An Efficient Feature Selection Using Ant Colony Optimization Algorithm

Md. Monirul Kabir[1], Md. Shahjahan[2], and Kazuyuki Murase[1]

[1] Department of Human and Artificial Intelligence Systems
University of Fukui, Bunkyo 3-9-1, Fukui 910-8507, Japan
{kabir,murase}@synapse.his.fukui-u.ac.jp
[2] Department of Electrical and Electronic Engineering
Khulna University of Engineering and Technology, Khulna 9203, Bangladesh
jahan@mail.kuet.ac.bd

Abstract. This paper presents an efficient feature selection algorithm by utilizing the strategy of ant colony optimization, called as ACOFS. Initially, ACOFS uses a modified framework to guide the ants in the right directions while constructing the graph (subset) paths. In the subsequent part, a set of new modified pheromone update rules as well as a set of new modified estimation of heuristic information for features are introduced. The effect of such modifications ultimately assists ants to generate salient feature subsets with reduced size. We evaluate the performance of ACOFS on four real-world benchmark datasets. The experimental results show that ACOFS has a remarkable capability to generate reduced size subsets of salient features with yielding significant classification accuracies.

Keywords: Feature selection, pheromone update, heuristic information, neural network, classification accuracy.

1 Introduction

Feature selection (FS) has a crucial role in data mining. The aim of FS is to reduce the dimension of original feature set by identifying the spurious features properly which ultimately provides the best performances under some classification datasets. However, searching of spurious features by their dependency on the classification algorithm can be divided into two categories, such as: filters and wrappers [1], [2].

In accordance with the researchers, FS is basically a search process. In finding the salient features, search can be categorized by forward search [3] and backward search [4]. Furthermore, search can also be carried out by genetic algorithm [5] or genetic programming [6], called genetic search. Apart from these, there are also some other search techniques in FS which are: Tabu seach, Simulated annealing, and so on. Ant colony optimization (ACO) is an algorithm that is inspired by social behavior of ant colonies and has already been used in many combinatorial optimization problems for optimum solutions. For instance, ACO in TSP [7], QAP [8], routing in telecommunication networks, graph coloring problem, and so on. In case of FS problems, recently a number of approaches have been introduced [9]-[15].

© Springer-Verlag Berlin Heidelberg 2009

In this paper, we propose an efficient FS algorithm, called ant colony optimization based FS algorithm (ACOFS). The main focus of this algorithm is to modify the inner operation of the ACO algorithm according to the activity of wrapper tool rather than filter tool. It is reasonable in the sense that the performance of wrapper model is outperformed filter model [2][3]. The proposed technique adapts a set of new modified pheromone update rules and heuristic information estimations based on the evaluation of constructed subsets by means of trained neural network (NN) classifier. Thus ACOFS differs from previous works (e.g., [9]-[15]) on three vital aspects.

First, ACOFS emphasizes on deciding the size of the subset in a restricted random fashion while ants attempt to traverse upon the node (or, feature) space to construct a path (or, subset). This subject is represented by means of a modified framework mentioned in Section 2.1. Such approach is quite different from the existing works [9][10][14] in view of guiding ants to be used the forward selection strategy (FSS) during subset constructions (SCs). It is needed to say here that FSS always suffers by two problems: (a) a suitable stopping criterion is necessary to stop the SC; (b) proper guidance is required for adding or deleting features during SC. However, to solve the above problems, a strategy of fixed size is incorporated in [12][13][15] which is augmented by a constant rate or randomly chosen, but needs suitable boundary in between upper and lower limit. Thus deciding the subset size randomly within a constrained limit may be a novel approach in this paper during SCs.

Second, ACOFS uses a conventional probabilistic transition rule [9][14] in selecting features during SCs. Since, initially all the features in a given dataset possess equal pheromone value and heuristic value, such transition rule shows the random behavior. For this, ACOFS imposes an extra constraint upon the ants during feature SC. It is reasonable because of three aims which are: (a) all original features may be included, (b) entire features may get chance to be evaluated themselves, and (c) ACOFS can be free from the premature convergence. According to our sense, such amendment may lead to achieve a better result for the subsequent iterations. No other approaches [9][13][14] consider such technique in case of random manner.

Third, there are two new modified sets of rules are proposed in ACOFS which are pheromone updating and heuristic information estimating, respectively. Because of appearing the random behavior in selecting features during SCs initially, the above rules are proposed which afford some advantages, such as: (a) providing the correct update information to the features, (b) providing the effective balance between exploitation and exploration of ants. Mostly, the proposed rules are dependent on the outcomes of trained NN. The existing FS approaches do not consider such modifications in designing those rules (e.g. [9][10][13][14]).

The remainder of this paper is organized as follows. The details of ACOFS are discussed in Section 2. Experimental results, comparison to other methods, and the brief computation of computational complexity are reported in Section 3. Short conclusions with few remarks are given in Section 4.

2 Proposed ACOFS

In this proposed approach, we use NN training classifier for evaluating the constructed subsets. Involving any heuristic strategy in the classifier part also brings great impact in generating the salient subsets. It is noted that, most of the ACO based FS algorithms have less attention for the betterment of the activity of classifiers. However, a strategy of determining the required number of hidden neuron in the hidden layer of NN, called constructive strategy is induced in ACOFS. Details of this strategy can be seen in [16].

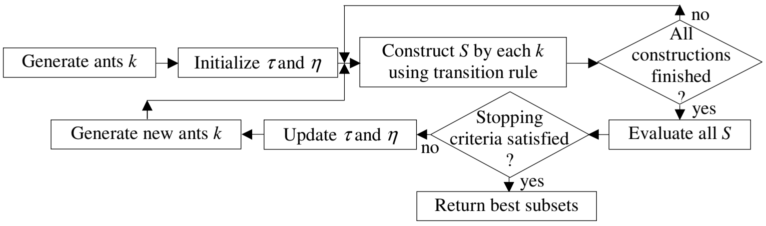
Generate ants k
Initialize τ and η
Construct S by each k using transition rule
All constructions finished ?
no
yes
Evaluate all S
Stopping criteria satisfied ?
no
yes
Update τ and η
Generate new ants k
Return best subsets

Fig. 1. Flow chart of ACOFS

The steps of ACOFS can be described by the flowchart shown in Fig. 1, which are explained further as follows.

Step 1) Initially generate a set of artificial ants k and the number of k must be equal to the number of original features n, i.e., $k \equiv n$.

Step 2) Initialize the level of pheromone trails τ and the heuristic information η of all features n by setting the equal value.

Step 3) Follow the scheme mentioned in Section 2.1 to be decided the subset size r in feature SC initially. After that, follow the conventional probabilistic transition rule [9][14] to select the features that is mentioned as follows,

$$P_i^k(t) = \begin{cases} \dfrac{[\tau_i(t)]^\alpha * [\eta_i(t)]^\beta}{\sum\limits_{u \in j^k} [\tau_u(t)]^\alpha * [\eta_u(t)]^\beta} & \text{if } i \in j^k \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where, j^k is the set of feasible features that can be added to the partial solution. τ_i and η_i are the pheromone value and heuristic value associated with feature i. α and β are the parameter of their relative importance. It should be noted that as initially all features contain equal value of τ and η, the Eq. (1) shows the random behavior. Thus to get the fair selection of features we impose one constraint upon the ants k during SCs at t iteration as,

$$S^k(t) \neq S^{k+1}(t) \qquad k = 1,2,3,........., n \tag{2}$$

Step 4) Check the construction progresses of S to determine whether it is finished or not. If finished by all ants k then continue, otherwise, go to Step 3).

Step 5) Evaluate the subsets $S^k(t)$ using NN during training. The classification accuracy (CA) of those subsets can be calculated as,

$$\gamma(S^k(t)) = 100 * (1 - TER) \tag{3}$$

Here, *TER* refers to the error rate of the NN in testing set.

Step 6) Check whether the algorithm has executed a certain number of iterations T or not. If executed then the FS process is stopped and the best salient feature subset can be obtained according to Section 2.2, otherwise, continue.

Step 7) Update τ and η of each feature i by following Section 2.3 and 2.4, respectively.

Step 8) Generate a new set of artificial ants k and go to Step 3) to be iterated the above mentioned similar procedures.

It is now clear that the basic steps behind ACOFS is nearly similar and straightforward compared to the existing algorithms (e.g., [9-15]) but differ in terms of some modifications integrated here. The following sections give further more details about the different components of ACOFS.

2.1 Determination of Subset Size

Unlike other algorithms, ACOFS uses a new modified determination scheme for subset size r from [6], called *restricted random scheme*. The aim is to provide the value of r for the individual ant k in a reasonable boundary. For that reason, the proposed scheme works under a restricted area which is described as follows.

Firstly, the probabilistic value of r in a bounded region can be defined as,

$$P(r) = \frac{n-r}{\sum_{i=1}^{l}(n-i)} \qquad where,\ 2 \le r \le \delta\ and\ l \in n-r \tag{4}$$

Here, δ is define by μ of n. Specifically, μ is a user specified parameter and its value is set up to [0.1, 0.6] depending on the number of n of different given datasets.

Secondly, ACOFS arranges all the possible values of P(r) to the conventional "roulette-wheel selection" scheme [17] to achieve the size of the subsets consistently.

Finally, deciding the suitable value of μ in Eq. (4) is a difficult issue. In ACOFS, the salient subset is being generated by a predefined range of its size which is in between 2 to 12. In most of the cases, whether the dimensionality of a given dataset is medium or high, ACOFS tries to generate subsets within this range. Thus, on basis of such assumption the value of μ is decided.

2.2 Determination of Salient Feature Subset

In this paper, unlike others, the determination of the salient feature subset is suggested basically by two ways: i) determining *local best* subset S^l, ii) determining *global best* subset S^g. Particularly, in every iteration t where $t \in 1,2,3,\ldots..,T$, ACOFS first computes $S^l(t)$ according to $Max(\gamma(S^k(t)))$. Then, $\gamma(S^l(t))$ is checked whether it is greater than the current $\gamma(S^g)$ where initially $\gamma(S^g) \equiv 0$. If so then $S^l(t)$ replaces S^g and associated all related components are replaced as the global components. The similar process is also happened while $\gamma(S^l(t))$ is equal to the current $\gamma(S^g)$ but

$|S^l(t)| < |S^g|$. Furthermore, if $\gamma(S^l(t)) \equiv \gamma(S^g)$ and $|S^l(t)| \equiv |S^g|$ then it requires one additional computation, i.e., information gain (λ) of the features associated with the both subsets. The subset which having higher computed value of λ regarded as S^g for the current iteration *t*. This is because, according to the statistical analysis, best feature is the one which has highest λ [18]. Thus, to find out the salient feature subset globally the above-mentioned process is repeated until *t* is equal to *T*.

2.3 Pheromone Update Rule

Pheromone updating is a vital part for working the ACO algorithm suitably. In course of SCs, there are two kinds of phenomena usually happened which are: random and probabilistic. On basis of these phenomena, a set of two modified pheromone update rules are introduced from the original one mentioned in [9] [19] as follows,

i) *Random case*: A modified rule is presented in Eq. (5). Here the second part (local updating) is responsible for the modification from the original formula where m_i is used to divide. The aim of such modification is to provide an equal level of pheromone to the feature *i* where $i \in n$.

$$\tau_i(t+1) = (1-\rho) * \tau_i(t) + \frac{1}{m_i} \sum_{k=1}^{n} \Delta\tau_i^k(t) + e * \Delta\tau_i^g(t) \tag{5}$$

$$where,\ \Delta\tau_i^k(t) = \begin{cases} \gamma(S^k(t)) & if\ i \in S^k(t) \\ 0 & otherwise \end{cases},\quad \Delta\tau_i^g(t) = \begin{cases} \gamma(S^l(t)) & if\ i \in S^l(t) \\ 0 & otherwise \end{cases} \tag{6}$$

Here, m_i, ρ and *e* refer to the number of count for *i* that is associated with $S^k(t)$ in the current iteration, pheromone decay and elitist parameter, respectively. Furthermore, $\Delta\tau^k$ and $\Delta\tau^g$ are the amount of local updating and global updating, respectively.

ii) *Probabilistic case*: The modified rule is shown in Eq. (7) where the modification is in the third part (global updating) of the original form.

$$\tau_i(t+1) = (1-\rho) * \tau_i(t) + \sum_{k=1}^{n} \Delta\tau_i^k(t) + e * \Delta\tau_i^g(t) \tag{7}$$

Here, $\Delta\tau_i^k(t)$ and $\Delta\tau_i^g(t)$ can be defined by the similar way as mentioned in Eq. (6). In addition, one additional caution is incorporated in the above updating rule at the time of adding $\Delta\tau_i^g(t)$ to the particular feature *i*. It means that, if $\gamma(S^l(t)) < \max(\gamma(S^l(t')), \varepsilon)$ where *t* is the current iteration which can be $\in 2,3,\ldots,T$. ε refers to the number of *CAs* of those associated subsets that maintain $|S^l(t')| \equiv |S^l(t)|$ and $t' \in (t\text{-}1),(t\text{-}2),\ldots,1$. Thus, a number of feature i' is ignored to get $\Delta\tau^g$. Here, $i' \in S^l(t')$ but $\notin S^{l*}$ where S^{l*} provides $\max(\gamma(S^l(t'-\tau)), \varepsilon)$.

It is worth mentioning that the aim of such above restriction is to provide the global updating to those features only that are really significant. It is due to the fact that, global updating has a vital role in selecting the salient features in ACOFS.

2.4 Estimation of Heuristic Information

Generally, the representation of heuristic value is the attractiveness of features and depends on the dependency degree. Indeed, it is necessary to limit the value of η, otherwise, the algorithm may be too greedy and the better solution may not come [14].

In this paper, we propose a set of new heuristic information estimation rule depending on two basic phenomena in SC. Both rules are mainly developed from the results of subset evaluation which is indeed more effective than statistical investigation of the feature set.

i) *Random case*: The estimation of η for the feature i can be performed according to the Eq. (8).

$$\eta_i = \frac{1}{m_i} \sum_{k=1}^{n} \gamma_a(S^k(t)) * (1 + \phi * e^{-\frac{|S^k(t)|}{n}}) \qquad \textit{if } i \in S^k(t) \tag{8}$$

ii) *Probabilistic case*: η is estimated according to Eq. (9).

$$\eta_i = m_i * \varphi_i * \omega * \lambda_i(p) * \sum_{k=1}^{n} \gamma_a(S^k(t)) * (1 + \phi * e^{-\frac{|S^k(t)|}{n}}) \qquad \textit{if } i \in S^k(t) \tag{9}$$

Here, φ_i refers to the number of count for the particular selected feature i that is a part of $S^k(t'')$. Here, t'' implies the whole completed iterations. However, the aim of multiplying m_i and φ_i in Eq. (9) is to combine the importance of i in terms of locally and globally. Conversely, the computation of information gain λ [18] of features is incorporated here because of making diversity among the different features in terms of η as well as removing the greedy manner of the specific features in feature SC. Finally, one additional exponential term is multiplied in Eq. (8) and (9). The aim of such incorporation is to provide the higher value of η to those features that are associated with the reduced subsets and provided better performances.

3 Experimental Setup

In order to evaluate the performance of ACOFS, four real-world benchmark datasets [20] were used. The used datasets are: breast cancer (BCR), glass (GLS), thyroid (THY), and sonar (SNR). The characteristics of these datasets and their partitions are shown in Table 1. Each experiment was performed 20 times and the presented results are the average of these 20 runs. The performance of ACOFS was evaluated in terms of the number of selected features (n_s) as well as classification accuracy (*CA*). All experiments were done in Pentium-core 2 duo, 2.66 GHz personal computer.

In this study, in order to achieve better results, we set T=20, α=2, β=3, τ=0.5, η=0.1, ρ=0.4, ω=0.1, and ϕ=0.1. There were also some other parameters used in NN training while the constructed subsets were evaluated. The initial connection weights for an NN were randomly set to [-1.0, 1.0]. The learning rate and the momentum term were set to [0.1, 0.2] and [0.5, 0.9], respectively. The number of partial training epochs was chosen between [10, 70]. The training was conducted by the well-known

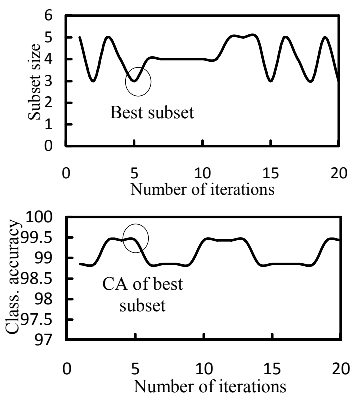

Fig. 2. Finding best subset of cancer dataset for a single run

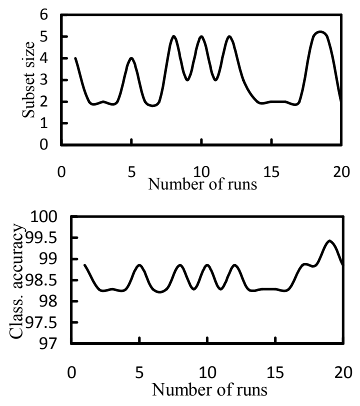

Fig. 3. Variation of best solutions for cancer dataset in different runs

BP algorithm [21]. We conducted one additional set of experiments to investigate the performance of the all original features using trained NN.

3.1 Experimental Results

The results presented in Table 2 were found by applying ACOFS on four real-world datasets. The *CA* in the table refers to the percentage of correct classification by the trained NN on the testing set. It is seen that ACOFS could select a small number of n_s from the original feature set. For example, for the thyroid dataset, ACOFS selected 2.85 features on average from a set of 21 features. This indicates that ACOFS could find salient features in a reduced way. The positive effect of a small number of features can also be seen when we look the *CA*. For example, the sonar dataset, the *CA* of all features was 75.49%, when it was 86.44% with 6.65 features. ACOFS exhibits good results for other datasets as well.

Furthermore, the use of n_s cause a small standard deviation (*SD*) as presented in the Table 2 for each entry. The low *SDs* imply robustness of ACOFS. Robustness is the consistency of an algorithm under different initial condition.

In order to observe how the subset solution of ACOFS progresses, Fig. 2 shows the scenery of cancer dataset in a single run how the best solution was identified. It is seen that the local best performances is varied with varying the local best subsets. There are also several points where the *CAs* are maximized but the best solution is selected (indicated by circle) in that particular run by considering the size of subset which is reduced much. On the other hand, Fig. 3 shows the variation between the subset size and *CAs* in total 20 runs. Thus, from the above discussion it can be assumed that the performance of a subset is roughly dependent on its size.

3.2 Comparison with Other Works

The obtained results of ACOFS on four benchmark datasets are compared with the results of different FS algorithms. Three well-known algorithms ACOFS$_S$[12], GPFS[6], and MLPFS[2] are chosen for comparison. The results are summarized in Table 3. Since, the different algorithms were evaluated in different experimental set-ups, for this, we cannot compare the results of ACOFS completely with other algorithms until all experiments are performed in the same setup.

Table 3 shows the comparisons between ACOFS and other algorithms on basis of average percentage of *CAs* and average number of n_s. Observing this Table, the comparative analysis for four datasets and other algorithms are stated below.

Cancer: The overall performance of ACOFS in terms of n_s and *CA* is much better than all other algorithms except for one case. In this case, the number of n_s in ACOFS is comparable to that of GPFS.

Glass: ACOFS has shown a remarkable performance in all cases comparing to MLPFS.

Thyroid: In terms of number of n_s, ACOFS achieved a few number of features comparing to ACOFS_S while in case of *CA*, it also achieved better results.

Sonar: In terms of number of n_s, ACOFS achieved a few number of salient features comparing to GPFS and MLPFS. In contrast, ACOFS performed better in terms of *CA* comparing to the above two algorithms.

3.3 Computational Complexity

The study of computational complexity helps to understand the actual computational cost (CC) of an algorithm. For this, we are inspired to compute the CC of ACOFS briefly according to the form of big-O notation. In ACOFS, there are five vital steps: (i) subset construction, (ii) subset evaluation, (iii) stopping criterion, (iv) salient subset determination, and (v) updating pheromone and heuristic value.

i) *Subset Construction*: In this work, we use two kinds of computation for the SC because of two different kinds of phenomena exhibited by Eq. (1). For the random case, the total computational cost takes $O(k \times r)+O(k \times r^2)$ operations for k ants. In terms of probabilistic case, ACOFS uses the Eq. (1) for selecting features in SC which shows the total $O(k \times r \times q)$ operations. Here, q equals to $(T\text{-}1)$. In practice, the cost of generating r is negligible. In contrast, the operations $O(k \times r)+O(k \times r^2)$ are performed only for once which can also be ignored.

Table 1. Characteristics of datasets

Datasets	Features	Classes	Examples	Partition sets		
				Training	Validation	Testing
BCR	9	2	699	349	175	175
GLS	9	6	214	108	53	53
THY	21	3	7200	3600	1800	1800
SNR	60	2	208	104	52	52

Table 2. Average results of BCR, GLS, THY, and SNR datasets. SD refers standard deviation.

Datasets	Avg. results with all features		Avg. results with selected features			
	CA (%)	SD	No. of features	SD	CA (%)	SD
BCR	97.60	0.003	3.10	1.26	98.57	0.33
GLS	68.59	0.065	3.10	1.17	80.75	1.27
THY	97.99	0.002	2.85	1.15	98.36	0.13
SNR	75.49	0.082	6.65	1.93	86.44	0.73

Table 3. Comparisons between ACOFS, ACOFS$_S$[12], GPFS[6], and MLPFS[2]

Datasets		Comparisons			
		ACOFS	ACOFS$_S$	GPFS	MLPFS
BCR	No. of features	3.10	12.00	2.23	8.00
	Class. acc. (%)	98.57	95.57	96.84	89.40
GLS	No. of features	3.10	--	--	8.00
	Class. acc. (%)	80.75	--	--	44.10
THY	No. of features	2.85	14.00	--	--
	Class. acc. (%)	98.36	94.50	--	--
SNR	No. of features	6.65	--	9.45	29.00
	Class. acc. (%)	86.44	--	86.26	59.10

ii) Subset Evaluation: In ACOFS, we evaluate all subsets S using constructive NN in which four kinds of operations are necessarily required for each case, such as, (a) partial training, (b) termination criterion, (c) further training, (d) contribution computation, and (e) adding a hidden neuron. If we analyze and summarize then it can be found that the computation takes $O(k \times \tau_e \times M \times T \times P_t \times W)$ operations for evaluating all S in T iterations. Here, k is the number of constructed S in one iteration. τ_e and M refer to the number of epochs in one partial training and total number of partial training, whereas P_t and W denote the number of examples in the training set, and the number of weights in the current NN, respectively.

For the remaining steps it can be found that there are some constant and negligible computations are taken place. Thus, finally we get the total computational cost by neglecting and summarizing: $O(k \times r \times q)$+$O(k \times \tau_e \times M \times T \times P_t \times W)$. Since, $O(k \times r \times q) << O(k \times \tau_e \times M \times T \times P_t \times W)$, hence the total computational cost of ACOFS is $O(k \times \tau_e \times M \times T \times P_t \times W)$, which is almost similar to the existing FS algorithm [12]. Thus, it is clear that the incorporation of several techniques in ACOFS does not increase its computational cost.

4 Conclusion

In this paper, an efficient FS algorithm ACOFS is presented that is based on the ACO algorithm. The proposed ACOFS utilizes the strategy of real ants for finding shortest path from nest to food sources in FS task. A new modification between the pheromone update rules and the estimation of heuristic information was done in this proposed algorithm in order to enhance the performance of ACO algorithm for FS.

In ACOFS, neither the incorporation of forward selection strategy nor fixed size subset strategy is taken into account for its feature SC process. Apart from these both issues, ACOFS uses a probabilistic scheme for determining the subset size adaptively. Such incorporation ultimately guides to find the best solutions in terms of lower number of salient features and of better classification performances.

Extensive experiments have been carried out in this paper to evaluate how well ACOFS performed on different real-world datasets in comparison with three prominent FS algorithms. In almost all except only for once, ACOFS outperformed the others in terms of the number of selected features and classification performances. The results of the low standard deviations of the classification accuracies exhibit the robustness of this algorithm. Furthermore, the estimated computational complexity of this algorithm reflected that incorporation of several techniques does not increase the computational cost during FS in comparison of other ACO based FS algorithms.

Since the focus of this paper is to present the fundamental ideas and technical details of ACOFS, the detailed comparisons with other algorithms using rigorous statistical methods are left as the future work.

Acknowledgements

Supported by grants to KM from the Japanese Society for Promotion of Sciences, the Yazaki Memorial Foundation for Science and Technology, and the University of Fukui.

References

1. Hall, M.A.: Correlation-based feature selection for discrete and numeric class machine learning. In: 17th International Conference on Machine Learning (2000)
2. Gasca, E., Sanchez, J.S., Alonso, R.: Eliminating redundancy and irrelevance using a new MLP-based feature selection method. Pattern Recognition 39, 313–315 (2006)
3. Guan, S., Liu, J., Qi, Y.: An incremental approach to contribution-based feature selection. Journal of Intelligence Systems 13(1) (2004)
4. Abe, S.: Modified backward feature selection by cross validation. In: Proceedings of the European Symposium on Artificial Neural Networks, pp. 163–168 (2005)
5. Yang, J.H., Honavar, V.: Feature subset selection using a genetic algorithm. IEEE intelligent systems 13(2), 44–49 (1998)
6. Muni, D.P., Pal, N.R., Das, J.: Genetic Programming for Simultaneous Feature Selection and Classifier Design. IEEE Trans. on Syst., Man, and Cybern.-Part B: Cybern. 36(1) (2006)
7. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: Optimization by a colony of cooperative agents. IEEE Trans. on Systems, Man, and Cybern. 26(1), 29–41 (1996)
8. Maniezzo, V., Colorni, A.: The ant system applied to the quadratic assignment problem. IEEE transaction on knowledge and data engineering 11(5), 769–778 (1999)
9. Aghdam, M.H., Aghaee, N.G., Basiri, M.E.: Text feature selection using ant colony optimization. Expert systems with applicantions 36, 6843–6853 (2009)

10. Kanan, H.R., Faez, K., Taheri, S.M.: Feature selection using ACO: a new method and comparative study in the application of face recognition system. In: Intl. conf. on data mining, pp. 63–76 (2007)
11. Khushaba, R.N., Alsukker, A., Ani, A.A., Jumaily, A.A.: Enhanced feature selection algorithm using ant colony optimization and fuzzy memberships. In: Intl. conf. on biomedical engineeri., pp. 34–39 (2008)
12. Sivagaminathan, R.K., Ramakrishnan, S.: A hybrid approach for feature subset selection using neural networks and ant colony optimization. Expert systems with applications 33, 49–60 (2007)
13. Ani, A.: Feature subset selection using ant colony optimization. International journal of computational intelligence 2, 53–58 (2005)
14. Ke, L., Feng, Z., Ren, Z.: An efficient ant colony optimization approach to attribute reduction in rough set theory. Pattern Recognition Letters 29, 1351–1357 (2008)
15. Robbins, K.R., Zhang, W., Bertrand, J.K.: The ant colony algorithm for feature selection in high-dimension gene expression data for disease classification. J. of Math. Medi. and Biol., 1–14 (2008)
16. Kwok, T.Y., Yeung, D.Y.: Objective functions for training new hidden units in constructive neural networks. IEEE Trans. Neural Network 8(5), 1131–1148 (1997)
17. Goldberg, D.E.: Genetic Algorithms in search, optimization and machine learning (2004)
18. Mitchell, T.M.: Machine learning. McGraw-Hill, New York (1997)
19. Dorigo, M., Stutzle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
20. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases, Dept. of Information and Computer Sciences, University of California, Irvine (1998)
21. Rumelhart, D.E., McClelland, J.: Parallel distributed processing. MIT Press, Cambridge (1986)