A Novel Hybrid ACO-GA Algorithm for Text Feature Selection
Mohammad Ehsan Basiri, Shahla Nemati

Abstract—In our previous work we have proposed an ant colony optimization (ACO) algorithm for feature selection. In this paper, we hybridize the algorithm with a genetic algorithm (GA) to obtain excellent features of two algorithms by synthesizing them. Proposed algorithm is applied to a challenging feature selection problem. This is a data mining problem involving the categorization of text documents. We report the extensive comparison between our proposed algorithm and three existing algorithms – ACO-based, information gain (IG) and CHI algorithms proposed in the literature. Proposed algorithm is easily implemented and because of use of a simple classifier in that, its computational complexity is very low. Experiments are carried out on Reuters-21578 dataset. Simulation results on Reuters-21578 dataset show the superiority of the proposed algorithm.

I. INTRODUCTION

FEATURE selection (FS) is a commonly used step in machine learning, especially when dealing with a high dimensional space of features. The main objective of FS is to choose a subset of features from the original set of features forming patterns in a given dataset [1]. Feature selection is extensive and it spreads throughout many fields, including text categorization [2], data mining [3], machine learning [4], pattern recognition, and signal processing [5].

Recently, text categorization has become a key technology to deal with and organize a large number of documents. A major problem of text categorization is the high dimensionality of the feature space. Most of these dimensions are not relative to text categorization; even some noise data hurt the performance of the classifier. Hence, we need to select some representative features from the original feature space to reduce the dimensionality of feature space and improve the efficiency and performance of classifier [6].

Among too many methods, which are proposed for FS, population-based optimization algorithms such as genetic algorithm (GA)-based method and ant colony optimization (ACO)-based method have attracted a lot of attention [2]. These methods attempt to achieve better solutions by application of knowledge from previous iterations. In this paper, we propose a novel ant colony-optimization/genetic algorithm (ACO-GA) hybrid method that selects the most representative features for text categorization.

ACO is a branch of newly developed form of artificial intelligence called swarm intelligence (SI). Swarm intelligence is a field, which studies “the emergent collective intelligence of groups of simple agents” [7]. In groups of insects, which live in colonies, such as ants and bees, an individual can only do simple task on its own, while the colony’s cooperative work is the main reason determining the intelligent behavior it shows [8]. ACO algorithm is inspired by social behavior of ant colonies and was represented in the early 1990’s by M. Dorigo and colleagues [9].

Genetic algorithms (GAs) were introduced by Holland as a computational analogy of natural adaptive systems. Genetic algorithms are optimization techniques based on the mechanism of natural selection. They used operations found in natural genetics to guide itself through the paths in the search space [10]. Because of their advantages, recently, GAs have been widely used as a tool for feature selection in data mining [11].

In this paper, a new hybrid ACO-GA text feature selection algorithm has been introduced. The classifier performance and the length of selected feature subset are adopted as heuristic information for algorithm. Thus, proposed algorithm needs no priori knowledge of features. Proposed algorithm is applied to text features of bag of words model in which a document is considered as a set of words or phrases (called terms) and each position in the input feature vector corresponds to a given term in original document [12]. Finally, the classifier performance and the length of selected feature vector are considered for performance evaluation.

The rest of this paper is organized as follows. Section II presents a brief overview of feature selection methods. Ant colony optimization is described in Section III. Section IV discusses ACO algorithm for feature selection. Section V explains the proposed feature selection algorithm. Section VI reports computational experiments. It also includes a brief discussion of the results obtained and finally the conclusion is offered in the last section.

II. FEATURE SELECTION APPROACHES

Feature selection is included in discrete optimization problems. The whole search space for optimization contains all possible subsets of features, meaning that its size is

$$\sum_{k=0}^{n} \binom{n}{k} = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n} = 2^n. \quad (1)$$
Where \( n \) is the dimensionality (the number of features) and \( s \) is the size of the current feature subset [10]. Usually FS algorithms involve heuristic or random search strategies in an attempt to avoid this prohibitive complexity. However, the degree of optimality of the final feature subset is often reduced [13].

Feature selection algorithms can be classified into two categories based on their evaluation procedure [14]. If an algorithm performs FS independent of any learning algorithm (i.e., it is a completely separate preprocessor), then it is included in filter approach category. This approach is mostly includes selecting features based on inter-class separability criterion [14]. If the evaluation procedure is tied to the task (e.g., classification) of the learning algorithm, the FS algorithm is a sort of wrapper approach. This method searches through the feature subset space using the estimated accuracy from an induction algorithm as a measure of subset suitability. Although wrappers may produce better results, they are expensive to run and can break down with very large numbers of features. This is due to the use of learning algorithms in the evaluation of subsets, some of which can encounter problems while dealing with large datasets [13], [15].

In the wrapper approach the evaluation function calculates the suitability of a feature subset produced by the generation procedure and it also compares that with the previous best candidate, replacing it if found to be better. A stopping criterion is tested in each iteration to determine whether the FS process should continue.

The two mentioned approaches are also classified into five main methods [16]:

- **Forward selection**: begins with an empty set and features are added to greedily one at a time.
- **Backward elimination**: begins with a feature set containing all features and features are removed from greedily one at a time.
- **Forward stepwise selection**: begins with an empty set and features are either added to or removed from greedily one at a time.
- **Backward stepwise elimination**: begins with a feature set containing all features and features are either added to or removed from greedily one at a time.

Other famous FS approaches are based on the genetic algorithm (GA) [17], simulated annealing (SA), particle swarm optimization (PSO) and ant colony optimization (ACO) [8], [13], [18]–[20].

Reference [18] proposes a hybrid approach for fitness classification problem. This method has used combination of mutual information and ACO. The hybrid of ACO and mutual information has been used for feature selection in the forecaster [19]. Furthermore, ACO is used for finding rough set reducts [13] and a new Ant-Miner that used a different pheromone updating strategy has been introduced in [8]. In addition, an ACO-based method has been used in the application of speaker verification systems [5] and some surveys of feature selection algorithms are given in [11], [21], and [22].

### III. Genetic Algorithm (GA)

Genetic algorithms belong to a class of population-based stochastic search algorithms that are inspired from principles of natural evolution known as evolutionary algorithms (EA) [10]. GA is based on the principle of survival of fittest, as in the natural phenomena of genetic inheritance and Darwinian strife for survival. These algorithms are general-purpose optimization algorithms with a probabilistic component that provide a means to search poorly understood, irregular spaces.

Primarily, GA was designed to optimally solve sequential decision processes more than to perform function optimization but over the years, it has been used widely in both learning and optimization problems [23].

Instead of working with a single point, GAs work with a population of points. Each point is a vector in hyperspace representing one potential (or candidate) solution to the optimization problem. A population is, thus, just an ensemble or set of hyperspace vectors. Each vector is called a chromosome in the population. The number of elements in each chromosome depends on the number of parameters in the optimization problem and the way to represent the problem [10].

The operators of the GA are selection, crossover, and mutation. Selection is a process in which \( p \) individuals are selected with a probability proportional to their fitness to be parents. To many evolutionary computation practitioners, crossover is what distinguishes a GA from other evolutionary computation paradigms. Crossover is the process of exchanging portions of the strings of two “parent” chromosomes. An overall probability is assigned to the crossover process, which is the probability, that given two parents, the crossover process will occur. This probability is often in the range of 0.65–0.80. The final operation in the typical GA procedure is mutation. Mutation consists of changing an element’s value at random, often with a constant probability for each element in the population. The probability of mutation can vary widely according to the application and the preference of the person exercising the GA. However, values of between 0.001 and 0.01 are not unusual for mutation probability.

### IV. Ant Colony Optimization (ACO)

In the early 1990s, ant colony optimization (ACO) was introduced by M. Dorigo and colleagues as a novel nature-inspired meta-heuristic for the solution of hard combinatorial optimization (CO) problems. ACO belongs to the class of meta-heuristics, which includes approximate algorithms used to obtain good enough solutions to hard CO problems in a reasonable amount of computation time. The inspiring source of ACO is the foraging behavior of real ants [24].

The first ACO algorithm developed was the ant system (AS) [25], [26], and since then several improvement of the...
The ACO algorithm is based on a computational paradigm inspired by real ant colonies and the way they function. The underlying idea was to use several constructive computational agents (simulating real ants). A dynamic memory structure incorporating information on the effectiveness of previous choices based on the obtained results, guides the construction process of each agent. The behavior of each single agent is therefore inspired by the behavior of real ants [24].

The paradigm is based on the observation made by ethologists about the medium used by ants to communicate information regarding shortest paths to food by means of pheromone trails. A moving ant lays some pheromone on the ground, thus making a path by a trail of this substance. While an isolated ant moves practically at random, exploration, an ant encountering a previously laid trail can detect it and decide with high probability to follow it, exploitation, and consequently reinforces the trail with its own pheromone. What emerges is a form of autocatalytic process through which the more the ants follow a trail, the more attractive that trail becomes to be followed. The process is thus characterized by a positive feedback loop, during which the probability of choosing a path increases with the number of ants that previously chose the same path. The mechanism above is the inspiration for the algorithms of the ACO family [24].

ACO algorithms can be applied to optimization problems, for which the following problem-dependent aspects can be defined [1], [2]:

- An appropriate graph representation to represent the discrete search space. The graph should accurately represent all states and transitions between states. A solution representation scheme also has to be defined.
- Heuristic desirability of links the representation graph.
- An autocatalytic (positive) feedback process; that is, a mechanism to update pheromone concentrations such that current successes positively influence feature solution construction.
- A constraint-satisfaction method to ensure that only feasible solutions are constructed.
- A solution construction method which defines the way in which solutions are built and a state transition probability.

V. ANT COLONY OPTIMIZATION FOR FEATURE SELECTION

Feature selection is one of the applications of subset problems (SSP). Given a feature set of size n, the FS problem is to find a minimal feature subset of size s (s < n) while retaining a suitably high accuracy in representing the original features. Therefore, there is no concept of path. A partial solution does not define any ordering among the components of the solution, and the next component to be selected is not necessarily influenced by the last component added to the partial solution [30], [31]. Furthermore, solutions to an FS problem are not necessarily of the same size. To apply an ACO algorithm to solve a feature selection problem, these aspects need to be addressed. The first problem is addressed by redefining the way that the representation graph is used.

A. Graph Representation

The feature selection problem may be reformulated into an ACO-suitable problem. ACO requires a problem to be represented as a graph. Here nodes represent features, with the edges between them denoting the choice of the next feature. The search for the optimal feature subset is then an ant traversal through the graph where a minimum number of nodes are visited that satisfies the traversal stopping criterion. Figure 1 illustrates this setup. Nodes are fully connected to allow any feature to be selected next. The ant is currently at node \( f_1 \) and has a choice of which feature to add next to its path (dotted lines). It chooses feature \( f_2 \) next based on the transition rule, then \( f_3 \) and then \( f_4 \). Upon arrival at \( f_4 \), the current subset \( \{f_1, f_2, f_3, f_4\} \) is determined to satisfy the traversal-stopping criterion (e.g. suitably high classification accuracy has been achieved with this subset). The ant terminates its traversal and outputs this feature subset as a candidate for data reduction [1].

Based on this reformulation of the graph representation, the transition rules and pheromone update rules of standard ACO algorithms can be applied. In this case, pheromone and heuristic value are not associated with links. Instead, each feature has its own pheromone value and heuristic value.

Fig. 1. ACO problem representation for FS.

B. Heuristic Desirability

The basic ingredient of any ACO algorithm is a constructive heuristic for probabilistically constructing solutions [24]. A constructive heuristic assembles solutions as sequences of elements from the finite set of solution components. A solution construction starts with an empty partial solution. Then, at each construction step, the current partial solution is extended by adding a feasible solution component from the set of solution components. A suitable heuristic desirability of traversing between features could be any subset evaluation function for example, an entropy-based measure or rough set dependency measure [13]. In proposed algorithm classifier, performance is mentioned as heuristic desirability for feature selection. The heuristic desirability of traversal and node pheromone levels are combined to form the so-called probabilistic transition rule, denoting the probability that ant \( k \) will include feature \( i \) in its
solution at time step $t$:  
\[ p_i^k(t) = \left\{ \begin{array}{ll} 
\left[ \tau_i(t) \right]^\alpha \left[ \eta_i \right]^\beta & \text{if } i \in J^k \\
\sum_{u \in J^k} \left[ \tau_u(t) \right]^\alpha \left[ \eta_u \right]^\beta & \text{otherwise}
\end{array} \right. \]  
(2)

Where $J^k$ is the set of feasible features that can be added to the partial solution; $\tau_i$ and $\eta_i$ are respectively the pheromone value and heuristic desirability associated with feature $i$. $\alpha$ and $\beta$ are two parameters that determine the relative importance of the pheromone value and heuristic information.

The transition probability used by ACO is a balance between pheromone intensity (i.e. history of previous successful moves), $\tau_i$, and heuristic information (expressing desirability of the move), $\eta_i$. This effectively balances the exploitation–exploration trade-off. The search process favors actions that it has found in the past and which proved to be effective, thereby exploiting knowledge obtained about the search space. On the other hand, in order to discover such actions, the search has to investigate previously unseen actions, thereby exploring the search space. The best balance between exploitation and exploration is achieved through proper selection of the parameters $\alpha$ and $\beta$. If $\alpha = 0$, no pheromone information is used, i.e. previous search experience is neglected. The search then degrades to a stochastic greedy search. If $\beta = 0$, the attractiveness (or potential benefit) of moves is neglected.

C. Pheromone Update Rule

After all ants have completed their solutions, pheromone evaporation on all nodes is triggered, and then according to equation (3) each ant $k$ deposits a quantity of pheromone, $\Delta \tau_i^k(t)$, on each node that it has used

\[ \Delta \tau_i^k(t) = \left\{ \begin{array}{ll} 
\phi \gamma(S^k(t)) + \frac{\phi(n - |S^k(t)|)}{n} & \text{if } i \in S^k(t) \\
0 & \text{otherwise}
\end{array} \right. \]  
(3)

Where $S^k(t)$ is the feature subset found by ant $k$ at iteration $t$, and $|S^k(t)|$ is its length. The pheromone is updated according to both the measure of the classifier performance, $\gamma(S^k(t))$, and feature subset length. $\phi$ and $\varphi$ are two parameters that control the relative weight of classifier performance and feature subset length, $\phi \in [0,1]$ and $\varphi = 1 - \phi$.

This formula means that the classifier performance and feature subset length have different significance for feature selection task. In our experiment we assume that classifier performance is more important than subset length, so they were set as $\phi = 0.8$, $\varphi = 0.2$.

In practice, the addition of new pheromone by ants and pheromone evaporation are implemented by the following rule applied to all the nodes:

\[ \tau_i(t + 1) = (1 - \rho)\tau_i(t) + \sum_{k=1}^{m} \Delta \tau_i^k(t) + \Delta \tau_i^g(t). \]  
(4)

Where $m$ is the number of ants at each iteration and $\rho \in (0,1]$ is the pheromone trail decay coefficient. The main role of pheromone evaporation is to avoid stagnation, that is, the situation in which all ants constructing the same solution. In our experiment, we consider $\rho = 0.2$. $g$ indicates the best ant at each iteration. All ants can update the pheromone according to equation (4) and the best ant deposits additional pheromone on nodes of the best solution. This leads to the exploration of ants around the optimal solution in next iterations.

D. Solution Construction

The process begins by generating a number of ants that are then placed randomly on the graph i.e. each ant starts with one random feature. Alternatively, the number of ants to place on the graph may be set equal to the number of features within the data; each ant starts path construction at a different feature. From these initial positions, they traverse nodes probabilistically until a traversal stopping criterion is satisfied. The resulting subsets are gathered and then evaluated. If an optimal subset has been found or the algorithm has executed a certain number of times, then the process halts and outputs the best feature subset encountered. If none of these conditions hold, then the pheromone is updated, a new set of ants are created and the process iterates once more.

VI. PROPOSED ACO-GA ALGORITHM

Typically, a text categorization system consists of several essential parts including feature extraction and feature selection. After preprocessing of text documents, feature extraction is used to transform the input text document into a feature set (feature vector). Feature selection is applied to the feature set to reduce the dimensionality of it. This process is shown in Fig. 2.

As mentioned earlier, in this paper we intend to hybridize ACO and GA in such a manner that they complement each other for feature selection in text categorization. A colony of ants is used to evolve a set of solutions using ACO (explained in Section V). Every ant represents a tradeoff between two disparate parameters, namely, exploration of new paths and exploitation of greedy algorithm. This population is ameliorated genetically by genetic algorithm to generate the next generation of solutions. This causes the contribution of even the bad ants to solution building which was absent in non-hybridized applications of GA and ACO.
ACO-GA is used to explore the space of all subsets of given feature set. The performance of selected feature subsets is measured by invoking an evaluation function with the corresponding reduced feature space and measuring the specified classification result. The best feature subset found is then output as the recommended set of features to be used in the actual design of the classification system.

The main steps of proposed feature selection algorithm are shown in Figure 3.

The time complexity of proposed algorithm is $O(Imn)$, where $I$ is the number of iterations, $m$ the number of ants, and $n$ the number of original features. This can be seen from Fig. 3. In the worst case, each ant selects all the features. As the heuristic is evaluated after each feature is added to the candidate subset, this will result in $n$ evaluations per ant. After the first iteration in this algorithm, $mn$ evaluations will have been performed. After $I$ iterations, the heuristic will be evaluated $Imn$ times.

VII. EXPERIMENTS AND RESULTS

A series of experiments was conducted to show the utility of proposed feature selection algorithm. All experiments have been run on a machine with 3.0 GHz CPU and 512 MB of RAM. We implement proposed ACO-GA algorithm and other two feature selection algorithms in Matlab R2006a. The operating system was Windows XP Professional. The following sections describe Reuters21578 dataset and implementation results.

A. Reuters Dataset

Some publicly available standard datasets can be used as test collections for text categorization (TC). The most widely used is the Reuters collection, consisting of stories from Reuter’s news agency that classified under categories related to economics [32]. The Reuters collection accounts for most of the experimental works in TC so far.

We used Reuters-21578, the newer version of the corpus. In Reuters-21578 dataset, we adopt the top ten classes; 5785 documents in training set and 2299 documents in test set. The distribution of the class is unbalance. The maximum class has 2721 documents, occupying 47.04% of training set. The minimum class has 72 documents, occupying 1.24% of training set. Table I shows the ten most frequent categories along with the number of training and test.

<table>
<thead>
<tr>
<th>Category Name</th>
<th>Number of Train</th>
<th>Number of Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acquisition</td>
<td>1484</td>
<td>664</td>
</tr>
<tr>
<td>Corn</td>
<td>170</td>
<td>53</td>
</tr>
<tr>
<td>Crude</td>
<td>288</td>
<td>126</td>
</tr>
<tr>
<td>Earn</td>
<td>2721</td>
<td>1052</td>
</tr>
<tr>
<td>Grain</td>
<td>72</td>
<td>32</td>
</tr>
<tr>
<td>Interest</td>
<td>165</td>
<td>74</td>
</tr>
<tr>
<td>Money-fx</td>
<td>313</td>
<td>106</td>
</tr>
<tr>
<td>Ship</td>
<td>122</td>
<td>42</td>
</tr>
<tr>
<td>Trade</td>
<td>297</td>
<td>99</td>
</tr>
<tr>
<td>Wheat</td>
<td>153</td>
<td>51</td>
</tr>
</tbody>
</table>
examples in each.

B. Feature extraction

A classifier cannot directly interpret texts. Because of this, an indexing procedure that maps a text document into a compact representation of its content needs to be uniformly applied to documents. As showed in equation (5) a text $d_j$ is usually represented as a vector of term weights

$$d_j = \{w_{1,j}, w_{2,j}, \ldots, w_{|T|, j}\}$$

(5)

Where $T$ is the set of terms (features) that occur at least once in at least one document of training set ($Tr$), and $0 \leq w_{kj} \leq 1$ represents, how much term $t_k$ contributes to the semantics of document $d_j$.

Typically each position in the input feature vector corresponds to a given word or phrase. This representation often called bag of words model. Weights are determined using normalized $tfidf$ function [33], defined as

$$w_{kj} = \frac{tfidf(t_k, d_j)}{\sqrt{\sum_{s=1}^{|[T]|} (tfidf(t_s, d_j))^2}}$$

(6)

Where

$$tfidf(t_k, d_j) = \#(t_k, d_j) \log \frac{|Tr|}{\#_{Tr}(t_k)}.$$  

(7)

Where $\#(t_k, d_j)$ is the number of occurrence of $t_k$ in $d_j$, and $\#_{Tr}(t_k)$ denote the number of documents in $Tr$ in which $t_k$ occurs.

C. Performance Measure

Usually precision ($\pi$) and recall ($\rho$) are used for performance measurement. They showed in the following equations:

$$\pi_i = \frac{TP_i}{TP_i + FP_i}$$

(8)

$$\rho_i = \frac{TP_i}{TP_i + FN_i}$$

(9)

Where $TP_i$ is the number of test documents correctly classified under $i$th category ($c_i$), $FP_i$ is the number of test documents incorrectly classified under $c_i$, and $FN_i$ is the number of test documents incorrectly classified under other categories. These probabilities may be estimated in terms of the contingency table for $c_i$ on a given test set which is shown in Table II. Another commonly used measure in TC is $F1$ measure that is defined in equation (10)

$$F1 = \frac{2 \times \pi \times \rho}{(\pi + \rho)}$$

(10)

When dealing with multiple classes there are two possible ways of averaging above measures, namely, macro average and micro average. The macro average weights equally all the classes, regardless of how many documents belong to it. The micro average weights equally all the documents, thus favoring the performance on common classes. The global contingency table which is shown in Table III is thus obtained by summing over category-specific contingency tables; Eqs. (11) - (14) show micro averaging and macro averaging on precision and recall.

$$\pi^M = \frac{\sum_{i=1}^{[C]} TP_i}{\sum_{i=1}^{[C]} (TP_i + FN_i)}$$

(11)

$$\rho^M = \frac{\sum_{i=1}^{[C]} TP_i}{\sum_{i=1}^{[C]} (TP_i + FN_i)}$$

(12)

$$\pi^M = \frac{\sum_{i=1}^{[C]} \pi_i}{|C|}$$

(13)

$$\rho^M = \frac{\sum_{i=1}^{[C]} \rho_i}{|C|}$$

(14)

D. Results

To show the utility of proposed ACO-GA algorithm we compare proposed algorithm with two statistical approaches namely, information gain and CHI [33], and an ACO-based algorithm proposed in our previous work [2]. Various values were tested for the parameters of proposed algorithm. The results show that the highest performance is achieved by setting the parameters to values shown in Table IV. These values were empirically determined in our preliminary experiments; but we make no claim that these are optimal

<table>
<thead>
<tr>
<th>GA AND ACO PARAMETER SETTINGS</th>
<th>Population</th>
<th>Iteration</th>
<th>Crossover probability</th>
<th>Mutation probability</th>
<th>Initial pheromone</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACO</td>
<td>50</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Analyzing the precision and recall shown in Table V, we see that on average, the ACO and ACO-GA algorithms obtained a higher accuracy value than the information gain and CHI.

To graphically illustrate the progress of the ant colony as it searches for optimal solutions, we take percent features as the horizontal coordinate and the F1 measure as the vertical coordinate. This should illustrate the process of improvement of the best ant as the number of features increase. Figs. 4 and 5 show the micro-averaged and macro-averaged F1 measure for each of the feature selection algorithms as we change the number of selected features. The results show that as the percentage of selected features exceeds 14% in micro-F1 and macro-F1 measures, the ACO-GA algorithm outperforms ACO, IG, and CHI.

Table VI describes micro-F1 and macro-F1 for IG, CHI, ACO, and ACO-GA feature selection algorithms.

### TABLE VI
**MICRO-F1 AND MACRO-F1 OF FOUR ALGORITHMS**

<table>
<thead>
<tr>
<th>Feature Selection Algorithms</th>
<th>Macro-F1</th>
<th>Micro-F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Gain</td>
<td>68.0241</td>
<td>80.9482</td>
</tr>
<tr>
<td>CHI</td>
<td>68.2595</td>
<td>82.2097</td>
</tr>
<tr>
<td>ACO</td>
<td>78.4224</td>
<td>89.0822</td>
</tr>
<tr>
<td>ACO-GA</td>
<td>80.4691</td>
<td>92.1862</td>
</tr>
</tbody>
</table>

In this paper a novel ant colony optimization / genetic algorithm hybrid feature selection algorithm is presented. In the proposed algorithm, the classifier performance and the length of selected feature subset are adopted as heuristic information. Therefore, it can select the optimal feature subset without the prior knowledge of features. The proposed algorithm has the ability to converge quickly; it has a strong search capability in the problem space and can efficiently find a minimal feature subset. Experimental results demonstrate competitive performance.

Proposed algorithm, ACO-GA, was compared with an existing ACO-based feature selection and two statistical feature selection methods in text categorization. In order to evaluate the performance of proposed algorithm, experiments were carried out on the most widely used dataset in the literature, Reuters-21578 dataset.

The computational results indicate that proposed algorithm outperforms ACO, information gain and CHI methods since it achieved better performance with the lower number of features. For future work, the authors intend to investigate the performance of proposed feature selection algorithm by taking advantage of using classifiers that are more complex in that. Another research direction will...


