BSO-FS: Bee Swarm Optimization for Feature Selection in Classification

Souhila Sadeg[1(✉)], Leila Hamdad[2], Karima Benatchba[3], and Zineb Habbas[4]

[1] Ecole nationale Supérieure d'Informatique, Oued Smar, Algiers, Algeria
s_sadeg@esi.dz
[2] LCSI, Ecole nationale Supérieure d'Informatique, Oued Smar, Algiers, Algeria
l_hamdad@esi.dz
[3] LMCS, Ecole nationale Supérieure d'Informatique, Oued Smar, Algiers, Algeria
k_benatchba@esi.dz
[4] LCOMS, Université de Lorraine, Metz, France
zineb.habbas@univ-lorraine.fr

Abstract. Feature selection is an important data-preprocessing step that often precedes the classification task. Because of large amount of features in real world applications, feature selection is considered as a hard optimization problem. For such problems, metaheuristics have been shown to be a very promising solving approach. In this work, we propose to use Bee Swarm Optimization (BSO) for feature selection. The proposed algorithm, BSO-FS, is based on the wrapper approach that uses BSO for the generation of feature subsets, and a classifier algorithm to evaluate the solutions. BSO-FS is tested on well-known datasets and its performances are compared with those of recently published methods. Obtained results show that for the majority of datasets, BSO-FS selects efficiently relevant features while improving the classification accuracy.

Keywords: Bee Swarm Optimization · Metaheuristic · Feature selection · Wrapper approach · Classification · Data mining

1 Introduction

Classification is a supervised learning task that is used in many real world problems. Its purpose is "to slot objects in a population into one or more categories based on a set of features measured on each object" [1]. To achieve this goal, a function (classifier) is inferred from labeled training dataset in order to correctly determine the class (category) of new examples.

In real-world application, especially when one is faced with unknown situations, many features are usually introduced in order to have the best possible representation without knowing, a priori, which of them are relevant. However, irrelevant or redundant features can lead to a classification accuracy decrease and to an unnecessary increase of computational cost [24][25]. Indeed, one might think that having more features would result in more discriminating power whereas, in practice, adding irrelevant or distracting attributes to a dataset often confuses

I. Rojas et al. (Eds.): IWANN 2015, Part I, LNCS 9094, pp. 387–399, 2015.
DOI: 10.1007/978-3-319-19258-1_33

machine learning systems, although most of these algorithms are designed to learn which are the most appropriate attributes for making their decisions [2]. Therefore, selecting a subset of relevant features among a large initial set, often allows yielding better classification results. In addition, it offers other advantages such as facilitating data visualization and data understanding, reducing the measurement and storage requirements, reducing training and execution times, defying the curse of dimensionality to improve prediction performance [26].

Feature subset selection is usually considered as a data-preprocessing task that precedes a data mining one. It is defined by Kira and Rendell [31] as the problem of choosing a small subset of data attributes for a particular application that is, ideally, necessary and sufficient to describe the target. It can be achieved in two ways: One is to rank features according to some criterion and select the top k features. The other one is to select the smallest subset of features that does not deteriorate learning performance[3].

According to [30], a feature selection process, as illustrated in figure 1, mainly consists of four steps : subset generation, subset evaluation, stopping criterion, and result validation. The generation procedure is a search procedure that generates subsets of features. Each of them is evaluated according to the evaluation function in order to save the best one. This procedure is repeated iteratively until a stopping criterion is reached. It can be a predefined number of features, a predefined number of iterations, or some value of the evaluation function. Feature

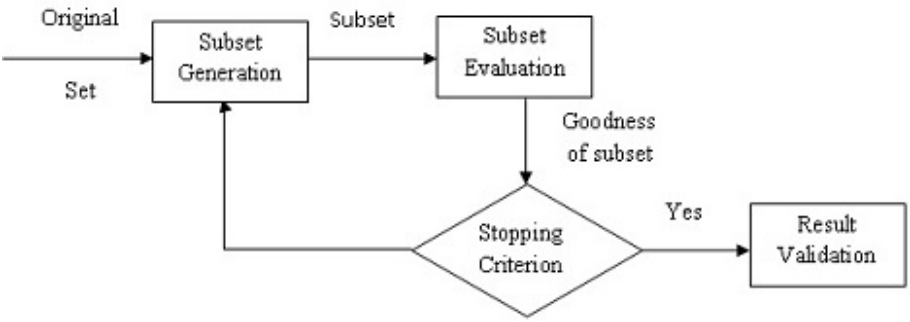

Fig. 1. Key steps for feature selection [30]

selection methods are generally divided into three categories according to how and when the relevance of the selected features is evaluated [4]. In other words, a feature selection method falls into one of the three categories depending on its use (or not) of a learning algorithm (a classifier) and on the way it uses it. In the filter approach, the algorithm selects the features by evaluating them while analyzing the general characteristics of data without involving any learning [4]. On the other hand, in the wrapper and embedded approaches, the feature selection method uses a learning algorithm but differ in the way they interact. Wheras

embedded approach integrates it in model building[3], the wrapper approach searches through the feature subset space and computes the estimated accuracy of a learning algorithm [8,11]. The feature space can be explored by using various strategies. Usually an exhaustive search is too expensive. Indeed, if the cardinality of the original set is N, finding the best subset requires to search among the 2^N candidate subsets which is computationally intractable [30]. Therefore, methods based on heuristics or random search are used to find good solutions in reasonable time following an evaluation function and a stopping criterion.

Among existing works on feature selection based on the wrapper approach and using metaheuristics, those using the genetic algorithm are the most frequent [6–9]. Metaheuristics based on Swarm Intelligence have also been applied to feature selection such as Ant Colony Optimization (ACO)[16,17,27,28], and Particle Swarm Optimization (PSO) [10,18].

In this paper, we propose a feature selection algorithm that follows the wrapper approach and uses the Bee Swarm Optimization algorithm (BSO) [12] as search algorithm. BSO is a swarm intelligence algorithm inspired from the foraging behavior of natural bees that has been successfully applied to various optimization problems [15,20–23]. Its good performances can be explained by a good balance between intensification and diversification, which leads respectively to a good exploitation and exploration of the search space. Here, we apply it for the first time to the feature selection problem with the aim to find the best feature subset that maximizes the classification accuracy. Experiments are conducted on a large number of data sets and the results are compared with those of PSO (Particle Swarm Optimization), ACO (Ant Colony Optimization), GA (Genetic Algorithm) and ABC (Artificial Bee Colony). The comparison showed that for most tested datasets, BSO outperformed the other methods.

The rest of the paper is organized as follows: In sections 2, the general algorithm of BSO is presented and its application for feature selection is described in section 3. Section 4 shows the results of BSO-FS and compares them to the results of other methods before we conclude with final remarks and directions for future works in section 5.

2 Bee Swarm Optimization Metaheuristic

Honeybee swarm in nature has a very interesting behavior characterized by collective intelligence that allows self-adaptation to the environment and dynamic tasks assignment. During the last decade, many new algorithms inspired by natural bees behavior were proposed and applied to different problems. A survey of these algorithms is given in [13].

Bee Swarm Optimization (BSO) metaheuristic proposed in [12] is inspired from the foraging behavior of real bees. Many researchers[5,14] have studied this behavior, considered as the most important task in the hive. In its foraging process, a bee starts by leaving the hive in order to find a flower and gather nectar. Then, it returns to the hive and unloads the nectar. If the food source is rich, the bee communicates its direction and distance to its mates via a dance. The other bees naturally follow the one that indicates the best food source.

BSO is an iterative search process based on a population of artificial bees cooperating to solve an instance of an optimization problem by imitating the foraging behavior described above. First, an initial solution is generated randomly or via a heuristic. This solution will be the reference solution, *refSol*, from which a search region, a set of candidate solutions, is determined. After that, each solution is assigned to a bee as a starting point of a local search. At the end of the search, each bee communicates its best found solution to the other ones through a table, named *dance*. One of the solutions in this table will be selected to be the reference solution in the next iteration. In order to avoid cycles, the reference solutions are stored in a Tabu list.

To ensure a good balance between exploitation and exploration, the selection of the reference solution follows the intensification and diversification principles. The first one aims to find good solutions by exploiting promising search region while the second allows a good coverage of the search space by visiting new regions. Intensification and diversification are performed according to the results obtained in a search region during a certain time: while the best global solution (bestGlobalSol) is improved, an intensification is performed by selecting a reference solution among those in dance table. Otherwise, either a diversification is performed, in case the search region has reached the maximum number of chances granted to it, or an intensification is performed and the number of chances is decreased. A diversification consists in choosing a solution from dance which is the furthest from all the solutions stored in the Tabu list. If all the solutions of dance are in Tabu list, a random solution is generated to be the next reference solution. The algorithm stops when the optimal solution is found or the maximum number of iterations is reached. The BSO general algorithm is described bellow.

3 BSO-FS: BSO for Feature Selection

In this paper, we propose a wrapper approach based method for feature selection that uses the BSO metaheuristic to perform the search process of subsets. Applying BSO to feature selection requires the adaptation of the general algorithm to the specificities of the problem. In this section, we will define how a solution is encoded, how its quality is evaluated, and how the search region is determined. The selection of the reference solution and the local search performed by the bees do not differ from the general algorithm given in the previous section.

- **Solution encoding**. A solution is represented by a binary vector of length n, where n is the original number of features. A position of the vector is set to 1 if the corresponding feature is selected and to 0 otherwise.
- **Fitness**. It represents the quality of the solution and is noted f(s). In our work, it is equivalent to the classification accuracy returned by the used classifier and is calculated by the following formula:

$$accuracy = \frac{number\,of\,true\,positive\ +\ number\,of\,true\,negative}{total\,population} \tag{1}$$

Algorithm 1. General algorithm BSO

```
Input: An instance of a combinatorial optimization problem
Output: The best solution found
refSol ← an initial solution found randomly or via a heuristic
while not stopping criterion do
    Insert refSol in Tabu list
    Determine searchRegion from refSol
    nbChances ←maxChances
    Assign a solution from searchRegion to each bee
    for each bee k do
        Perform a local search
        Store the result in the table dance
    if f(bestSol) > f(bestGlobalSol) then
        bestGlobalSol ←bestSol
        nbChances ←maxChances
        Intensification
    else
        if nbChances >0 then
            nbChances ←nbChances-1
            Intensification
        else
            diversification
return bestGlobalSol
```

Note that if two solutions have the same fitness, the one that uses less features is considrered to be the best.

– **Search region**. It is a set of solutions generated from the reference solution by flipping in refSol a number of bits equal to n/flip, flip being an empirical parameter. The size of this set equals the number of bees since each solution will be assigned to one bee as a starting point of its local search (see Figure 2).The value of flip has an impact on the performance of the research process because it determines the distance between refSol and the solutions which define the search region . Indeed, a too small value will favor the exploration instead of exploitation of the search space, while a too high value might lead the algorithm to converge to a local optimum.

To obtain the solutions of the search region we use two strategies that ensure that the obtained solutions are as distinct as possible. In the first one, the k^{th} solution is generated by flipping in refSol the variables separated by flip bits starting at the k^{th} variable. As an example, let n=20 and flip = 5. If the variables are subscripted from 0 to 19, then, as illustrated in figure 3, solutions s_0, s_1, s_2, s_3 and s_4 are obtained respectively by flipping the following bits: (0,5,10,15), (1,6,11,16), (2,7,12,17), (3,8,13,18) and (4,9,14,19). In the second strategy, a solution number k is obtained while flipping $n/flip$ contiguous bits starting by the k^{th} bit. Following the previous example the solutions s_0, s_1, s_2, s_3 and s_4

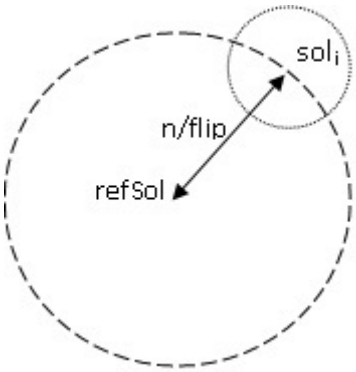

Fig. 2. Search region

are obtained respectively by flipping the following bits: (0,1,2,3) , (4,5,6,7) , (8,9,10,11) , (12,13,14,15) and (16,17,18,19).

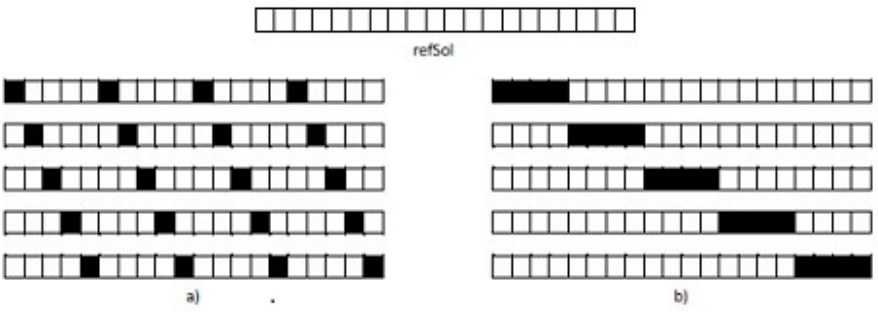

Fig. 3. a- solutions generated by the first strategy, b- solutions generated by the second strategy

4 Experimental Results

In order to investigate the performance of BSO-FS on feature selection, we implemented it in java programming language and used weka[1] and LibSVM[2] to execute the classification. Experiments were conducted on a personal computer running windows 7 System, and equipped with an Intel Core i3 2.20 GHz CPU and 6 GB RAM. In this section, we present the datasets used in the experimentations

[1] http://www.cs.waikato.ac.nz/ml/weka/
[2] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

then show the results obtained with BSO-FS. These results are compared to recent published results of GA, ACO, PSO and ABC metaheuristics. Only classification accuracies are compared since the algorithms are not run on similar machines.

4.1 Datasets

To evaluate our algorithm, we used Seventeen well-known benchmark classification datasets available in UCI Machine Learning Repository[3] that are frequently studied in machine learning. We considered datasets with considerable diversity in characteristics that are the number of features, classes and instances. According to the number of features, we considered small-sized datasets with less than 19 features (Iris, Breast Cancer, Glass, Diabetes, Heart-C, Heart-StatLog, Labor, Wine, Vehicle, Zoo, Labor), medium-sized datasets having between 20 and 49 features (German, Auto, WBCD, Ionosphere) and large sized datasets having more than 50 features (Lung cancer and Sonar). A summary of these datasets is shown in table 1.

Table 1. Datasets characteristics

Dataset	features	classes	instances	Dataset	features	classes	instances
Iris	4	2	150	Zoo	17	7	101
Breast-cancer	9	2	286	German	24	2	1000
Glass	9	7	214	WBCD	30	2	569
Diabetes	8	2	768	Wine	13	3	178
Heart-C	13	5	303	Ionosphere	34	2	351
Heart-StatLog	13	2	270	Vehicle	18	4	846
Hepatisis	19	2	155	Lung cancer	56	3	32
Labor	16	2	57	Sonar	60	2	208
Auto	25	7	205				

4.2 Results of BSO-FS

Like most metaheuristics, the performances of BSO-FS are greatly impacted by the values of its parameters. In our experiments, the optimal values were determined by performing a considerable number of runs using several values. Thus, the parameters were set to the following values: Flip = 5, MaxChances = 3 and Number of bees = 10 (or equal to the number of features if the latter is less than 10). Table 2 shows the accuracies obtained with BSO-FS using KNN classifier and applying 1 x 10 fold cross validation. For each dataset, we give the initial number of features, the accuracy obtained using all the features, the average classification accuracy over 10 runs, the average number of selected features, the best classification accuracy and the corresponding number of selected features.

[3] http://archive.ics.uci.edu/ml/

Table 2. Results of BSO using KNN classifier

Dataset	Features	Accuracy	Ave Accuracy	Ave Features	Best Accuracy	Features
Iris	4	95.33	96	2	96	2
Breast-cancer	9	72.38	76.57	5.3	76.57	4
Glass	9	70.56	79.44	5	79.44	5
Diabetes	8	70.18	71.48	3	71.48	3
Heart-C	13	76.24	83.50	6	83.50	6
Heart-StatLog	13	73.43	84.13	3	84.13	3
Hepatisis	19	80.64	90.97	7	90.97	7
Labor	16	82.46	98.25	8.8	98.25	7
Auto	25	76.1	87.80	6.3	87.80	5
Zoo	17	96.04	99.01	11.8	99.01	11
German	24	72	75.6	7	75.6	7
WBCD	30	95.96	98.24	14.3	98.24	11
Wine	13	94.94	99.44	7	99.44	7
Ionosphere	34	86.32	95.95	11.2	96.01	10
Vehicle	18	69.86	74.56	9.8	74.59	8
Lung-cancer	56	68.75	94.68	19.6	96.87	14
Sonar	60	86.54	98.22	27.3	98.56	23

The numerical results show that for all the benchmarks, the best accuracies are reached with reduced sets of features. For some datasets such as auto and lung cancer, the reduction rate is very significant. Also note that for almost all the benchmarks, the average results accuracies are equal to the best ones and this means that BSO-FS is a very stable algorithm thanks to the fact that it rarely uses randomness.

In order to analyze the behavior of BSO-FS, we perform 10 executions where the number of iterations equals 10. The fitness of the reference solutions are represented by a curve that illustrates the progression of the algorithm. The curves of 10 executions on lung-cancer dataset are represented in figure 4.

We can see that BSO-FS always reaches a very good solution (often its best solution) very quickly and maintains the same quality level until the end of the execution. The curves show some reference solution of lower quality that can be explained by a diversification performed by the algorithm.

4.3 Comparison with Other Algorithms

In order to further test the performances of BSO-FS, we compare our results with four recently published works based on the wrapper approach and using metaheuristics [10,18,19]. As these works do not use the same classifier, we carried out two series of tests using SVM and KNN classifiers. Table 3 compares the results of BSO-FS with those of PSO (Particle Swarm Optimization), ACO (Ant Colony Optimization), GA (Genetic Algorithm) and ABC (Artificial Bee Colony) using SVM [19]. The results show that BSO reaches better average accuracies than the others for 6 among the 9 datasets used in this part of the experiments. Indeed, ABC reached better results for Glass (71.50 %) and Labor

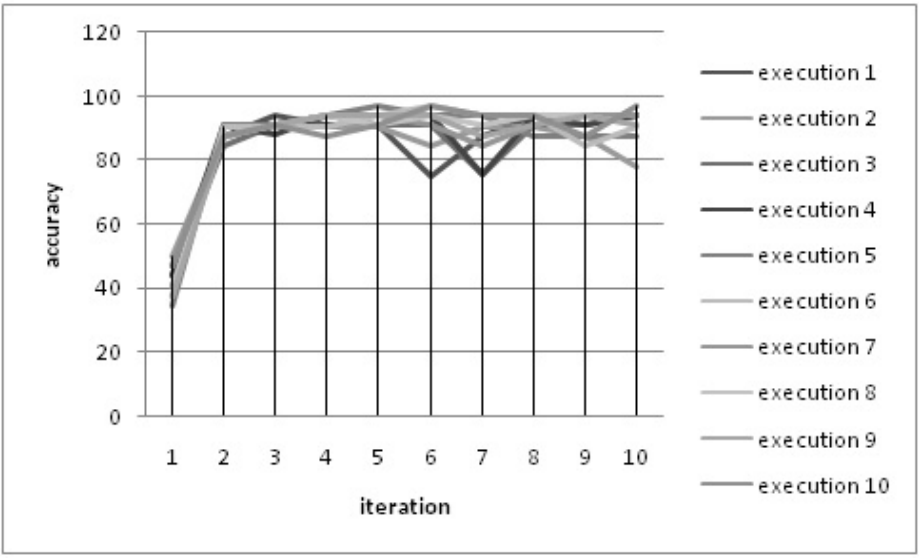

Fig. 4. Accuracies of ten solutions obtained on 10 executions on Lung-Cancer dataset

(98.26 %) for which BSO-FS obtained a very close result equal to 98.25 %. For the Diabetes dataset, we can observe that PSO, GA and ACO reached 75.65 % against 72.53 % for BSO-FS and 71.48 % for ABC. Nevertheless, we note that PSO, GA and ACO use all the features (8) whereas BSO-FS uses only 2 features and ABC only one feature. For the other datasets (Iris, Breast Cancer, Heart-C, Heart-StatLog, Hepatitis, Auto), BSO-FS obtained the best results. We can notice that the best accuracies are often obtained with the smallest subsets of features, the best example is the Auto dataset for which BSO-FS reached 86.34 % with only six features among the 25 constituting the original set.

In table 4, we present a comparison between our results and two variants of PSO recently published in literature [10,18]. It can be seen that the best classification accuracies are obtained with BSO-FS for all the benchmarks except the Vehicle dataset. We can also note that for the datasets that have the largest numbers such as Ionosphere, Sonar and Lung cancer, BSO-FS is much more efficient than PSO based methods.

5 Conclusion

In this paper, we proposed a wrapper approach based method for feature selection problem. Our method, named BSO-FS, uses the BSO metaheuristic to perform the search process in its subset generation step. The results show that for all datasets, reduced number of features allow to achieve better classification accuracy than the initial set of features. We also observed that, some equal size subsets could give different results, that means that some features are more

Table 3. Comparison with other algorithms using SVM classifier

Dataset	features	Algorithm	Ave features	Ave Accuracy
		BSO-FS	3	97.33
		PSO	4	96.66
Iris	4	ACO	4	96.66
		GA	4	96.66
		ABC	3	97.33
		BSO-FS	3.4	75.87
		PSO	8	73.08
Breast-cancer	9	ACO	9	73.08
		GA	8	73.08
		ABC	4	75.87
		BSO-FS	7	71.03
		PSO	8	71.03
Glass	9	ACO	8	71.03
		GA	8	71.03
		ABC	6	71.50
		BSO-FS	2	72.53
		PSO	8	75.65
Diabetes	8	ACO	8	75.65
		GA	8	75.65
		ABC	1	71.48
		BSO-FS	6	84.16
		PSO	8	83.17
Heart-C	13	ACO	7	80.86
		GA	7	80.20
		ABC	7	83.17
		BSO-FS	3.4	84.87
		PSO	8	82.96
Heart-Statlog	13	ACO	7	81.11
		GA	8	73.70
		ABC	3	84.81
		BSO-FS	8.7	87.74
		PSO	7	86.45
Hepatitis	19	ACO	7	83.23
		GA	7	83.26
		ABC	9	87.10
		BSO-FS	7.8	98.25
		PSO	5	89.47
Labor	16	ACO	6	92.98
		GA	9	89.47
		ABC	8	98.26
		BSO-FS	6	86.34
		PSO	8	68.78
Auto	25	ACO	9	72.20
		GA	9	69.21
		ABC	9	82.93

Table 4. Comparison with other algorithms using KNN classifier

Dataset	features	Algorithm	Ave features	Ave Accuracy	best accuracy	standard deviation
Zoo	17	BSO-FS	11.8	99.01	99.01	0.00
		PSO [10]	6.46	95.52	97.14	1.62
German	24	BSO-FS	7	75;6	75.6	0.00
		PSO [10]	12.76	68.47	70.67	2.20
WBCD	30	BSO-FS	14.3	98.24	98.24	0.00
		PSO [18]	14.2	98.17	97.24	0.07
		PSO [10]	3.46	93.98	94.74	0.76
Wine	13	BSO-FS	7	99.44	99.44	0.00
		PSO [18]	8	99.44	99.44	0.00
		PSO [10]	6.84	95.26	98.77	3.51
Ionosphere	34	BSO-FS	11.2	95.95	96.01	0.06
		PSO [10]	3.26	87.27	91.43	4.16
Vehicle	13	BSO-FS	7	99.44	99.44	0.00
		PSO [18]	8	99.44	99.44	0.00
		PSO [10]	6.84	95.26	98.77	3.51
Lung-Cancer	56	BSO-FS	19.6	94.68	96.87	2.19
		PSO [10]	6.74	78.4	90	11.60
Sonar	13	BSO-FS	27.3	98.22	98.56	0.34
		PSO [18]	30.2	96.92	97.12	0.20
		PSO [10]	11.24	78.16	85.71	7.55

relevant when they are associated with others. The study shows that BSO-FS is an effective search algorithm. Indeed, It gives very promising results after few iterations of the search process. The results show besides that BSO-FS is highly stable since it generally gives the same results at each execution, which is translated by a very small standard deviation often equal to zero. This is explained by a good balance between exploitation and exploration ensured by a good compromise between intensification and diversification. The application of these fundamental principles of BSO is allowed by a right setting of the Flip and MaxChances parameters. Finally, the comparison of the results of BSO-FS with the results of other recently published metaheuristics shows that our algorithm gives generally best accuracies with good reduction rates and very little deviations. In our future works, we plan to investigate hybridization of BSO-FS with filter algorithms in the phase of initialization and to parallelize the search process of the bees in order to perform experiments on larger datasets.

References

1. Clarke, B., Fokoue, E., Zhang, H.H.: Principles and theory for data mining and machine learning. Springer Science Business Media (2009)
2. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2005)
3. Liu, H., Motoda, H. (eds.).: Computational methods of feature selection. CRC Press (2007)
4. Liu, H., Motoda, H., Setiono, R., Zhao, Z.: Feature Selection: An Ever Evolving Frontier in Data Mining. FSDM **10**, 4–13 (2010)

5. Von Frisch, K., Lindauer, M.: The "Language" and Orientation of the Honey Bee. Annual Review of Entomology **1**, 45–58 (1956, 1973)
6. Oliveira, L.S., Sabourin, R., Bortolozzi, F., Suen, C.Y.: A methodology for feature selection using multiobjective genetic algorithms for handwritten digit string recognition. International Journal of Pattern Recognition and Artificial Intelligence **17**(06), 903–929 (2003)
7. Inza, I., Merino, M., Larraaga, P., Quiroga, J., Sierra, B., Girala, M.: Feature subset selection by genetic algorithms and estimation of distribution algorithms: a case study in the survival of cirrhotic patients treated with TIPS. Artificial Intelligence in Medicine **23**(2), 187–205 (2001)
8. Yang, J., Honavar, V.: Feature subset selection using a genetic algorithm. In: Feature Extraction, Construction and Selection, pp. 117–136. Springer, US (1998)
9. Huang, J., Cai, Y., Xu, X.: A hybrid genetic algorithm for feature selection wrapper based on mutual information. Pattern Recognition Letters **28**(13), 1825–1844 (2007)
10. Xue, B., Zhang, M., Browne, W.N.: Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. Applied Soft Computing **18**, 261–276 (2014)
11. Yusta, S.C.: Different metaheuristic strategies to solve the feature selection problem. Pattern Recognition Letters **30**(5), 525–534 (2009)
12. Drias, H., Sadeg, S., Yahi, S.: Cooperative bees swarm for solving the maximum weighted satisfiability problem. In: Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) IWANN 2005. LNCS, vol. 3512, pp. 318–325. Springer, Heidelberg (2005)
13. Karaboga, D., Akay, B.: A survey: algorithms simulating bee swarm intelligence. Artificial Intelligence Review **31**(1–4), 61–85 (2009)
14. Seeley, T.D.: Honeybee ecology: a study of adaptation in social life. Princeton University Press (2014)
15. Sadeg, S., Drias, H.: A selective approach to parallelise Bees Swarm Optimisation metaheuristic: application to MAX-W-SAT. International Journal of Innovative Computing and Applications **1**(2), 146–158 (2007)
16. Kabir, M.M., Shahjahan, M., Murase, K.: A new hybrid ant colony optimization algorithm for feature selection. Expert Systems with Applications **39**(3), 3747–3763 (2012)
17. Robbins, K.R., Zhang, W., Bertrand, J.K., Rekaya, R.: The ant colony algorithm for feature selection in high-dimension gene expression data for disease classification. Mathematical Medicine and Biology **24**(4), 413–426 (2007). ISO 690
18. Chuang, L.Y., Tsai, S.W., Yang, C.H.: Improved binary particle swarm optimization using catfish effect for feature selection. Expert Systems with Applications **38**(10), 12699–12707 (2011)
19. Schiezaro, M., Pedrini, H.: Data feature selection based on Artificial Bee Colony algorithm. EURASIP Journal on Image and Video Processing **2013**(1), 1–8 (2013)
20. Belkebir, R., Guessoum, A.: A hybrid BSO-Chi2-SVM approach to Arabic text categorization. In: 2013 ACS International Conference on Computer Systems and Applications (AICCSA), pp. 1–7. IEEE, May 2013
21. Drias, H., Mosteghanemi, H.: Bees swarm optimization based approach for web information retrieval. In: 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), vol. 1, pp. 6–13. IEEE, August 2010

22. Djeffal, M., Drias, H.: Multilevel bee swarm optimization for large satisfiability problem instances. In: Yin, H., Tang, K., Gao, Y., Klawonn, F., Lee, M., Weise, T., Li, B., Yao, X. (eds.) IDEAL 2013. LNCS, vol. 8206, pp. 594–602. Springer, Heidelberg (2013)
23. Djenouri, Y., Drias, H., Chemchem, A.: A hybrid bees swarm optimization and tabu search algorithm for association rule mining. In: 2013 World Congress on Nature and Biologically Inspired Computing (NaBIC), pp. 120–125. IEEE, August 2013
24. Blum, A.L., Langley, P.: Selection of relevant features and examples in machine learning. Artificial intelligence **97**(1), 245–271 (1997)
25. Koller, D., Sahami, M.: Toward optimal feature selection (1996)
26. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. The Journal of Machine Learning Research **3**, 1157–1182 (2003)
27. Alsukker, A., Khushaba, R., Al-Ani, A., Al-Jumaily, A.A.: Enhanced feature selection algorithm using ant colony optimization and fuzzy memberships. IASTED (2008)
28. Ke, L., Feng, Z., Ren, Z.: An efficient ant colony optimization approach to attribute reduction in rough set theory. Pattern Recognition Letters **29**(9), 1351–1357 (2008)
29. Ke, L., Feng, Z., Ren, Z.: An efficient ant colony optimization approach to attribute reduction in rough set theory. Pattern Recognition Letters **29**(9), 1351–1357 (2008)
30. Dash, M., Liu, H.: Feature selection for classification. Intelligent data analysis **1**(3), 131–156 (1997)
31. Kira, K., Rendell, L.A.: The feature selection problem: traditional methods and a new algorithm. In: AAAI, vol. 2, pp. 129–134, July 1992