW **(l)** HeartEW

(m) M-of-n **(n)** Zoo **(o)** base_Brain_T21

(p) base_leuk1

Fig. 9 Convergence curve for ST-AL against other competitors—UCI datasets



(a) Exactly **(b)** Exactly 2 **(c)** Lymphography

(d) SpectEW **(e)** CongressEW **(f)** IonosphereEW

(g) Vote **(h)** WineEW **(i)** BreastEW

(j) PenglungEW **(k)** SonarEW **(l)** HeartEW

(m) M-of-n **(n)** Zoo **(o)** base_Brain_T21

(p) base_leuk1

Fig. 10 Boxplot for ST-AL against other competitors—UCI datasets

Table 14 A comparative study based on the classifier accuracy, with the state of the art feature selection methods

Dataset	ISOA	WOASA	SCHHO	GWOPSO	ASGW	GWOCrowSA	ST-AL
Exactly	**1**	**1**	0.812	**1**	0.999	0.99	**1**
Exactly2	0.7686	0.75	0.783	0.76	0.777	0.746	**0.806**
Lymphography	0.9252	0.89	**0.97**	0.92	0.884	0.87	0.9643
SpectEW	0.906	0.88	0.887	0.88	0.87	0.816	**0.9259**
CongressEW	0.985	0.98	0.97	0.98	0.97	0.963	**0.9874**
IonosphereEW	0.97	0.966	0.947	0.95	0.972	0.915	**0.9831**
Vote	0.985	0.97	0.987	0.97	0.984	0.948	**1**
WineEW	**1**	0.99	0.994	**1**	**1**	0.982	**1**
BreastEW	0.976	**0.985**	0.981	0.97	0.981	0.962	0.9658
PenglungEW	–	0.94	–	0.96	**1**	0.8595	0.856
SonarEW	0.9736	0.97	–	0.96	0.948	0.9058	**0.9929**
HeartEW	–	**0.85**	–	0.85	0.831	0.8326	0.8129
M-of-n	**1**	**1**	–	**1**	**1**	0.996	**1**
Zoo	**1**	0.97	–	**1**	**1**	0.9686	**1**
base_Brain_T21	–	–	–	–	–	–	**0.894**
base_leuk1	–	–	–	–	–	–	**1**

The bold values highlight the largest, or the highest value received per row of the data. It signifies that which algorithm is producing best result under same external conditions on a specific dataset

Table 15 A comparative study based on the size of selected features, with the state of the art feature selection methods

Dataset	ISOA	WOASA	SCHHO	GWOPSO	ASGW	GWOCrowSA	ST-AL
Exactly	6.89	6	**4.43**	6	6.87	6.4	5.5
Exactly2	3	**1**	2.07	1.6	7.93	4.6	7.4
Lymphography	7.62	6.8	**2.23**	9.2	11.2	8	6.95
SpectEW	8.43	9.6	6.23	8.4	10.17	8	**4.7**
CongressEW	5.6	4.4	**2.23**	4.4	8.83	5	3.95
IonosphereEW	8.4	11.4	**4.27**	13	17.3	13	5.6
Vote	7.25	5.8	3.7	**3.4**	8.97	4.6	5.05
WineEW	6.6	6.8	2.73	6	7.6	6.4	**2.2**
BreastEW	**7.58**	13.6	7.67	13.6	15.83	13.8	10.4
PenglungEW	–	325	–	130.8	170.3	165.8	**117.15**
SonarEW	20	60	–	31.2	35.3	29.6	**16.35**
HeartEW	–	13	–	5.8	6.367	5	**4.35**
M-of-n	7	13	–	6	6.867	6.4	**5.95**
Zoo	9.33	16	–	6.8	7.6	5.2	**2.5**
base_Brain_T21	–	–	–	–	–	–	**70.5**
base_leuk1	–	–	–	–	–	–	**13**

The bold values highlight the largest, or the highest value received per row of the data. It signifies that which algorithm is producing best result under same external conditions on a specific dataset

5.2.4 Exploration–exploitation analysis

Using Fig. 4, which depicts the 2D view of Exploration–Exploitation behavior when looking for the optimal global value maintained by the proposed ST-AL while solving the CEC'2020 test functions, we can better explain the phases of Exploration and exploitation. It is obvious from Fig. 4 that the proposed ST-AL has a high exploration to exploitation ratio in the beginning. Despite this, the majority of the time spent seeking is spent in the exploitation stage of the process. This behavior

demonstrates the proposed ST-AL is capable of balancing the exploration and exploitation stages efficiently.

5.3 Experimental series 2: feature selection problems

The proposed hybrid ST-AL approach is employed in this section on feature selection. It is an NP-hard combinatorial problem. Assuming that the dataset D has d features, the number of possible feature subsets is $2^d - 1$ (Eid 2018). Afterward, the ST-AL approach is used to discover the optimal possible subset of features. According to the proposed approach, the number of features and classification error rate are used to calculate a fitness value. The mathematical formula for the fitness function is (Mafarja and Mirjalili 2017):

$$\text{Fit} = \alpha CR(D) + \beta \frac{|FS|}{|d|} \tag{41}$$

where $CR(D)$ represents the error rate (calculated using the KNN classifier), $|d|$ represents the original feature set, and $|FS|$ shows the selected features. The parameters α and β can be selected within the range [0, 1]. α and β are the weights of error rate and the selection ratio, respectively, where α is the complement of β. As stated in the literature, control parameters α and β are set to 0.99 and 0.001, respectively (Kumar and Kaur 2020).

The proposed ST-AL method for determining the best subset of features is evaluated in this experimental by comparing it to other meta-heuristic feature selection algorithms. These algorithms are tested on fourteen distinct datasets, each of which has a different kind. These algorithms have the same parameter setting as defined in Table 3. From the UCI machine learning repository Asuncion (2007), the datasets utilized in this study were retrieved. Table 9 provides a short overview of each dataset utilized in the study. Moreover, different evaluation criteria are used in this work to assess the performance of the ST-AL method, for example: evaluating the fitness function values, the accuracy of the classifier according to the selected features, the size of the selected features, and the computational time as in Sect. 5.1.

5.3.1 Results and discussion of UCI datasets

The mean and standard deviation of the fitness function for the comparative methods is described in Table 10. The experimental results reveal that the proposed ST-AL yielded better results than other competitor algorithms. On 75% of the datasets, ST-AL had the best average outcomes (12 out of 16). It is also noteworthy to note that in three datasets (IonosphereEW, WineEW, and Zoo datasets), the MPA is superior to the other algorithms in terms of their mean fitness function values. ST-AL is the second-best algorithm for this dataset. For the SonarEW dataset, GWO performs better than the competitors. ST-AL is the second-best algorithm for this dataset. The results produced demonstrated the capability of the proposed ST-AL to address various feature selection problems. On the other hand, ST-AL is a more robust method for most datasets when compared to other strategies by analyzing standard deviation. The Std values proved that the proposed ST-AL produced close values throughout many runs with low distribution, which shows that the proposed ST-AL is a powerful method for handling different feature selection problems. In the Std measurement, ST-AL got the best value in twelve out of fourteen datasets. Figure 5 displays the average of the fitness values for all algorithms.

A comparison of accuracy results between ST-AL and other algorithms evaluated in the same settings is shown in Table 11. The ST-AL outperforms others on six of the sixteen tested datasets, whereas MPA performed best on two of the sixteen datasets (PenglungEW, IonosphereEW). There are seven datasets in which the results were identical between MPA and ST-AL. The ST-AL algorithm also performs better than the standard STOA and ALO algorithms. Figure 6 shows the average of the accuracy results from all methods. When compared to all other methods, ST-AL performed best. Experiment findings showed that the proposed ST-AL method selected the most informative features with higher accuracy values.

Table 12 compares the average number of features selected by the ST-AL and competing algorithms classifiers for the same UCI datasets. This study used sixteen datasets, and the average number of selected features acquired by ST-AL over seven datasets is the best compared to other optimizers. While STOA method obtained the best results in six datasets. MPA and GWO provide a minimum set of selected features for the WineEW and Exactly2 two datasets, respectively. Compared to other competitor methods, Fig. 7 shows that the conventional STOA got the smallest number of features, while the ST-AL method got the best second number of selected features. Based on an examination of the standard deviation, ST-AL, compared to other methods, is a reliable approach for the majority of datasets.

The computational time of the comparative methods is recorded in Table 13. When compared to the other approaches, the STOA method has the shortest execution time and is the fastest. As demonstrated in Fig. 8, the proposed ST-AL approach is almost the second method in terms of the speed of the execution time. Because of its combined structure, this proposed method requires some

time to discover the optimal solution, and this type of problem does not require real execution time.

Figures 9 and 10 depict the average fitness value and the competitive algorithms' convergence curve and boxplots. It can be observed that the proposed ST-AL approach, which integrates the STOA and ALO, increases the rate of convergence towards optimal solutions. This can be noticed for example at Exactly, Exactly2, Lymphography, SpectEW, CongressEW, Vote, HeartEW, M-of-n, base_Brain_T21, and base_leuk1. Furthermore, it can be seen from the boxplot that ST-AL has the lowest box.

According to the evaluation metrics and most of the test cases, the suggested ST-AL approach shows a significant improvement when compared to the other competitor methods. The combination of STOA and ALO is largely responsible for the impressive results of the proposed ST-AL method. Overall, we found that the proposed ST-AL method gave the best results and proven to be an efficient and effective optimization strategy for dealing with diverse feature selection problems.

5.3.2 Comparison with the state-of-the-art feature selection methods

This section compares the proposed ST-AL method with other hybrid approaches reported in recent literature relevant to feature selection. In Table 14, the accuracy values for the ST-AL approach are compared with various hybrid feature selection algorithms reported in the recent literature, including ISOA (Ewees et al. 2022), WOASA (Mafarja and Mirjalili 2017), SCHHO (Hussain et al. 2021), GWOPSO (Al-Tashi et al. 2019), ASGW (Mafarja et al. 2020), and GWOCrowSA (Arora et al. 2019). As revealed from Table 14, ST-AL obtained the most informative features resulting in the highest classification accuracy compared to other hybrid approaches in the literature, as the ST-AL method is more accurate on twelve of sixteen datasets. Likewise, Table 15 shows the number of selected features using ST-AL compared to other hybrid methods. This comparison revealed that the proposed ST-AL method picked a significantly low number of features than other hybrid approaches proposed in the literature since it obtained the smallest number of attributes with the highest accuracy values in six datasets out of sixteen.

6 Conclusions and future work

This paper presents a novel hybrid optimization algorithm based on the sooty tern optimization algorithm (STOA) and ant lion optimization (ALO) to handle function optimization problems as well as feature selection problems. In the proposed ST-AL, four strategies have been applied to improve the efficiency of STOA. The first strategy is the use of control randomization parameters, which plays a significant role in balancing the exploration–exploitation phases and avoiding falling in local optimum and premature convergence. The second strategy is concerned with developing a new exploration phase based on the ALO algorithm, where the ALO algorithm is known as a sound exploration strategy. The third strategy is enhancing the STOA exploitation phase by modifying the main equation of position updating. Finally, the last strategy is applying the greedy selection to neglect the poor generated population and prevent divergence of the algorithm from the existing promising regions. In order to assess the efficacy of the proposed ST-AL algorithm, the experiment is done on ten benchmark functions and 16 data set as a feature selection approach. Then, it has been compared with seven original meta-heuristic algorithms MPA, MFO, HHO, GWO, PSO, ALO, and STOA. The experimental results reveal that the ST-AL algorithm has generally outperformed the other seven compared algorithms and proved its capability and stability in solving the optimization issues. In terms of feature selection, the proposed ST-AL has achieved the mean best results on 75% of the datasets. Thus, it can be concluded that ST-AL can be an efficient optimization approach for addressing various feature selection problems. In future work, the proposed hybrid algorithm can be used to solve more realistic challenges in real-world scenarios.

Funding The author declares that there is no funding associated for this project.

Data availability Enquiries about data availability should be directed to the authors.

Declarations

Conflict of interest The authors of this manuscript declare that there is no conflict of interest.

Ethical approval The author of this manuscript confirms that: (i) Informed, written consent has been obtained from the relevant sources wherever is required; (ii) All procedures followed were in accordance with the ethical standards of the responsible committee on human experimentation (institutional and national) and with the Helsinki Declaration of 1964 and its later amendments. (iii) The approval and/or informed consent were obtained by human subjects where ever is applicable.

References

Abdel-Basset M, Ding W, El-Shahat D (2021) A hybrid Harris hawks optimization algorithm with simulated annealing for feature selection. Artif Intell Rev 54(1):593–637

Adamu A, Abdullahi M, Junaidu SB, Hassan IH (2021) An hybrid particle swarm optimization with crow search algorithm for feature selection. Mach Learn Appl 6:100108

Aghdam MH, Ghasem-Aghaee N, Basiri ME (2009) Text feature selection using ant colony optimization. Expert Syst Appl 36(3):6843–6853

Ali HH, Fathy A, Al-Shaalan AM, Kassem AM, MH Farh H, Al-Shamma'a AA, A Gabbar H (2021) A novel sooty terns algorithm for deregulated MPC-LFC installed in multi-interconnected system with renewable energy plants. Energies 14(17):5393

Al-Tashi Q, Jadid AKS, Rais HM, Mirjalili S, Alhussian H (2019) Binary optimization using hybrid grey wolf optimization for feature selection. IEEE Access 7:39496–39508

Anand P, Arora S (2020) A novel chaotic selfish herd optimizer for global optimization and feature selection. Artif Intell Rev 53(2):1441–1486

Arcuri A, Fraser G (2013) Parameter tuning or default values? An empirical investigation in search-based software engineering. Empir Softw Eng 18(3):594–623

Arora S, Singh S (2019) Butterfly optimization algorithm: a novel approach for global optimization. Soft Comput 23(3):715–734

Arora S, Singh H, Sharma M, Sharma S, Anand P (2019) A new hybrid algorithm based on grey wolf optimization and crow search algorithm for unconstrained function optimization and feature selection. IEEE Access 7:26343–26361

Arora S, Sharma M, Anand P (2020) A novel chaotic interior search algorithm for global optimization and feature selection. Appl Artif Intell 34(4):292–328

Asuncion A (2007) UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. http://www.ics.uci.edu/∼mlearn/MLRepository.html

Adel Assiri AS (2021) On the performance improvement of butterfly optimization approaches for global optimization and feature selection. Plos One 16(1):e0242612

Che Y, He D (2021) A hybrid whale optimization with seagull algorithm for global optimization problems. Math Probl Eng

Desuky AS, Hussain S, Kausar S, Islam MA, El Bakrawy LM (2021) EAOA: an enhanced archimedes optimization algorithm for feature selection in classification. IEEE Access 9:120795–120814

Dhiman G, Kaur A (2019) STOA: a bio-inspired based optimization algorithm for industrial engineering problems. Eng Appl Artif Intell 82:148–174

Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, MHS'95. IEEE, pp 39–43

Eid HF (2018) Binary whale optimisation: an effective swarm algorithm for feature selection. Int J Metaheuristics 7(1):67–79

EL-Hasnony IM, Elhoseny M, Tarek Z (2021) A hybrid feature selection model based on butterfly optimization algorithm: Covid-19 as a case study. Expert Syst e12786

Emary E, Zawbaa HM, Hassanien AE (2016) Binary ant lion approaches for feature selection. Neurocomputing 213:54–65

Ewees AA, Al-qaness MAA, Abualigah L, Oliva D, Algamal ZY, Anter AM, Ali IR, Ghoniem RM, Abd Elaziz M (2021) Boosting arithmetic optimization algorithm with genetic algorithm operators for feature selection: case study on cox proportional hazards model. Mathematics 9(18):2321

Ewees AA, Mostafa RR, Ghoniem RM, Gaheen MA (2022) Improved seagull optimization algorithm using lévy flight and mutation operator for feature selection. Neural Comput Appl 1–36

Faramarzi A, Heidarinejad M, Mirjalili S, Gandomi AH (2020) Marine predators algorithm: a nature-inspired metaheuristic. Expert Syst Appl 152:113377

Fausto F, Cuevas E, Valdivia A, González A (2017) A global optimization algorithm inspired in the behavior of selfish herds. Biosystems 160:39–55

Ghanem K, Layeb A (2021) Feature selection and knapsack problem resolution based on a discrete backtracking optimization algorithm. Int J Appl Evol Comput (IJAEC) 12(2):1–15

Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning

Gu S, Cheng R, Jin Y (2018) Feature selection for high-dimensional classification using a competitive swarm optimizer. Soft Comput 22(3):811–822

Hashim FA, Hussien AG (2022) Snake optimizer: a novel metaheuristic optimization algorithm. Knowl Based Syst 108320

Hashim FA, Houssein EH, Hussain K, Mabrouk MS, Al-Atabany W (2022) Honey badger algorithm: new metaheuristic algorithm for solving optimization problems. Math Comput Simul 192:84–110

Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Future Gener Comput Syst 97:849–872

Houssein EH, El-din HB, Rezk H, Nassef AM (2021) An enhanced archimedes optimization algorithm based on local escaping operator and orthogonal learning for PEM fuel cell parameter identification. Eng Appl Artif Intell 103:104309

Huang Y, Jin W, Yu Z, Li B (2020) Supervised feature selection through deep neural networks with pairwise connected structure. Knowl Based Syst 204:106202

Hussain K, Neggaz N, Zhu W, Houssein EH (2021) An efficient hybrid sine-cosine Harris hawks optimization for low and high-dimensional feature selection. Expert Syst Appl 176:114778

Hussien AG, Amin M (2021) A self-adaptive Harris hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection. Int J Mach Learn Cybern 1–28

Jia H, Xing Z, Song W (2019) A new hybrid seagull optimization algorithm for feature selection. IEEE Access 7:49614–49631

Kader M, Zamli KZ (2022) Comparative study of five metaheuristic algorithms for team formation problem. Human-centered technology for a better tomorrow. Springer, Berlin, pp 133–143

Dervis Karaboga, Bahriye Basturk (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471

Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks, vol 4. IEEE, pp 1942–1948

Khaleel LR, Mitras BA (2020) Hybrid whale optimization algorithm with modified conjugate gradient method to solve global optimization problems. Open Access Libr J 7(6)

Khamees M, Al-Baset RA (2020) Hybrid SCA-CS optimization algorithm for feature selection in classification problems. In: AIP conference proceedings, vol 2290. AIP Publishing LLC, p 040001

Kumar V, Kaur A (2020) Binary spotted hyena optimizer and its application to feature selection. J Ambient Intell Humaniz Comput 11(7):2625–2645

Long W, Jiao J, Liang X, Wu T, Xu M, Cai S (2021) Pinhole-imaging-based learning butterfly optimization algorithm for global optimization and feature selection. Appl Soft Comput 103:107146

Mafarja MM, Mirjalili S (2017) Hybrid whale optimization algorithm with simulated annealing for feature selection. Neurocomputing 260:302–312

Mafarja M, Qasem A, Heidari AA, Aljarah I, Faris H, Mirjalili S (2020) Efficient hybrid nature-inspired binary optimizers for feature selection. Cogn Comput 12(1):150–175

Mirjalili S (2015a) The ant lion optimizer. Adv Eng Softw 83:80–98

Mirjalili S (2015b) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Syst 89:228–249
Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133
Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
Mirjalili SZ, Mirjalili S, Saremi S, Faris H, Aljarah I (2018) Grasshopper optimization algorithm for multi-objective optimization problems. Appl Intell 48(4):805–820
Mohamed AW, Hadi AA, Mohamed AK, Awad NH (2020a) Evaluating the performance of adaptive gainingsharing knowledge based algorithm on CEC 2020 benchmark problems. In: 2020 IEEE congress on evolutionary computation (CEC). IEEE, pp 1–8
Mohamed AK, Hadi AA, Mohamed AW (2020b) Generalized adaptive differential evolution algorithm for solving CEC 2020 benchmark problems. In: 2020 2nd Novel intelligent and leading emerging sciences conference (NILES). IEEE, pp 391–396
Motoda H, Liu H (2002) Feature selection, extraction and construction. Commun IICM (Institute of Information and Computing Machinery, Taiwan) 5(67–72):2
Neggaz N, Ewees AA, Abd Elaziz M, Mafarja M (2020) Boosting salp swarm algorithm by sine cosine algorithm and disrupt operator for feature selection. Expert Syst Appl 145:113103
Oh IS, Lee J-S, Moon B-R (2004) Hybrid genetic algorithms for feature selection. IEEE Trans Pattern Anal Mach Intell 26(11):1424–1437
Oliva D, Elaziz MA (2020) An improved brainstorm optimization using chaotic opposite-based learning with disruption operator for global optimization and feature selection. Soft Comput 24(18):14051–14072
Papa JP, Pagnin A, Schellini SA, Spadotto A, Guido RC, Ponti M, Chiachia G, Falcão AX (2011) Feature selection through gravitational search algorithm. In: 2011 IEEE International conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 2052–2055
Rani ASS, Rajalaxmi RR (2015) Unsupervised feature selection using binary bat algorithm. In: 2015 2nd International conference on electronics and communication systems (ICECS). IEEE, pp 451–456
Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell 48(10):3462–3481
Soliman GMA, Abou-El-Enien THM, Emary E, Khorshid MMH (2018) A novel multi-objective moth-flame optimization algorithm for feature selection. Indian J Sci Technol 11(38):1–13
Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
Teng X, Dong H, Zhou X (2017) Adaptive feature selection using v-shaped binary particle swarm optimization. PloS One 12(3):e0173907
Uzer MS, Yilmaz N, Inan O (2013) Feature selection method based on artificial bee colony algorithm and support vector machines for medical datasets classification. Sci World J
Wang J, Hedar A-R, Wang S, Ma J (2012) Rough set and scatter search metaheuristic based feature selection for credit scoring. Expert Syst Appl 39(6):6123–6128
Wang S, Jia H, Liu Q, Zheng R (2021) An improved hybrid Aquila optimizer and Harris hawks optimization for global optimization. Math Biosci Eng 18(6):7076–7109
Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
Yang XS (2010) Nature-inspired metaheuristic algorithms, firefly algorithm
Yang X-S (2012) Flower pollination algorithm for global optimization. In: International conference on unconventional computing and natural computation. Springer, Berlin, pp 240–249
Zhang J, Hu X, Li P, He W, Zhang Y, Li H (2014) A hybrid feature selection approach by correlation-based filters and SVM-RFE. In: 2014 22nd International conference on pattern recognition. IEEE, pp 3684–3689
Zhang L, Liu L, Yang X-S, Dai Y (2016) A novel hybrid firefly algorithm for global optimization. PloS One 11(9):e0163230
Zheng T, Zhang J, Zhu H (2021) Uncalibrated visual servo system based on Kalman filter optimized by improved STOA. In: 2021 IEEE 2nd International conference on information technology, big data and artificial intelligence (ICIBA), vol 2. IEEE, pp 119–124

Authors and Affiliations

Reham R. Mostafa[1] · Noha E. El-Attar[2] · Sahar F. Sabbeh[2,3] · Ankit Vidyarthi[4] · Fatma A. Hashim[5]

✉ Reham R. Mostafa
reham_2006@mans.edu.eg

✉ Ankit Vidyarthi
dr.ankit.vidyarthi@gmail.com

Noha E. El-Attar
noha.ezzat@fci.bu.edu.eg

Sahar F. Sabbeh
sahar.fawzy@fci.bu.edu.eg

Fatma A. Hashim
fatma_hashim@h-eng.helwan.edu.eg

[1] Information Systems Department, Faculty of Computers and Information Sciences, Mansoura University, Mansoura 35516, Egypt

[2] Faculty of Computers and Artificial Intelligence, Benha University, Banha, Egypt

[3] College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia

[4] Department of CSE&IT, Jaypee Institute of Information Technology, Noida, India

[5] Faculty of Engineering, Helwan University, Helwan, Egypt