Journal of Computers Vol. 33 No. 5, October 2022, pp. 121-133
doi: 10.53106/199115992022103305011

Particle Swarm Optimization with Long and Short Term Memory in Feature Selection

Xiang Xu[1,2], Yi Li[1*], Yi-Fan Wang[1]

[1] School of Computer Science & School of Cyberspace Science of Xiangtan University, Xiangtan, Hunan Province, China
[2] School of Computer Science and Engineering of Hunan University of Information Technology, Changsha, Hunan Province, China
xiangx@smail.xtu.edu.cn, Li_Yi@xtu.edu.cn

Received 13 December 2021; Revised 3 February 2022; Accepted 3 March 2022

Abstract. Taking each iteration of Particle swarm optimization (PSO) algorithm as a time node, the change of population in PSO algorithm can be regarded as a time series model. Particle population learns and evolves in multiple time nodes, which can be regarded as a dependent behavior on leader particles. In the traditional particle swarm optimization algorithm, this dependence behavior is independent of time, and its consideration standard is only the fitness value of particles. We deeply study the leadership mechanism of PSO algorithm in order to find a more robust leadership mechanism and improve the ability of PSO algorithm to explore the solution space, by extending the dependence behavior in the time dimension, we propose an improved PSO algorithm with long-term and short-term memory ability. In order to verify its performance, in the experimental part, we select 32 public data sets in UCI data to find the optimal feature subset. In a large number of feature selection experiments. The experimental results proofed that the performance of proposed algorithms is better than some state of the art algorithms.

Keywords: feature selection, particle swarm optimization, long and short term memory, classification

1 Introduction

In the classification problem, uniform data sampling is very important, and non-uniform data sampling can not truly reflect the distribution of the actual samples, which affects the effect of the classification model. In addition, if there is no prior knowledge, the number of selected features will be very large, which will cause Curse of Dimensionality. The purpose of feature selection is to solve the above problems. The strength of feature selection is that it can effectively reduce the data dimension and obtain better model performance. In the case that each feature can be selected, if there are n features, then the number of solutions is 2N [1] and the optimal solution of this kind of problem has been proved to be NP hard [2-3]

Some early search methods, such as sequential forward selection (SFS) [4], sequential backward selection (SBS) [5], sequential floating forward selection (SFFS) [6], sequential floating backward selection (SFBS) [6], have obvious problems. Such greedy algorithms are prone to local optimization and almost exhaustive search methods lead to a huge computational overhead. With the development of swarm intelligence algorithm and evolutionary algorithm, solving combinatorial optimization problems such as feature selection shows better performance. Particle swarm optimization (PSO) [7], genetic algorithm (GA) [8], ant colony optimization (ACO), simulated annealing (SA) [9] and other metaheuristic algorithms have been proposed successively. PSO is a metaheuristic algorithm based on swarm intelligence. Its main idea is to simulate the foraging behavior of swarms such as fish and birds, which is a bionic algorithm. PSO has the ability to quickly converge [10] and can be implemented quickly, it has a strong search capability in the problem space and can efficiently find minimal reducts. It's shortcoming is that it is difficult to implement in discrete problems and easy to fall into local optimum. To solve the Discrete problem, a binary particle swarm optimization algorithm (BPSO) have been proposed by Kennedy et al. [11]. Researchers have given a lot of improvement schemes on the basis of BPSO, but the two major problems of BPSO have not been solved. At present, the hybrid PSO algorithm has received more attention [12-14].

In order to solve the conventional problem of PSO algorithm, considering the time dimension characteristics of PSO algorithm population learning and evolution, an algorithm named LSTMPSO is proposed by expanding the learning behavior of the population. In LSTMPSO, the learning of long-term and short-term historical experience is maintained as a trade-off, which reduces the possibility of population falling into local optimum and improves the search ability of the algorithm for solution space.

* Corresponding Author

The remainder of this paper is organized as follows. Sect 2 provides background information. Sect 3 describes the proposed LSTMPSO algorithm. Sect 4 describes the experimental design, experimental results with discussions, and Sect 5 presents the conclusion.

2 Literature Review

This section will introduce the basic theory and improvement of PSO algorithm, as well as the research status of feature selection model.

2.1 PSO

Particle swarm optimization (PSO) is an algorithm based on swarm intelligence proposed by Kennedy et al. [7] In 1995. By simulating the foraging behavior of fish and birds, the particles learn from each other, and then make the population converge near the optimal solution. Suppose the solution space of the problem is Ω, define a population of m particles as $X_i \in \{\Omega \mid i = 1, 2, 3, ..., m\}$ and each particle has a velocity vector of in the solution space $V_i \in \{Q \mid i = 1, 2, ..., m\}$. After randomly initializing the position and velocity of the particles in the solution space, *pbest* is selected for each particle and the unique *gbest* of the population is selected, where *pbest* represents the historical optimal solution of the particle and *gbest* represents the historical optimal solution of the population.

$$V_i(t+1) = \omega(t) \times V_i(t) + c_1 r_1(t) \times \left(pbest_i(t) - x_i(t)\right) + c_2 r_2(t) \times \left(gbest(t) - x_i(t)\right). \quad \textbf{(1)}$$

$$X_i(t+1) = X_i(t) + V_i(t+1). \quad \textbf{(2)}$$

As shown in Eq. (1-2), the update formulas of particle position and velocity are given, where $w(t)$ is the inertia coefficient inherited from the current velocity, c_1, c_2 are the learning factors for *pbest* and *gbest* respectively, usually $c_1 = 2$, $c_2 = 2$. r_1, r_2 are random values in [0, 1].

$$IF\ V_i^{t+1} < V_{\min}, then\ V_i^{t+1} \leftarrow V_{\min};\ IF\ V_i^{t+1} > V_{\max}, then\ V_i^{t+1} \leftarrow V_{\max}. \quad \textbf{(3)}$$

As shown in Eq. (3), M. Clerc et al. [15] suggested that the limitation of V can achieve better results. By limiting the maximum velocity, particles cannot fly too far away from the optimal solution thus it can ensure that particles have better global search ability and local search ability at the same time. M.S. Mohamad et al. [16] Proposed to replace V with a length or magnitude of V, which makes it easier to select small feature subsets.

$$\omega(t+1) = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{MAXITER} * Iter(t). \quad \textbf{(4)}$$

S. Naka et al. [17] and T. Peram et al. [18] adopt different adaptive inertia coefficient adjustment strategies. In this paper, we will use the update strategy of Eq. (4), where $w(T)$ is the value of inertia coefficient at iteration of t, w_{max} means Initial value of weighting coefficient and w_{min} means final value of weighting coefficient, $Iter(t)$ is the current number of iterations, respectively. The larger inertia weights at the beginning help to find good seeds and the later small inertia weights facilitate fine search [19].

2.2 BPSO

In order to make PSO algorithm deal with discrete problems better, Kennedy improved it and proposed the binary particle swarm optimization (BPSO) [11] algorithm. In the BPSO algorithm for solving an D-dimensional feature selection problem, a population with m particles is defined, and the position and velocity of the ith particle in the population are defined as $X_i = (x_1, x_2, ..., x_D)$ and $V_i = (v_1, v_2, ..., v_D)$ where $i \in [1, m]$ represents the number of particles, $d \in D$ represents the number of dimensions. Each particle has a binary value (0 or 1) in any dimension, In the feature selection problem, 1 means to select this feature, 0 means not to select this feature.

$$V_i^d(t+1) = \omega(t) \times V_i^d(t) + c_1 r_1^d(t) \times \left(pbest_i^d(t) - x_i^d(t)\right) + c_2 r_2^d(t) \times \left(gbest(t) - x_i^d(t)\right). \quad \textbf{(5)}$$

$$Sigmoid\left(v_i^d(t+1)\right) = \frac{1}{1+e^{-v_i^d(t+1)}} \ . \tag{6}$$

$$\begin{aligned} &IF\ Sigmoid\left(v_i^{d(t+1)}\right) > r_3^d(t)\ , \\ &then\ x_i^d(t+1) = 1\ ; \\ &else\ x_i^d(t+1) = 0\ . \end{aligned} \tag{7}$$

As shown in Eq. (5-7), BPSO algorithm is different from classic PSO algorithm in updating strategy. BPSO uses a sigmoid function to map the velocity value of any dimension to [0, 1] interval as a probability to decide whether to update the position of this dimension to 1. The flow chart of BPSO was shown in Fig. 1.

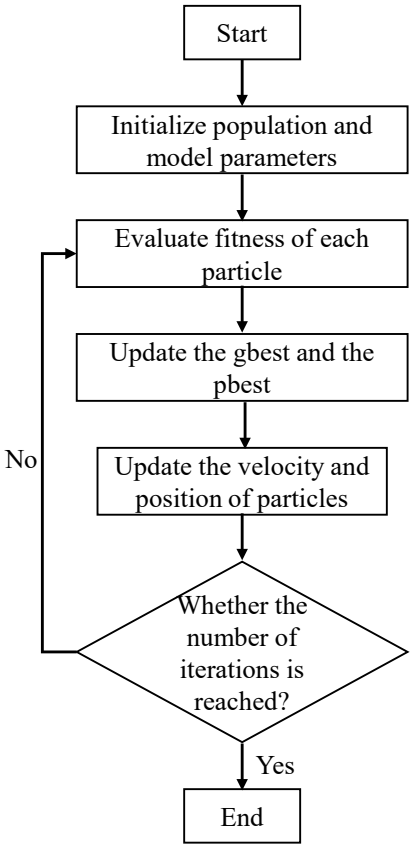


Fig. 1. BPSO algorithm flow

2.3 Feature Selection

According to evaluation criteria, feature selection methods can be divided into two categories, Filter method and Wrapper method [20].

2.3.1 Filter Method

According to information theory, fuzzy set theory and other methods, the filtering method designs fitness function as evaluation criteria, and selects redundant features. B. Chakraborty et al. [21] proposed a computationally light fuzzy fitness function which is more efficient than the conventional classifier algorithm. Kundu et al. [22] proposed a MOGA-based fuzzy proximity algorithm which was performed for selecting a feature subset. Xiangyang Wang et al. [23] proposed a PSORSFS algorithm based on rough sets and PSO. L. Cervante et al. [24] proposed

Two fitness functions based on entropy and mutual information and the main idea of these functions is to maximize the correlation between feature subset and class label and minimize the redundancy of feature subset. Alper Unler et al. [25] proposed a hybrid algorithm called mr2PSO which uses the mutual information available from the filter model to weigh the bit selection probabilities in the discrete PSO.

2.3.2 Wrapper Method

Wrapper method takes the results of classification model as the evaluation criteria, so it has better model performance than filtering method in most cases, but the defect of wrapper method is that it needs more computation.

E. Alba et al. [26] proposed a GPSO algorithm which use a three-parent mask-based crossover (3PMBCX) operator to replace the movement of particles. An algorithm called IBPSO is proposed to reset gbest by the criterion that gbest is not updated in three iterations. Alper Unler et al. [27] proposed an extensible social learning behavior and added a new learning target ibest (optimal solution of current iterative population). Classical genetic algorithm (GA) has appeared many improved algorithms, such as the nondominated sorting genetic algorithm II (NSGA-II) [28], proposed a fast non dominated sorting method to reduce the computational complexity, introduced the concept of crowding degree to maintain individual diversity, retained the elite strategy to rapidly improve the quality of the population. The related work of NSGA-II in feature selection is as follows: [29-31]. Bing Xue et al. [13] firstly proposed two algorithm called NSPSOFS and CMDPSOFS which combines PSO and multi-objective optimization algorithm. Pedram Ghamisi et al. [32] proposed a hybrid algorithm named HGAPSO can be used for road detection. Yong Zhang et al. [33] has conducted the first study on multi-objective PSO for cost-based feature selection problems. J. Vijayal et al. [14] proposed a hybrid algorithm based on PSO and simulated annealing to avoid the population falling into the optimal solution by randomly replacing the leading particles.

3. LSTMPSO

The long-term and short-term memory network is a time series network model, which can well capture the changing trend in time series problems, dig out hidden information and learn from it. As shown in Fig. 2, the PSO algorithm has multiple iterations, and if each iteration is considered as a time node, the entire PSO algorithm can also be considered as a time series model. In many iterations of the PSO algorithm, individuals learn from both *pbest* and *gbest* simultaneously. We interpret learning behavior as a dependence and name learning from two particles as individual dependence and population dependence, which are time independent and can only be evaluated by fitness value. Therefore, we extend individual dependency in the PSO algorithm from time dimension to long-term individual dependence (*lpbest*) to short-term individual dependence (*spbest*), and similarly, population dependence to long-term population dependence (*sgbest*) and short-term population dependence (*sgbest*). The following subsections describe the long-term and short-term division rules and the update rules of population dependence.

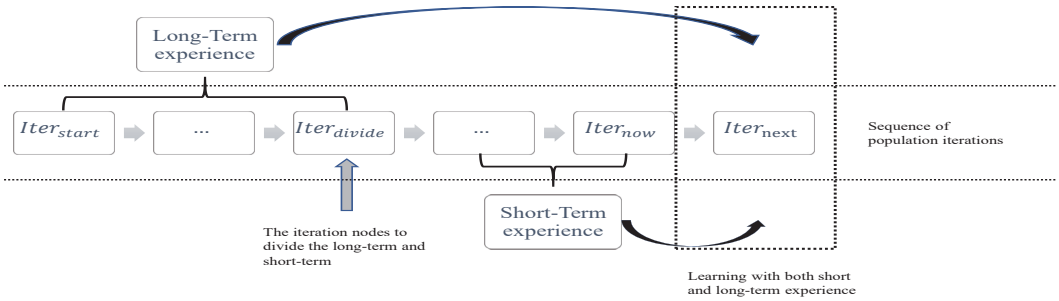


Fig. 2. LSTMPSO learning mode

1) Node division: To make the long-term and short-term division clearer, we consider the PSO algorithm iteration process as a time series and each iteration as a node. For the first node of the PSO algorithm, a dividing node is found based on the period of change. Iterations before this node are considered to be in the long-term range, while those after this node are considered to be in the short-term range. We can determine the division by defining

the length of the change period, that is, by making the change period a constant, we can simply divide the long and short term. Considering that the change period of the constant is unstable with the increase of the number of iterations, we propose a partition method based on the current number of iterations as follows:

$$node = \max\left(\frac{iter}{2}, iter - 10\right). \tag{8}$$

If we want to add a long-term dependency to the population, we must preserve the population history from the dividing node to the current node. When the number of iterations is too large, if the dividing node is still computed by multiplying the current node by a constant in the range [0, 1], the spatial complexity of the algorithm becomes $O(n * m)$, n and m are the number of iterations and the population size, respectively. To reduce the spatial complexity of the algorithm, we set a constant number of short-term dependent iterations when the number of iterations is too large, so that the spatial complexity of the algorithm will be $O(m)$.

2) Learning factor (Momentum method): In the classical PSO algorithm, the importance of *gbest* and *pbest* is reflected by learning factors c_1 and c_2. In most cases, they will be preset to the same value. However, for long-term and short-term memory, short-term memory is usually easier to remember, and long-term memory should be lost slowly over time. In this paper, a momentum method is proposed, which gives short-term memory greater learning weight, makes the population search in the solution space have a certain flight inertia, and can reduce the probability of oscillation of population learning behavior.

$$c_i = \frac{iter_{now} - iter_P}{maxiter} * F(P), i \in \{3,4\}. \tag{9}$$

As shown in Eq.(8), two new learning factors c_3 and c_4 are determined by momentum method, the $F(P)$ is the fitness value of the dependent particle (for both *gbest* and *pbest*), the $iter_{now}$, $iter_P$, *maxiter* are the current number of iterations, the number of iterations to which the dependent particle belongs, and the maximum number of iterations respectively. By calculating the proportion of the difference of iterations to the maximum number of iterations for dependent particles, the above method can better simulate the task of giving long-term dependent smaller weights, which is a simple and effective method.

3) Speed update: The update method of particle velocity is as follows:

$$\begin{aligned} V_i^d(t+1) = {} & \omega(t) V_i^d(t) \\ & + c_1 r_1^d(t) \times \left(c_3 spbest_i^d(t) + c_4 lpbest_i^d(t) - x_i^d(t)\right). \\ & + c_2 r_2^d(t) \times \left(c_3 sgbest_i^d(t) + c_4 lgbest_i^d(t) - x_i^d(t)\right) \end{aligned} \tag{10}$$

4) Dependency update: After extending dependencies, we need to update four types of dependencies at each node. Among them, short-term individual dependency and short-term population dependency can be updated according to the current node, while long-term individual dependency and long-term population dependency can be updated according to the partition node. The time complexity and spatial complexity of the update process are $O(1)$.

$$spbest_i = pbest_i, if\ F(pbest_i) > F(spbest_i). \tag{11}$$

$$sgbest = gbest, if\ F(gbest) > F(sgbest). \tag{12}$$

$$lpbest_i = pbest_i, if\ F(pbest_i) > F(lpbest_i). \tag{13}$$

$$lgbest = gbest, if\ F(gbest) > F(lgbest). \tag{14}$$

As shown in Eq. (11-12), we use the $F(x)$ function to calculate the fitness of all particles in the current population. If the fitness values of $pbest_i$ and *gbest* in the current node are greater than the existing values, the short-term dependent $spbest_i$ and *sgbest* will be updated. This process is actually the same as that of the traditional PSO algorithm. Importantly, we divide the historical solution set of the population once, so as to simulate that the population has long-term and short-term memory ability respectively. Therefore, in Eq. (13-14), after each partition

node is calculated through the current node, if the $pbest_i$ and *gbest* fitness values in the partition node are greater than the existing long-term dependency values, $lpbest_i$ and *lgbest* are updated.

5) Three part initialization: Different population initialization strategies have a significant impact on the solution space search scope, we use a three-part initialization strategy that divides the initial population into three parts. For the first part population, the probability of any dimension of all particles being 1 is 0.1, for the second part population, the probability of any dimension of all particles being 1 is 0.5, and for the third part population, the probability of any dimension of all particles being 1 is 0.9.

6) Flow chart: The flow chart of LSTMPSO was shown in Fig. 3.

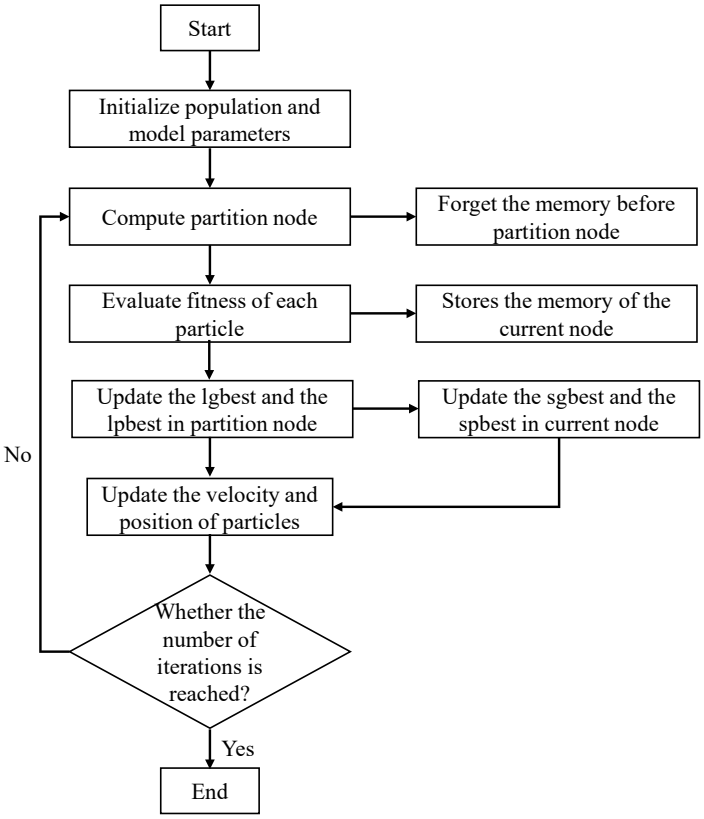


Fig. 3. LSTMPSO algorithm flow

4. Experiment & Discuss

In this section, we will first introduce the datasets from UCI benchmark repository, our experimental environment and methods. Finally, we will discuss the details of experimental results.

4.1 Experiment Preparation

The Table 1 shows 32 datasets from UCI database that we selected for the experiment. For the missing data in the dataset, we use mode to fill in the missing data or directly eliminate the seriously missing data, and use minmax method to normalize the data. In addition, for the data set that is not divided into training data and test data, we shuffle the data first, and then use the 10 fold method to divide the dataset. The machine used in this experiment is a HP high performance computer with Intel(R) Core(TM) i7-10700 CPU @ 2.9-4.8GHZ and a 32GB high speed memory. All the experiments were completed by the machine independently.

Table 1. Description of 32 UCI datasets used for evaluation of the proposed model

Dataset	Instances	Features	Classes	Dataset	Instances	Features	Classes
Arrhythmia	452	278	16	Australian	690	14	2
Breast	699	10	2	BreastEW	73	326	2
Congress	435	16	2	Diabetes	540	16	2
DryBean	13611	16	7	Exactly	1000	14	2
ForestType	198	28	4	German	1000	25	2
Glass	214	11	6	HillValley	606	101	2
Horse	300	28	2	Ionosphere	351	34	2
Iris	150	5	3	Madelon	2000	501	2
Monk1	124	7	2	Monk2	169	7	2
Monk3	122	7	2	PenglungEW	73	326	7
Seismic	2584	19	2	ShillBidding	6321	12	2
Sonar	208	61	2	Soybean (small)	47	36	4
SPECT	80	23	2	SpeakAccent	329	13	6
Tictactoe	958	10	2	SportsArticle	1000	60	2
Vowel	528	13	11	Urbanlandcover	168	148	9
Wine	178	14	3	Zoo	101	117	7

4.2 Fitness Function

The classifier we use is Extreme Learning Machine (ELM) [34], a single hidden layer neural network algorithm, can be used for regression or classification [35]. It is characterized by calculating the remaining weights by randomly initializing the partial weights and then using a ridge regression method, which eliminates the training process of the neural network and speeds up the calculation. Generally speaking, ELM is only a single layer neural network classifier and does not have the strong fitting ability like Multi-Layer Perception, but its efficient training speed can help us to quickly verify the effectiveness of LSTMPSO algorithm.

In classification problems, confusion matrix can be used to calculate various evaluation criteria. The relevant formulas are shown in Eq. (15-17) where *TP*, *TN*, *FP*, and *FN* stand for true positives, true negatives, false positives, and false negatives, respectively. But for imbalanced datasets, F-measure as a composite evaluation standard can better reflect the real performance of the classifier, so we use F-measure as the evaluation standard. The calculation method of F-measure is shown in Eq. (18).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} . \tag{15}$$

$$Precision = \frac{TP}{TP + FP} . \tag{16}$$

$$Recall = \frac{TP}{TP + FN} . \tag{17}$$

$$F - Measure = 2\frac{Precision * Recall}{Precision + Recall} . \tag{18}$$

Because the feature selection problem needs to optimize two objectives at the same time, which are to improve the classifier performance and reduce the feature dimension, the fitness function also considers the proportion of the number of features used in all features.

$$fitness = \alpha \times F - Measure + (1-\alpha) * \frac{Features}{All\ Features}. \quad (19)$$

As shown in Eq. (19), α represents a weight factor, *Features* represents the number of features used in classifier and *All Features* represents the number of all features, respectively. It is generally accepted that more attention should be paid to the performance of classifiers, so in this paper, $\alpha = 0.9$.

4.3 Result & Dicuss

To evaluate the proposed LSTMPSO algorithm for feature selection, we have implemented several classical algorithm for comparison, i.e., Random generation plus Sequential Backward Selection (RGSBS), Random generation plus Sequential Forward Selection (RGSFS), GA, PSO, PSO-FSSA [34], LSTMPSO and pure ELM classifier. Except the pure ELM classifier, each algorithm has 4800 times of calculation limit of fitness function. For GA, PSO, PSO-FSSA and LSTMPSO, the number of population is 30 and the number of iterations is 160. For other algorithms, including RGSBS and RGSFS only limits the number of fitness calculations. We run each algorithm 24 times and calculate the average result.

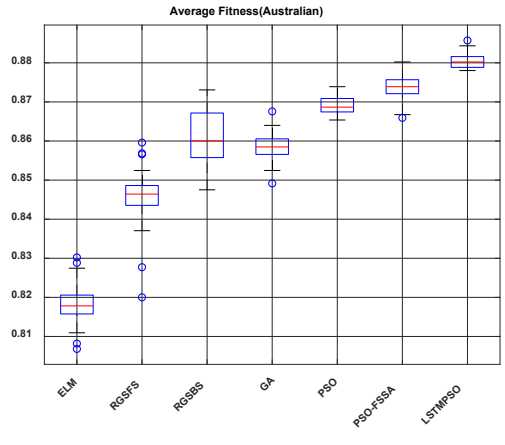
Table 2 shows the average running time of three PSO-based algorithms on 12 UCI datasets, and the results show that the computing time of the three algorithms is approximately the same. Drybean, the largest dataset, has more than 13,000 samples, which verifies that the LSTMPSO algorithm does not significantly increase the computational complexity in large datasets.

Fig. 4 shows a boxplot diagrams of the distribution of 24 runs detailed classification results for each dataset. LSTMPSO algorithm finds the optimal solution region and achieves a very perfect result in most datasets. In the Seismic and ShillBidding dataset, several algorithms have easily achieved good performance, so it is difficult to reflect the margin. In other data sets, LSTMPSO algorithm outperforms other algorithms in median results, and the results of 24 runs are more concentrated, which shows stronger robustness.

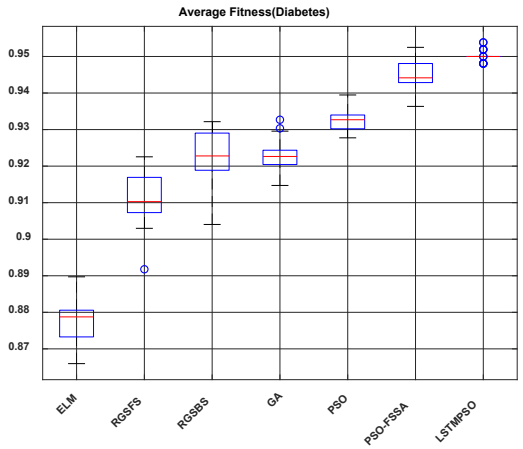
As shown in Fig. 5, ELM classifier without any optimization algorithm performs poorly, followed by RGSBS, RGSFS, which is easy to fall into local optimum. Because the number of the experimental population is not large enough, so the performance of GA algorithm is very general. The performance of PSO algorithm is slightly lower than the remain two improved PSO algorithm. On average, the proposed LSTMPSO algorithm achieves superior performance and outperforms all the other compared methods for each dataset significantly.

Table 2. Results of running time of three PSO-based algorithms on 12 UCI datasets (in minutes)

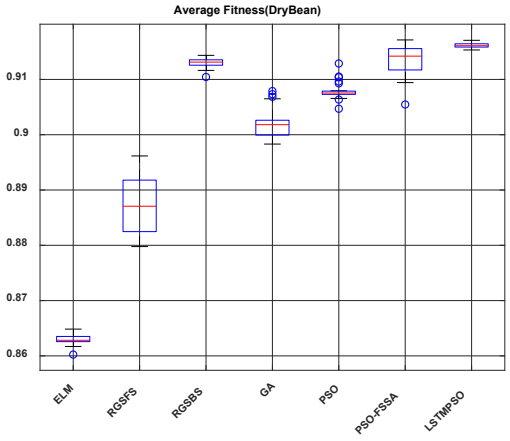
	Australian	Diabetes	DryBean	ForestType	German	Japan
PSO	0.4887	4.1930	30.0165	4.4404	0.2832	4.2801
PSO-FSSA	0.4937	4.2079	30.6920	4.2666	0.4210	4.6139
LSTMPSO	0.4933	4.1751	30.7088	4.2711	0.4139	4.2365
	Seismic	ShillBidding	SpeakAccent	SPECT	SportsArticle	Urban
PSO	15.0243	8.2719	0.2101	5.4742	4.3591	4.7824
PSO-FSSA	17.1791	9.0194	0.3688	5.0988	4.7270	4.6551
LSTMPSO	17.2298	9.0362	0.3520	5.2302	4.5647	4.5675



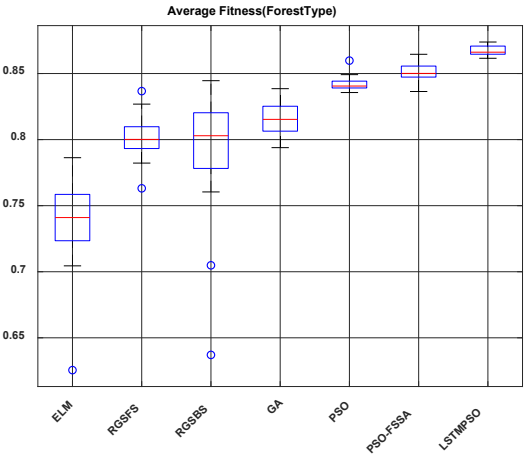
(a) Average fitness (Australian)



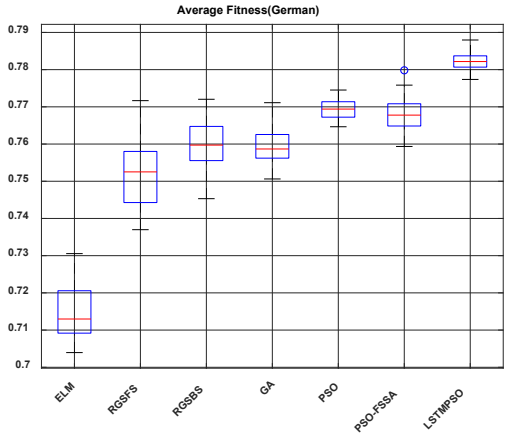
(b) Average fitness (Diabetes)



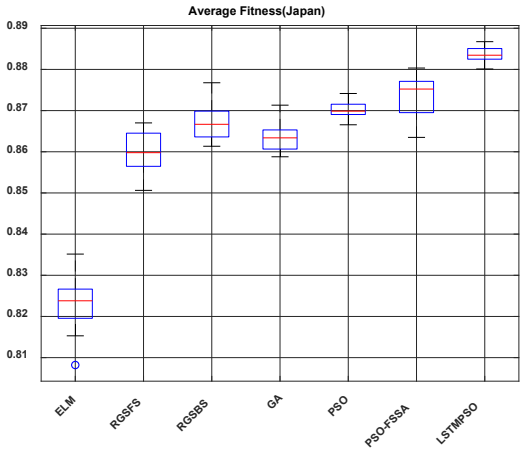
(c) Average fitness (DryBean)



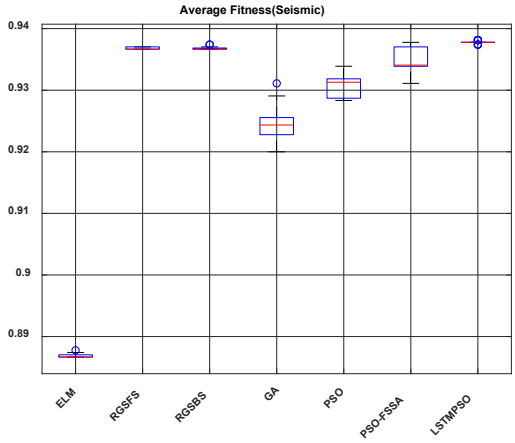
(d) Average fitness (ForestType)



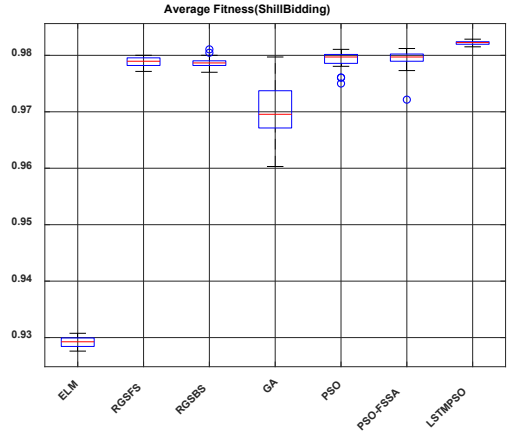
(e) Average fitness (German)



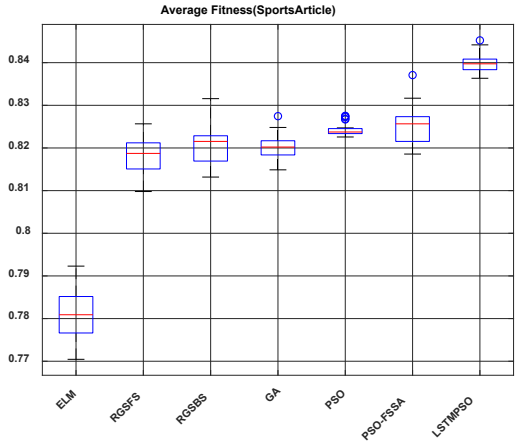
(f) Average fitness (Japan)



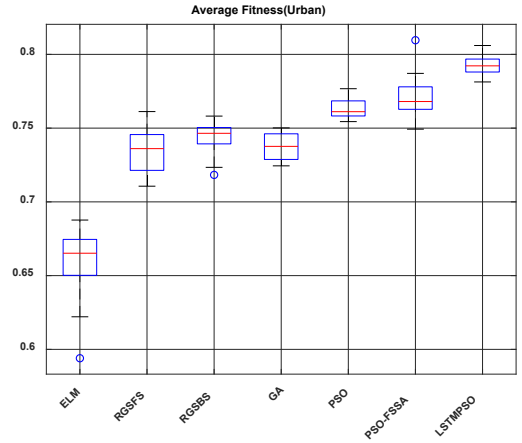
(g) Average fitness (Seismic)



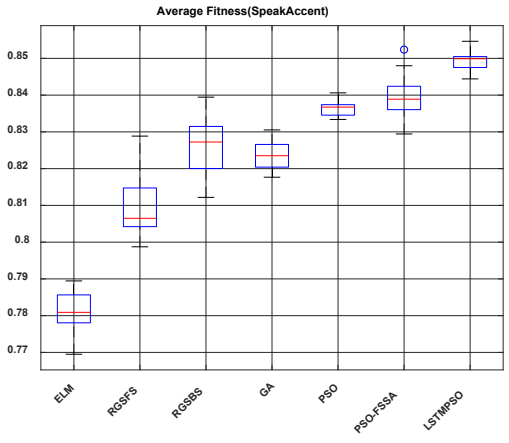
(h) Average fitness (ShillBidding)



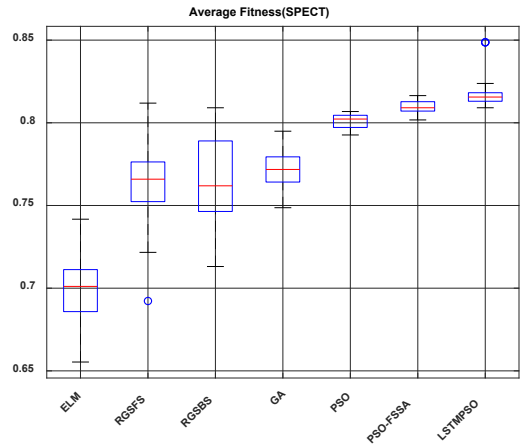
(i) Average fitness (SportsArticle)



(j) Average fitness (Urban)



(k) Average fitness (SpeakAccent)



(l) Average fitness (SPECT)

Fig. 4. Boxplot diagram for the distribution of average fitness results for each optimization algorithm + ELM-based over 24 runs

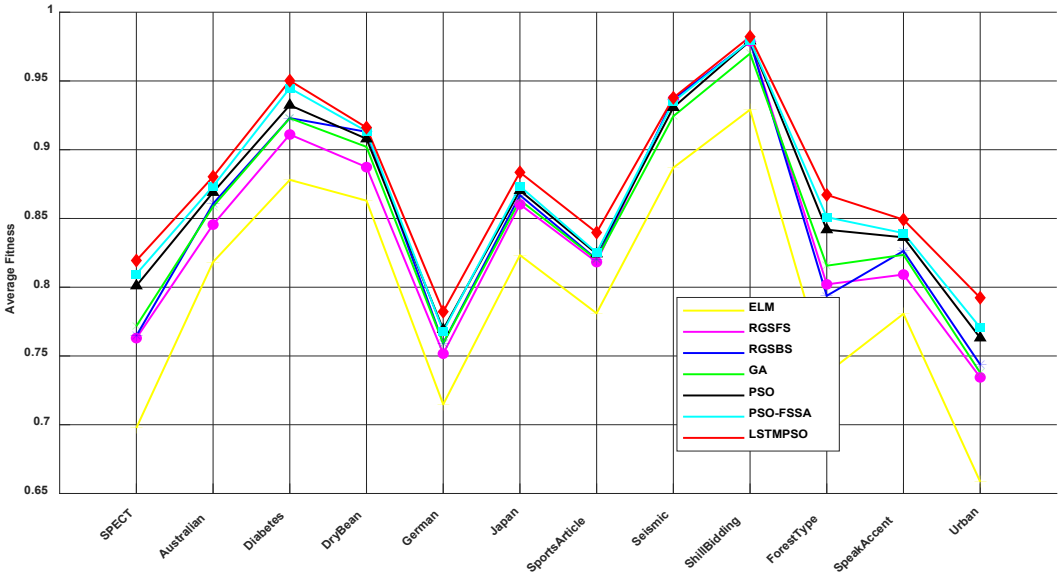


Fig. 5. Compared 24 runs average fitness results of the proposed method with other classical algorithm based on 12 UCI benchmark datasets

Table 3. Compared the accuracy (%) of the proposed method with other Feature Selection Model based on 20 UCI benchmark datasets

Datasets	KNN				MLP				Ours	Rank
	GA	PSO	HMOGA	BGSO	GA	PSO	HMOGA	BGSO		
BreastCancer	98.79	98.79	96.32	99	98.32	98.32	98.32	98.66	98.08	8
Glass	85.71	77.14	80.12	88.57	84.28	82.86	81.88	84.28	90.48	1
Hill-valley	54.76	54.76	51.5	55.68	54.22	55.31	53.22	56.04	87.13	1
Horse	97.06	97.05	97.05	100	100	100	100	100	94.12	9
Ionosphere	93.37	92.05	93.38	96.03	97.35	96.12	96.56	97.35	95.14	6
Madelon	57.33	54.17	60.33	59.67	60.17	57.83	59.8	60.05	61.83	?
Monk1	88.89	88.89	83.23	88.89	92.59	97.22	94.54	**100**	**100**	**1**
Monk2	74.77	74.77	55.09	74.77	81.94	74.31	69.21	67.13	**91.44**	**1**
Monk3	97.22	97.22	97.12	97.22	**100**	97.22	97	97.22	**100**	**1**
Sonar	56.72	58.21	68	79.1	76.11	80.59	77.12	79.1	**94.50**	**1**
Soybean-small	**100**	**100**	85.71	**100**	**100**	**100**	**100**	**100**	**100**	**1**
Vowel	89.61	88.74	87.85	88.53	91.77	89.83	87.25	88.74	65.37	9
Wine	97.87	**100**	70.21	**100**	**100**	**100**	99.98	**100**	**100**	**1**
Zoo	82.93	85.37	84	82.93	85.37	85.37	81.98	82.93	**99**	**1**
BreastEW	74.12	90.59	94.80	95.29	74.71	92.35	93.33	95.29	**98.21**	**1**
CongressEW	92.31	90.00	97.00	97.69	89.23	94.62	96.30	97.69	**97.97**	**1**
Exactly	91.50	69.50	72.00	89.00	**91.50**	69.25	88.25	90.75	78.8	6
Tic-tac-toe	82.77	73.63	78.00	82.77	80.68	74.41	75.00	82.25	**98.95**	**1**
PenglungEW	86.21	82.76	86.00	89.66	86.21	82.76	83.25	86.21	**94.29**	**1**
Avg	84.34	83.05	80.73	87.74	86.95	86.12	86.32	88	**92.31**	

4.4 Comparison with Others

The Table 3 showed the Highest classification accuracy which is obtained by GA, PSO, BGSO [36], HMOGA [37] using KNN (K-NearestNeighbor) and MLP (Multi-Layer Perception). In the Monk (1-3) datasets, our model achieves 100% accuracy in Monk1 and Monk3 and outperforms others nearly 10% accuracy in Monk2. In addition, we have made great strides in Hill-valley, Sonar, Zoo and Tic-tac-toe dataset. There is no free lunch in the classification model, our model also does not perform well in some datasets. In the horse dataset, there are many missing data and the filling effect is not ideal, so our feature selection model almost fails. There are 16 classes in Arrhythmia dataset and 11 classes in vowel dataset, which may be due to the small number of samples that makes our model perform poorly in multi classification problem. It should be noted that because the testset label of Madelon dataset is hidden, we use validset instead of testset, and the experimental results may be biased, which is not included in the ranking and average performance calculation. Generally speaking, as shown in Table 3, the LSTMPSO model proposed in this paper ranks first in 14 of 19 (excluding Madelon) datasets. It outperforms all Feature selection model in Table 3 by a significant margin of approximately 4.31%–11.58%.

5. Conclusion

In this paper, an LSTMPSO algorithm with long-term and short-term dependence mechanism is presented by strengthening the leadership mechanism of particle swarm optimization. This algorithm expands the time dimension of the PSO algorithm's leadership mechanism so that the population can take into account both long-term and short-term empirical learning, thereby improving the algorithm's ability to explore solution space. A comparison of 32 datasets on UCI public datasets shows that compared with the classical algorithm and the current advanced feature selection model, the algorithm can take into account both the model classification performance and redundant feature filtering capabilities. The quality of the optimal solution obtained by the search is also significantly improved.

References

[1] M. Dash, H. Liu, Feature selection for classification, Intelligent Data Analysis 1(1-4)(1997) 131-156.
[2] T.M. Cover, J.M. Van Campenhout, On the possible orderings in the measurement selection problem, IEEE Transactions on Systems, Man, and Cybernetics 7(9)(1977) 657-661.
[3] E. Amaldi, V. Kann, On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. Theoretical Computer Science 209(1-2)(1998) 237-260.
[4] A.W. Whitney, A direct method of nonparametric measurement selection, IEEE Transactions on Computers C-20(9)(1971) 1100-1103.
[5] T. Marill, D. Green, On the effectiveness of receptors in recognition systems, IEEE Transactions on Information Theory 9(1)(1963) 11-17.
[6] P. Pudil, J. Novoviova, J. Kittler, Floating search methods in feature selection, Pattern Recognition Letters 15(11)(1994) 1119-1125.
[7] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proc. ICNN'95 - International Conference on Neural Networks, 1995.
[8] W. Siedlecki, J. Sklansky, A note on genetic algorithms for large-scale feature selection, Pattern Recognition Letters 10(5) (1989) 335-347.
[9] M. Steinbrunn, G. Moerkotte, A. Kemper, Heuristic and randomized optimization for the join ordering problem, The VLDB Journal 6(3)(1997) 191-208.
[10]Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, in: Proc. the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), 1999.
[11]J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proc. 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, 1997.
[12]Q. Shen, W.M. Shi, W. Kong, Hybrid particle swarm optimization and tabu search approach for selecting genes for tumor classification using gene expression data, Computational Biology and Chemistry 32(1)(2008) 53-60.
[13]B. Xue, M. Zhang, W.N. Browne, Particle swarm optimization for feature selection in classification: a multi-objective approach, IEEE Transactions on Cybernetics 43(6)(2013) 1656-1671.
[14]J. Vijaya, E. Sivasankar, An efficient system for customer churn prediction through particle swarm optimization based feature selection model with simulated annealing, Cluster Computing 22(suppl. 5)(2019) 10757-10768.
[15]M. Clerc, J. Kennedy, The particle swarm - explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6(1)(2002) 58-73.
[16]M.S. Mohamad, S. Omatu, S. Deris, M. Yoshioka, A modified binary particle swarm optimization for selecting the small

subset of informative genes from gene expression data, IEEE Transactions on Information Technology in Biomedicine 15(6)(2011) 813-822.
[17]S. Naka, T. Genji, T. Yura, Y. Fukuyama, Practical distribution state estimation using hybrid particle swarm optimization, in: Proc. 2001 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.01CH37194), 2001.
[18]T. Peram, K. Veeramachaneni, C.K. Mohan, Fitness-distance-ratio based particle swarm optimization, in: Proc. the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706), 2003.
[19]Y. Shi, R.C. Eberhart, Parameter selection in particle swarm optimization, in: Proc. International Conference on Evolutionary Programming, 1998.
[20]B. Xue, M. Zhang, W.N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, IEEE Transactions on Evolutionary Computation 20(4)(2016) 606-626.
[21]B. Chakraborty, Genetic algorithm with fuzzy fitness function for feature selection, in:. Proc. the 2002 IEEE International Symposium on Industrial Electronics, 2002. ISIE 2002, 2002.
[22]P.P. Kundu, S. Mitra, Multi-objective evolutionary feature selection, in: Proc. International Conference on Pattern Recognition and Machine Intelligence, 2009.
[23]X. Wang, J. Yang, X. Teng, W. Xia, R. Jensen, Feature selection based on rough sets and particle swarm optimization, Pattern Recognition Letters 28(4)(2007) 459-471.
[24]L. Cervante, B. Xue, M. Zhang, L. Shang, Binary particle swarm optimisation for feature selection: A filter based approach, in: Proc. 2012 IEEE Congress on Evolutionary Computation, 2012.
[25]A. Unler, A. Murat, R.B. Chinnam, mr^{2}PSO: A maximum relevance minimum redundancy feature selection method based on swarm intelligence for support vector machine classification, Information Sciences 181(20)(2011) 4625-4641.
[26]E. Alba, J. Garcia-Nieto, L. Jourdan, E. Talbi, Gene selection in cancer classification using pso/svm and ga/svm hybrid algorithms, in: Proc. 2007 IEEE Congress on Evolutionary Computation, 2007.
[27]A. Unler, A. Murat, A discrete particle swarm optimization method for feature selection in binary classification problems, European Journal of Operational Research 206(3)(2010) 528-539.
[28]K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE Transactions on Evolutionary Computation 6(2)(2002) 182-197.
[29]A. Khan, A. Baig, Multi-objective feature subset selection using non-dominated sorting genetic algorithm, Journal of Applied Research and Technology 13(1)(2015) 145-159.
[30]H. Soyel, U. Tekguc, H. Demirel, Application of nsga-ii to feature selection for facial expression recognition, Computers & Electrical Engineering 37(6)(2011) 1232-1240.
[31]B. Huang, B. Buckley, T.M. Kechadi, Multi-objective feature selection by using nsga-ii for customer churn prediction in telecommunications, Expert Systems with Applications 37(5)(2010) 3638-3646.
[32]P. Ghamisi, J.A. Benediktsson, Feature selection based on hybridization of genetic algorithm and particle swarm optimization, IEEE Geoscience and Remote Sensing Letters 12(2)(2015) 309-313.
[33]Y. Zhang, D.W. Gong, J. Cheng, Multi-objective particle swarm optimization approach for cost-based feature selection in classification, IEEE/ACM Transactions on Computational Biology and Bioinformatics 14(1)(2017) 64-75.
[34]G.B. Huang, Q.Y. Zhu, K. Mao, C.K. Siew, P. Saratchandran, N. Sundararajan, Can threshold networks be trained directly?, IEEE Transactions on Circuits and Systems II: Express Briefs 53(3)(2006) 187-191.
[35]G.B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42(2)(2012) 513-529.
[36]M. Ghosh, R. Guha, I. Alam, P. Lohariwal, D. Jalan, R. Sarkar, Binary genetic swarm optimization: A combination of ga and pso for feature selection, Journal of Intelligent Systems 29(1)(2020) 1598-1610.
[37]V. Bhateja, C. Coello Coello, S. Satapathy, P. Pattnaik (Eds), Intelligent Engineering Informatics. Advances in Intelligent Systems and Computing, vol 695, Springer, Singapore, 2018 (pp. 471-479).