2009 International Conference of Soft Computing and Pattern Recognition

Keyword Combination Extraction in Text Categorization Based on Ant Colony Optimization

Zi-jun Yu, Wei-gang Wu, jing Xiao, Jun Zhang(Corresponding Author)
Dept. of Computer Science
SUN yat-sen Uninversity
Guangzhou, China
e-mail: junzhang@ieee.org

Rui-Zhang Huang and Ou Liu
Hong Kong Polytechnic University
Kowloon, Hong Kong

***Abstract*—Due to the increasing number of documents in digital form, the automated text categorization (TC) has become more and more promising in the last ten years. A TC system can automatically assign a document with the most suitable category, but the reason for such an assignment is usually unknown by users. To make the TC system be interpretable, it is necessary to select a group of keywords, or termed a keyword combination, to describe each text category. In this paper, we propose a novel algorithm, keyword combination extraction based on ant colony optimization (KCEACO), to search the optimal keyword combination of a target category. By extending the traditional feature selection techniques, an evaluation function is designed for evaluating a keyword combination. This function takes into account the relationships among different keywords. Experimental results show that KCEACO can efficiently find the optimal keyword combination from a large number of candidate combinations.**

Keywords-ant colony optimization; concept learning; feature selection; keyword combination extraction; text categorization

I. INTRODUCTION

In recent years, a large number of documents have been stored in the form of electronic media such as web documents, digital libraries, and electronic books [1]. Manual categorization of these documents for retrieving information in a much more flexible and faster way is very time-consuming and costly. Therefore, automated text categorization (TC) techniques have become quite promising. TC aims to automatically assign documents with the most suitable categories [2]-[8]. Bayes probabilistic approaches [3], decision trees [4], support vector machine (SVM) [5] and neural networks [6] have been applied to construct TC systems.

Unfortunately, the TC systems based on the above-mentioned methods are usually not interpretable. When the user input a document into the TC system, the document is automatically assigned with a suitable category. However, the reason for such an assignment is completely unknown by the user, because the TC system does not provide any description of the text categories.

Concept learning in TC can be viewed as acquiring the definition of each category from already categorized document samples [9]. Generally, keywords often provide semantic metadata that summarize and characterize documents [1]. An optimal group of keywords, or termed an optimal keyword combination can be extracted from a target category as a concept for defining the category [9]. The keywords of a category provide significant information about the reason why a document should be assigned with the category. If a document contains most of keywords of a category, the relationship between the document and the category is close.

The most popular method to extract keywords from a target text category is to evaluate the relevance of each individual word to the target category and then select the words with the largest evaluation values as keywords [10][11]. Lee and Kim [10] proposed a new model to evaluate individual words. Based on the proposed model, they successfully extracted main features and removed meaningless words from the text domain. Yuan and Yuan [11] introduced a novel concept mapping method based on core words to improve the traditional TC methods. Each word was evaluated by a function and the words with the largest evaluation values were selected as core words. The experimental results indicated that the information included in core words effectively improved the precision of the TC system.

The above keyword extraction methods, including Lee and Kim's method [10] and Yuan and Yuan's method [11], are based on an assumption that the occurrences of words are independent with each other. In fact, the relationships among individual words are intricate. For example, in the target category, the probabilities that words A, B and C appear in a document are 0.9, 0.8 and 0.7 respectively. A document containing word C always contains word A at the same time, but A and B, or B and C seldom appear in a document simultaneously. In this case, the combination of A and C is probably better than other combinations. Therefore, the combination consisting of the best individual words is not always the best.

To find the truly optimal keyword combination, An and Chen [9] attempted to enumerate all possible combinations with a brute-force method. However, difficulties on how to evaluate a keyword combination impartially and how to reduce the computation complexity remain unsolved. In their method, if a combination appears in any non-target category, it is discarded. This restriction is too rigid that a large number of combinations are excluded. What is more, since the number of possible combinations is so large, searching the truly optimal combination is an NP-hard problem [12]. Although it was claimed that their algorithm could be

978-0-7695-3879-2/09 $26.00 © 2009 IEEE
DOI 10.1109/SoCPaR.2009.90

conducted efficiently by introducing a pruning strategy [9], the pruning strategy is based on the rigid restriction in their evaluation method and cannot be applied in other situations. In addition, the dataset used in their experiments is very small. As the size of the corpus increase, their brute-force method becomes infeasible even with the pruning strategy.

This paper aims to design an evaluation function for keyword combinations and propose a novel algorithm, termed keyword combination extraction based on ant colony optimization (KCEACO) to efficiently search the optimal combination. The evaluation function regards a keyword combination as a whole and measures the relevance of the combination to the target category. The combination with more bias to the target category has a larger evaluation value. Ant colony optimization (ACO) [13]-[17] has been successfully applied to a large number of NP-hard problems [14][15][18]-[21]. The proposed algorithm adopts the basic idea of ACO. A set of ants construct keyword combinations in parallel by adding candidate keywords step by step. Each ant selects candidate keywords according to both information obtained from the past experience and a greedy heuristic. Past experience is represented by pheromone deposited by ants on candidate keywords, while heuristic information is calculated by the introduced evaluation function. Experimental results show that KCEACO can search the optimal keyword combination in a much more efficient way.

The remainder of this paper is organized as follows. Section II introduces the definition of the keyword combination extraction problem, as well as our evaluation function. Section III describes the implementation of KCEACO that searches the globally optimal solution from all possible keyword combinations in an efficient way. Section IV shows the experimental results and makes a discussion. Section V draws a conclusion and presents the future work.

II. KEYWORD COMBINATION EXTRACTION

In this section, we define the keyword combination extraction problem considered in this paper and introduce our evaluation function.

A. Definition of the Keyword Combination Extraction Problem

Suppose D is a set of text documents. $W = \{w_1, ..., w_n\}$ is a set of candidate keywords extracted from D, where n is the number of candidate keywords. $C = \{c_1, \ldots, c_{|C|}\}$ is a set of pre-defined text categories. Given a target category $c_t \in C$, the keyword combination extraction problem is to find an optimal keyword combination, maximizing an evaluation function, i.e.,

$$\text{maximize}\, F(S_i, c_t), \text{ satisfying } S_i \subseteq W \text{ and } |S_i| = m \,, \quad (1)$$

where $S_i = \{w_{i1}, w_{i2}, \ldots, w_{im}\}$, $w_{ij} \in W$, $j = 1,2,\cdots,m$ and F is the evaluation function that measures the relevance of a keyword combination to the target category. m is the number of candidate keywords in each keyword combination.

B. Evaluation Function

In this paper, we extend an evaluation function originated in feature selection techniques [7], the Galavotti-Sebastiani-Simi (GSS) coefficient [22], to evaluate a keyword combination. GSS coefficient is based on the information theory. It measures the relevance of an individual word $w_i \in W$ to the target category c_t. It is defined as

$$f(w_i, c_t) = p(w_i, c_t) \cdot p(\overline{w_i}, \overline{c_t}) - p(\overline{w_i}, c_t) \cdot p(w_i, \overline{c_t}) \,, \quad (2)$$

where $p(w_i, c_t)$ is the probability that a document contains w_i and belongs to c_t simultaneously. $p(\overline{w_i}, c_t)$, $p(w_i, \overline{c_t})$ and $p(\overline{w_i}, \overline{c_t})$ are the probabilities where the notation $\overline{w_i}$ represents the event that w_i does not appear in a document and the notation $\overline{c_t}$ stands for the event that a document does not belong to c_t.

In order to evaluate a keyword combination $S_i \subseteq W$ as a whole, the evaluation function (2) is extended as

$$F(S_i, c_t) = p(S_i, c_t) \cdot p(\overline{S_i}, \overline{c_t}) - p(\overline{S_i}, c_t) \cdot p(S_i, \overline{c_t}) \,. \quad (3)$$

If the occurrence of S_i concentrates on the target category c_t, the value of $p(S_i, c_t) \cdot p(\overline{S_i}, \overline{c_t})$ is relatively large and the evaluation value is competitive. However, if the occurrence of S_i concentrates on the non-target categories, the value of $p(\overline{S_i}, c_t) \cdot p(S_i, \overline{c_t})$ is much more influential so that the evaluation value is relatively small.

III. KEYWORD COMBINATION EXTRACTION BASED ON ANT COLONY OPTIMIZATION

In this section, we describe in detail the proposed algorithm, keyword combination extraction based on ant colony optimization (KCEACO).

A. Representation of a Keyword Combination

A keyword combination $S_i \subseteq W$ is represented as a set of corresponding candidate keywords. The number of candidate keywords is determined by a parameter n. The size of each combination is restricted to a parameter m.

In the current implementation, a combination is represented by a binary string, of which the length is equivalent to the parameter n. Each bit corresponds to a candidate keyword. Bits with values 1 and 0 imply that the corresponding candidate keyword is selected and excluded respectively.

B. Heuristic Information and Pheromone

In our proposed algorithm, each candidate keyword $w_i \in W$ is associated with a greedy heuristic value $\eta(w_i)$, which represents the local quality of w_i. The value of $\eta(w_i)$ is set the same as the value calculated by (2), that is,

$$\eta(w_i) = f(w_i, c_t)\,. \tag{4}$$

The pheromone in our proposed algorithm is also deposited on candidate keywords. The pheromone on $w_i \in W$ is denoted as $\tau(w_i)$. Pheromone represents the accumulated experience of the ants during the past keyword combination construction process. The better the quality of a combination constructed by an ant, the higher the amount of pheromone deposited on the corresponding candidate keywords. As time goes by, the best candidate keywords will have more and more pheromones. The initial pheromone value of each candidate keyword is set the same as:

$$\tau_0 = F(S_{init}, c_t) / m\,, \tag{5}$$

where S_{init} is the combination consisting of m candidate keywords with the largest heuristic values. The above initialization strategy is originated from [15].

C. Major Steps in KCEACO

With the aid of the flowchart in Fig. 1, the major steps of the proposed KCEACO are described as follows.

1) Step 1 – Initialization.

At the initialization stage, we have two main tasks:

a) Initialization of the Set of Candidate Keywords.

Firstly, all individual words are extracted from the original text domain to generate a vocabulary. The initial vocabulary is then preprocessed to remove stop words that are usually articles and pronouns and impossible to become keywords. Subsequently, calculate the evaluation value of each individual word according to (2) and select the first n words with largest evaluation values as candidate keywords.

b) Initialization of the Settings of KCEACO.

Before the start of the algorithm, a series of parameters, for example, the population size P and the maximum number of generations G, are initialized. A generation comprises the processes from keyword combination construction to global pheromone update (showed in Fig. 1). The heuristic information η and pheromone τ associated with each candidate keyword are initialized according to (4) and (5). Other parameters in KCEACO will be described in detail in the following sub-sections.

2) Step 2 – Keyword Combination Construction.

In each generation, each ant in the population keeps adding one candidate keyword to its current partial combination until its size has reached the parameter m. The l-th ant chooses the next candidate keyword by applying the rule:

$$w_{next} = \begin{cases} \arg\max_{w_k \in J(l)} \{\tau(w_k) \cdot \eta(w_k)^{\beta}\}, & \text{if } rnd < q_0 \\ random\_selection(\,), & \text{otherwise} \end{cases} . \tag{6}$$

$J(l)$ is the feasible set of candidate keywords in the remaining construction process of the l-th ant. The parameter

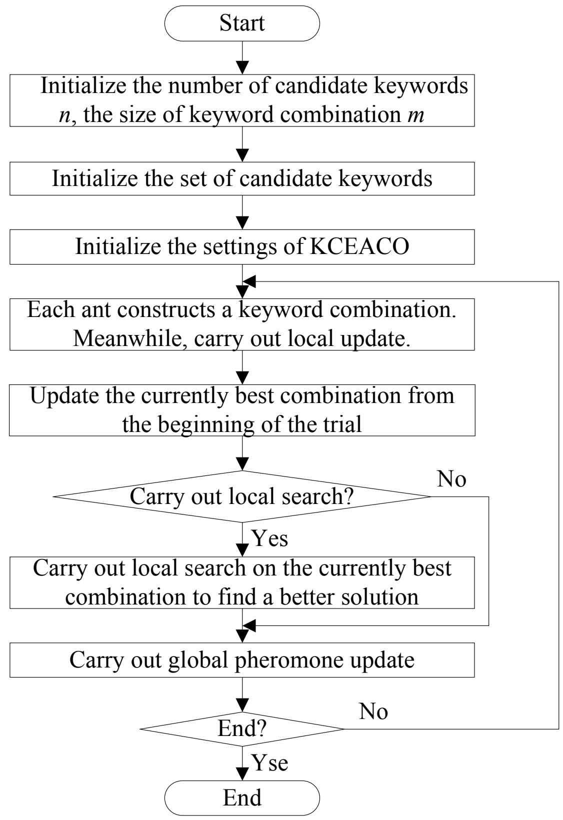


Figure 1. Flowchart of KCEACO

β adjusts the weights between the pheromone and the heuristic information. *rnd* is a randomly generated number uniformly distributed in the range [0, 1]. q_0 is a parameter that satisfies $0 \le q_0 \le 1$.

Firstly, *rnd* is randomly generated. If *rnd* is smaller than q_0, the l-th ant directly chooses the candidate keyword with the largest value of $\tau \cdot \eta^{\beta}$. Otherwise, it carries out the process *random_selection*(), in which the candidate keyword $w_i \in J(l)$ is selected with probability

$$p(w_i) = \frac{\tau(w_i) \cdot \eta(w_i)^{\beta}}{\sum_{w_k \in J(l)} \tau(w_k) \cdot \eta(w_k)^{\beta}}, \quad w_i \in J(l)\,. \tag{7}$$

Thus the parameter q_0 determines the frequency that the process *random_selection*() is activated.

Once an ant selects a candidate keyword $w_i \in W$, it modifies the associated pheromone by a local pheromone update operation:

$$\tau(w_i) = (1-\rho) \cdot \tau(w_i) + \rho \cdot \tau_0\,, \tag{8}$$

where $0 < \rho < 1$ is a parameter that controls the change rate of the pheromone. τ_0 is the initial pheromone value. The role of local update is to keep the diversity of keyword combinations constructed by the ant colony. Once a candidate keyword is selected by an ant, its pheromone will be slightly reduced. The probabilities that other candidate keywords are selected will increase. What is more, the pheromone of w_i decreases more quickly with a greater parameter ρ .

3) Step 3 – Local Search

In order to keep KCEACO from being trapped into local optima, a local search operation called *k*-local search is designed. Suppose $S_i \subseteq W$ is the keyword combination under consideration. Then *k*-local search is defined as exchanging arbitrary k candidate keywords in S_i and $(W - S_i)$ in order to find a combination better than S_i.

Such a local search operation needs to evaluate $C_m^k \cdot C_{n-m}^k$ newly generated combinations. As k grows, the increased computational cost becomes enormous. In the current implementation, we set $k = 1$. What is more, the local search operation is activated once every G_p generations.

4) Step 4 – Global Pheromone Update

After all ants have constructed their keyword combinations, global pheromone update is carried out:

$$\tau(w_i) = (1-\alpha)\cdot\tau(w_i) + \alpha\cdot F(S_{bs}, c_t), \quad \forall w_i \in S_{bs}, \quad (9)$$

where $0 < \alpha < 1$ is a parameter that adjusts the effect of global pheromone update. S_{bs} is the best-so-far combination found from the beginning of the trial. In global pheromone update, those candidate keywords in S_{bs} will receive reinforcement. Thus the search will explore within the neighborhood of S_{bs} . The parameter α determines the degree of the reinforcement. S_{bs} receives greater enhancement with a larger α .

D. Timing Analysis and Efficiencies

In the proposed KCEACO, the most time-consuming operation is the calculation of the evaluation value of a keyword combination especially when the size of the corpus is large. Thus we define such a calculation process as a computation unit. The total number of possible combinations is $C_n^m = n!/[m!(n-m)!]$. In order to find the optimal combination, a brute-force method always requires C_n^m computation units in each run.

The total number of computation units required by KCEACO is $[P\cdot G' + (m\cdot n - m^2)\cdot G'/G_p]$, where G' is the number of generations required until the optimal combination is found. The local search operation introduces the cost of $[(m\cdot n - m^2)\cdot G'/G_p]$ computation units.

IV. Experiments

In this section, we compare the computational cost of the brute-force method and that of KCEACO through experiments. The corpus used in our experiments is the Reuters-21578 collection [7]. It assembles the economic news published on the Reuters newswire in 1987. The original corpus contains 21578 documents and provides five classification criteria, that is, *exchanges*, *organization*, *people*, *places* and *topics*. Since the *topics* of the documents have been used as classification criteria in almost all researches that apply the Reuters data, we also use the *topics* for categorization in our experiments.

After preprocessing, the number of documents, categories and words are 11367, 120, and 32788 respectively. Table I shows the top five categories with the largest number of documents. In all experiments, we choose *acquisition* as the target category c_t.

TABLE I. The Top Five Categories with the Most Documents after Preprocessing

Category	*earn*	*acquisition*	*money-fx*
No. of Documents	3987	2448	801
Category	*crude*	*grain*	
No. of Documents	634	628	

A. Comparison between KCEACO and the Brute-force Method

In this experiment, we compare the efficiency of KCEACO and that of the brute-force search method. Firstly, the brute-force method is applied to find the truly optimal combination. The number of candidate keywords n and the size of each combination m are set to 20 and 5 respectively. The combination consisting of the m candidate keywords that have the largest evaluation values is denoted as S_{init}. The optimal combination found by the brute-force method is denoted as S_{opt}. The set of candidate keywords, S_{init} and S_{opt} in this experiment are summarized in Table II. It should be noted that the evaluation value of S_{init} is much smaller than that of S_{opt}. It proves that the combination constructed by the candidate keywords with the largest evaluation values is not always the optimal combination.

TABLE II. The Set of Candidate Keywords, S_{init} and S_{opt} when n and m Are 20 and 5 Respectively

Candidate Keywords	{*corporation*, *company*, *stock*, *group*, *acquisition*, *common*, *share*, *offer*, *stake*, *cash*, *terms*, *shareholders*, *outstanding*, *merger*, *buy*， *securities*, *subsidiary*, *purchase*, *agreement*, *sell*}
S_{init} **(Evaluation Value)**	{*corporation*, *company*, *stock*, *group*, *acquisition*} (0.001845)
S_{opt} **(Evaluation Value)**	{*corporation*, *company*, *stock*, *share*, *offer*} (0.005597)

Although the brute-force method can find the S_{opt}, the searching process is terribly slow. KCEACO can search the S_{opt} in a much more efficient way. Table III compares the efficiency of the brute-force method and that of KCEACO when n varies from 10 to 20. It should be noted that the optimal combination of each case is still the S_{opt} showed in Table II. The parameters in KCEACO are given in Table IV. Limited preliminary experiments were carried out for setting the parameters and they are insensitive in our experiments.

TABLE III. COMPARISION BETWEEN THE EFFICIENCY OF THE BRUTE-FORCE METHOD AND THAT OF KCEACO WHEN n VARIES FROM 10 TO 20

	No. of Computation Units			Empirical Computation Time (sec.)		
	Brute-force	***KCEACO Avg.***	***KCEACO Max.***	***Brute-force***	***KCEACO Avg.***	***KCEACO Max.***
n **= 10**	252	51	315	28	8	44
n **= 12**	792	75	355	85	11	45
n **= 14**	2002	157	670	203	22	92
n **= 16**	4368	242	1100	421	33	150
n **= 18**	8568	254	2110	789	34	264
n **= 20**	15504	508	2010	1370	65	256

The maximal number of generations G is not considered and the algorithm terminates when the S_{opt} is found. For each case, 50 independent runs of KCEACO are carried out. All cases are run on a computer with an Intel Core 2 1.86 GHz CPU. We compare both the number of computation units and the empirical computation time of the brute-force method and KCEACO. A computation unit is defined in Part D of Section III. The computation time is the practical time consumed during the whole searching process. It can be observed that the efficiency of KCEACO significantly outperforms that of the brute-force method.

TABLE IV. THE PARAMETERS IN KCEACO

P	G	G_p	α	ρ	β	q_0
10	500	50	0.02	0.1	1	0.2

Fig. 2 illustrates the searching process of KCEACO when n is 20. It shows the optimization curve of the best-so-far combination. The evaluation values are averaged over 50 independent runs. The dash line indicates the evaluation value of S_{opt}. At the initial stage, the evaluation value of the best-so-far combination is relatively small since each candidate keyword is associated with the same amount of initial pheromone. After a few generations, the evaluation value of the best-so-far combination increases sharply. Once a better combination is found, the global pheromone update reinforces the candidate keywords in this combination and leads the search to a more promising direction. Additionally, both of the local pheromone update and the local search prevent KCEACO from being trapped into local optima. It can be seen from the figure that the evaluation value of S_{opt} is reached within 200 generations.

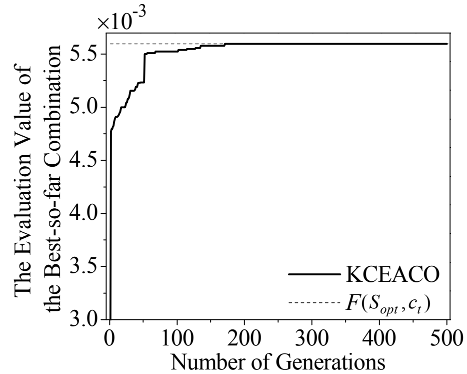
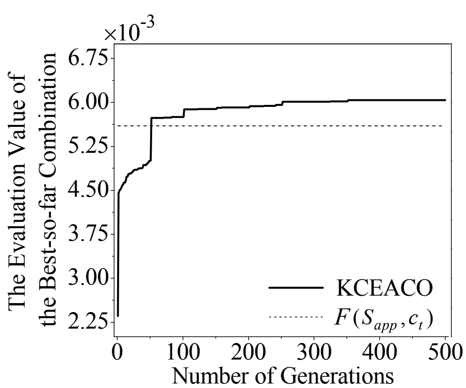


Figure 2. The searching process of KCEACO when n is 20

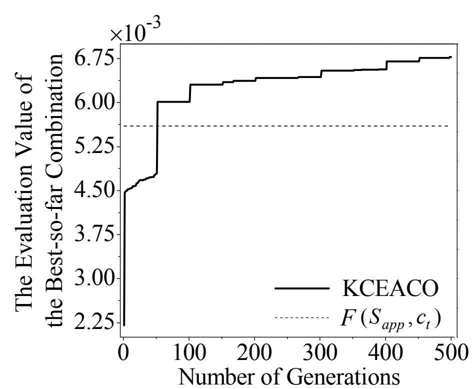
B. Searches in Larger Solution Spaces

We enlarge the searching space to verify the efficiency of KCEACO by using experiments on n=30, m=5 and n=50, m=5. The numbers of computation units required by the brute-force method are C_{30}^{5} and C_{50}^{5} respectively, which are so large that the brute-force method is infeasible.

Fig. 3 shows the searching processes of KCEACO in both cases. Because the truly optimal combination is unknown, we define an approximately optimal combination S_{app}, which equals to the S_{opt} in the previous sub-section when n is 20. In each case, 50 independent runs are carried out to observe the searching process. The parameter settings are the same as those in Table IV. The dash line in Fig. 3 shows the evaluation value of S_{app}. With the enlargement of the searching space, the convergence of KCEACO becomes



(a) n **= 30**



(b) n **= 50**

Figure 3. The searching processes of KCEACO in larger spaces

slower. But it is more likely to find a keyword combination better than S_{app} after G generations. Table V shows the statistics of the 50 runs in both cases. Table VI summarizes the best combinations found in the 50 runs and their corresponding evaluation values. The larger the number of candidate keywords, the greater the number of possible keyword combinations, and the better the optimal combination found in the 50 runs.

TABLE V. THE STATISTICS OF KCEACO IN THE 50 RUNS IN BOTH ENLARGED SEARCHING SPACES.

	The No. of Runs that Find S_{app}	**The No. of Runs that Find a Combination Better Than S_{app}**
n **= 30**	33	17
n **= 50**	32	18

TABLE VI. THE BEST COMBINATION FOUND IN THE 50 RUNS IN BOTH ENLARGED SEARCHING SPACES

	The Best Combination Found in the 50 Runs	**Evaluation Value**
n **= 30**	{*stock*, *common*, *stake*, *securities*, *commission*}	0.006884
n **= 50**	{*stock*, *common*, *securities*, *commission*, *exchange*}	0.009017

V. CONCLUSION AND FUTURE WORK

In this paper, we extend an evaluation function originated in feature selection techniques to measure the relevance of a keyword combination to a target category. Additionally, a novel algorithm, KCEACO is proposed to search the optimal keyword combination in a much more efficient way. Our future work is to further enhance the performance of the proposed algorithm and extend the application of the algorithm for other corpora.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China (NSFC) Joint Fund with Guangdong under Key Project U0835002, in part by the National High-Technology Research and Development Program ("863" Program) of China no. 2009AA01Z208. and in part by the Internal Competitive Research Grant, PolyU (Project Code: G-YG39).

REFERENCES

[1] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "KEA: practical automatic keyphrase extraction," Proc. DL '99, pp. 254-256, 1999.

[2] S. Tasci and T. Gungor, "An evaluation of existing and new feature selection metrics in text categorization," 23rd IEEE Int'l Symposium on Computer and Information Sciences, pp. 1-6, 2008.

[3] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorisation," Proc. 14th Int'l Conf. Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 412-420, 1997.

[4] D. D. Lewis and M. Ringuette, "Comparison of two learning algorithms for text categorization," In Proc. of the 3rd Annual Symposium on Document Analysis and Information Retrieval, pp. 81-93, 1994.

[5] Y. Yang and X. Liu, "A re-examination of text categorization methods," In Proc. of SIGIR-99, 22nd ACM Int'l Conf. on Research and Development in Information Retrieval, pp. 42-49, 1999.

[6] E. Wiener, J. O. Pedersen, and A. S. Weigend, "A neural network approach to topic spotting," In Proc. of the 4th Annual Symposium on Document Analysis and Information Retrieval, 1995.

[7] F. Sebastiani, "Machine learning in automated text categorisation," ACM Computing Survey, vol. 34, no. 1, pp. 1-47, 2002.

[8] M. Emms and S. Luz, "Machine learning for natural language processing," ESSLLI 2007 Course Reader, Trinity College Dublin, Aug. 2007.

[9] J. An and Y.-P. P. Chen, "Keyword extraction for text categorization," Proc. of the 2005 IEEE Int'l Conf. on Active Media Technology, pp. 556-561, 2005.

[10] S. Lee and H.-J. Kim, "News keyword extraction for topic tracking," 4th IEEE Int'l Conf. on Networked Computing and Advanced Information Management, vol. 2, pp. 554-559, 2008.

[11] F. Yuan and J.-Y. Yuan, "A concept mapping method based on core words of class," IEEE Int'l Conf. on Machine Learning and Cybernetics, vol. 6, pp. 3366-3371, 2007.

[12] M. R. Garey and D. S. Johnson, computers and intractability: a guide to the theory of NP-completeness, San Francisco, Freeman, 1979.

[13] M. Dorigo, "Optimization, learning and natural algorithms," Ph.D. dissertation (in Italian), Dip. Elettronica, Politecnico di Milano, Milano, Italy, 1992.

[14] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," IEEE Trans. Syst, Man, Cybern. B, vol. 26, no. 2, pp. 29-41, 1996.

[15] M. Dorigo and L. M. Gambardella, "Ant colony System: a cooperative learning approach to the traveling salesman problem," IEEE Trans. Evol. Comput., vol. 1, no. 1, pp. 53-66, 1997.

[16] T. Stützle and H. H. Hoos, "Improving the ant-system: a detailed report on the MAX-MIN ant system FG intellektik," TU Darmstadt, Ger-many, Tech. Rep. AIDA 96-12, 1996.

[17] Y. Lin, J. Zhang, and J. Xiao, "A pseudo parallel ant algorithm with an adaptive migration controller," Applied Mathematics and Computation, vol. 205, pp. 677-687, 2008.

[18] J. Zhang, X.-M. Hu, X. Tan, J. H. Zhong, and Q. Huang, "Implementation of an ant colony optimization technique for job scheduling problem," Transactions of the Institute of Measurement and Control, vol. 28, no. 1, pp. 93-108, 2006.

[19] W.-N. Chen and J. Zhang, "Ant colony optimization approach to grid workflow scheduling problem with various QoS requirements," IEEE Trans. Syst., Man, and Cybernetic-Part C: Applications and and Reviews, vol. 31, no. 1, pp. 29-43, Jan 2009.

[20] X.-M. Hu, J. Zhang, and Y. Li, "Orthogonal methods based ant colony search for solving continuous optimization problems," Journal of Computer Science and Technology, vol. 23, no. 1, pp. 2-18, 2008.

[21] J. Zhang, H. S.-H. Chung, A. W.-L. Lo, and T. Huang, "Extended ant colony optimization algorithm for power electronic circuit design," IEEE Trans. Power Electron., vol. 24, no, 1, pp. 147-162, 2009.

[22] L. Galavotti, F. Sebastiani, and M. Simi, "Experiments on the use of feature selection and negative evidence in automated text categorization," Proc. 4th European Conf. Research and Advanced Technology for Digital Libraries, Springer Verlag, pp. 59-68, 2000.