A Comparative Study on Statistical Machine Learning Algorithms and Thresholding Strategies for Automatic Text Categorization

Kang Hyuk Lee[1], Judy Kay[1], Byeong Ho Kang[2], and Uwe Rosebrock[2]

[1] School of Information Technologies, University of Sydney, NSW 2006, Australia
{kangl, judy}@it.usyd.edu.au

[2] School of Computing, University of Tasmania, Hobart, Tasmania 7001, Australia
{bhkang, uwer}@utas.edu.au

Abstract. Two main research areas in statistical text categorization are similarity-based learning algorithms and associated thresholding strategies. The combination of these techniques significantly influences the overall performance of text categorization. After investigating two similarity-based classifiers (k-NN and Rocchio) and three common thresholding techniques (RCut, PCut, and SCut), we describe a new learning algorithm known as the keyword association network (KAN) and a new thresholding strategy (RinSCut) to improve performance over existing techniques. Extensive experiments have been conducted on the Reuters-21578 and 20-Newsgroups data sets. The experimental results show that our new approaches give better results for both micro-averaged F_1 and macro-averaged F_1 scores.

1 Introduction

The goal of text categorization is to learn a classification scheme that can be used for the problem of automatically assigning arbitrary documents to predefined categories (or classes). Text categorization has many applications in which this plays a vital role, such as document routing, document management, and document dissemination. Traditionally, experts who are knowledgeable about the categories conduct text categorization manually. This requires substantial human resources. Given that the amount of online textual information is growing rapidly, the need for reliable automatic text categorization has been increasing.

There has been a wide range of statistical learning algorithms applied to this automatic text categorization task. They include the Rocchio relevance feedback algorithm [2, 4, 7], k-Nearest Neighbor (k-NN) classification [15, 19], naive Bayes probabilistic classification [4, 6, 8], support vector machines [5], and neural networks [14]. After two preprocessing steps (representation and feature selection), answering the questions of how to discriminate informative words in the reduced vector space and how to give them more weight than other non-informative words is the main task of classifiers. After exploring two representatives of the similarity-based classifiers, namely Rocchio and k-NN, this paper describes the keyword association network (KAN), a new representation and learning algorithm, designed to effectively address these open questions.

The last step, to obtain a mapping from a new document to relevant categories, is achieved with thresholding techniques applied to the similarity score for each document-category pair. Existing common techniques are rank-based thresholding (RCut), proportion-based assignment (PCut), and score-based optimization (SCut). These techniques have been extensively evaluated on various corpora in [16, 17]. The choice of thresholding strategy has a significant impact on the performance of classifiers. This choice is also influenced by the characteristics of dataset. This means that finding the optimal thresholding strategy for any given classifier and data set is difficult and combining the strengths of the existing thresholding strategies is a challenge in text categorization [17]. This paper presents a new thresholding technique (RinSCut) that combines the strengths of RCut and SCut to overcome the weaknesses of two thresholding strategies. The empirical results on the Reuters-21578 and 20-Newsgroups data sets show that our new approaches outperform existing techniques.

2 Reviews on Common Techniques

2.1 Representation and Feature Selection

The common representation adopted by most statistical learning algorithms is the "bag-of-words" representation. In this representation, each document D is transformed to have the form of a vector $d = (v_1, v_2, \dots, v_n)$. Here, each v_i is the weighting value of the ith feature (term or word) and n is the total number of features in D. The weights are calculated as a combination of two common weighting schemes, $TF(i,D)$ and $IDF(i)$. The term frequency $TF(i,D)$ is the number of times the ith feature occurs in document D and the inverse document frequency, $IDF(i)$, is $log\{|N| / DF(i)\}$, where $DF(i)$ is the number of documents in which the ith feature occurs at least once and $|N|$ is the total number of documents in the training set. Because the document lengths may vary widely, a length normalization factor is applied to the term weighting function. The weighting equation that is used [3] in this experiment is given as:

$$v_i(d) = \frac{[\, logTF(i, D) + 1.0 \,] \times IDF(i)}{\sqrt{\sum_{i=1,n} \{[logTF(i, D) + 1.0] \times IDF(i)\}^2}} \qquad \textbf{(1)}$$

Typically, this vector space is very high dimensional and this makes it computationally intractable to apply most statistical learning algorithms. So, it is critical to reduce this huge vector space before applying statistical algorithms. This dimension reduction, known as feture selection, can be achieved by using the following techniques: document frequency, term frequency, mutual information, information gain, and OddsRatio etc. [9, 18]. Then, the main task of learning algorithms is to find informative features for the resulting reduced vector space.

2.2 Statistical Learning Algorithms: Rocchio and k-NN

The Rocchio classifier is based on a relevance feedback algorithm [11]. Because of its various heuristic components, there have been several similar algorithms corresponding to the particular choice of heuristics. In this algorithm, each category C has a vector of the form $c = (x_1, x_2, \ldots , x_n)$. This prototype vector c is prepared by summing the vectors of the positive documents as well as of the negative documents and, then by calculating a weighted difference for each.

$$c = [\alpha \times |C|^{-1} \times \textstyle\sum_{d \in C} d] - [\beta \times (|N| - |C|)^{-1} \times \textstyle\sum_{d \notin C} d] \qquad \textbf{(2)}$$

where α and β are adjustment parameters for positive and negative examples, d is the vector of document, and $|C|$ is the number of documents in the category C. The similarity value between a category and a new document is obtained as the inner product between corresponding feature vectors. The problem in this classifier is that some relevant features in a rare category will have small weights if they appear equally in the negative document set. It is also very sensitive to the number of irrelevant words, since all features participate equally in the similarity calculation. As a result, if the set of discriminating features of a category is only a small subset of the overall vector space, the category will very likely have low performance.

K-nearest neighbor. The k-Nearest Neighbor (k-NN) classifier is an instance-based learning algorithm. The main idea of this algorithm is that a document itself has more representative power than the generalized category feature vector. For a new document, it computes the similarity to all the training documents using the cosine metric used in [15]. The similarity scores are then sorted in descending order. The final score for a category is the summation of the similarity scores of the documents of that category in the k top-ranking documents. One drawback of this algorithm is that noisy examples have direct impact on the quality of the ranking. Furthermore, the time taken for the similarity calculation increases in proportion to the size of the training data set. Like Rocchio, the k-NN classifier does not cope effectively with irrelevant features that cause overfitting.

2.3 Common Thresholding Strategies

A thresholding strategy is used in the last step of the similarity-based classifiers to obtain binary assignments of categories to arbitrary documents. Rank-based Thresholding (RCut) sorts the similarity scores of categories for each document and assigns a "YES" decision to the t top-ranking categories. Using a validation set or a training set, the threshold, t, is predefined automatically by optimizing the global performance, not the local performance of each category. RCut will give the best performance when all the test documents belong to the same number of categories. However, when documents have a variable number of categories this strategy may result in a low macro-averaged performance.

Given a ranking list of each category (c_i), Proportion-based Assignment (PCut) assigns a "YES" decision to k_i top-ranking test documents. The threshold, k_i, is $n \times P(c_i) \times \omega$ where n is the number of documents in a validation set or a training set, $P(c_i)$ is the probability of the ith category, and ω is the real-valued parameter given by the user or predetermined automatically in the same way as t for RCut. While performing well in the text categorization experiments [16], PCut cannot be used for online categorization.

Score-based Optimization (SCut) learns the optimal threshold for each category. The optimal threshold is the similarity score that optimizes the performance measure of each category. If documents belong to a variable number of categories and the local performance of each category is the primary concern, this strategy may be a better choice than RCut. However, it is not trivial to find an optimal threshold, and this problem becomes more apparent with the small set of training data. As a result, SCut has a potential weakness in overfitting to the training set.

3 Keyword Association Network (KAN)

3.1 Need for New Learning Algorithm

We noticed that a crucial question in statistical text categorization is how to cope with irrelevant features effectively. The answer to this question could be considered at the level of feature and category. With respect to the feature level, we have focused on how to find semantics and to assess the importance of a feature in a given document. And, with respect to the level of category, we focused on how to establish the discriminating features in each category and how to represent these features using a suitable representation.

To give a feature an appropriate weight according to both its relevant meaning and its importance, our main focus lies on the collocating features in a given document. As an example of capturing the correct meaning, consider the word, “apple”, that conveys different meanings in a category and a document. Suppose the feature “apple” in the document is in the context Farm and, in the category it is in the context Computer. By looking at other words in the document, a human can differentiate between the semantics of “apple” in the document and the same word in the category. This approach can be applied to measuring the importance of words in a given document. Because current farming uses computer technology, this document in the context Farm might contain several words overlapping with the Computer category, and those overlapping words should have minor importance in the document. If we adopt the statistical approaches already described, the similarity measure between the document and the category may be high and, as a result, this document may be incorrectly considered as being Computer-related. As a result, the similarity measurement without considering the features in the context of the document will lead to an incorrect classification.

The important aspect at the level of category is that the most critical feature set size is different for different categories and it is relatively small. Some categories have one or two discriminating words. Identifying the existence of those words in the documents is enough for a categorization. Using the same feature set size across all the categories could be a major cause of low system performance. It suggests that using a different representative feature set size for different categories would lead to higher performance for a classifier, if the proper size can be determined and a suitable representation method can be found.

3.2 New Representation and Learning Algorithm: KAN

Previous work showed that it is possible to automatically find words that are semantically similar to a given word based on the collocation of words [12, 13]. Our goal was to use this type of statistical information to determine the importance and semantic meaning of a word in a given document. KAN is constructed by means of a network representation based on statistical information. The degree of relationship between two features is represented by a confidence value. This measure was used in finding association rules [1] that have been identified as an important tool for knowledge discovery in huge transactional databases. In KAN, the confidence value is used for measuring how the presence of one word in a given document may influence the presence of another. When the category C has a set of k unique features $\{W = (w_1, w_2, \dots, w_k)\}$, the construction and utilization of KAN for the text categorization is based on the following statistical information: the support, the confidence, and the discriminative power functions. They are defined as follows.

Definition 1. For the feature w_i, the positive support, $SUP_P(w_i)$, is the number of documents in a category for the training set that contain w_i and the total support, $SUP_T(w_i)$, is the number of documents that contain w_i in the complete training set.

Definition 2. The confidence value of w_i to w_j in a category, $CONF(w_i,w_j)$, is the proportion of positive documents which contain w_i and also have w_j, i.e., $SUP_P(w_i, w_j) / SUP_P(w_i)$.

High confidence w_i to w_j can be interpreted as indication that the meaning and importance of w_j is associated with the existence of w_i. In each category, the discriminative features are identified automatically using two discriminative power functions *DPF1* and *DPF2* which are defined as follows.

Definition 3. Two discriminative power functions $DPF1(w_i)$ and $DPF2(w_i)$ for feature w_i are:

$$DPF1(w_i) = SUP_P(w_i) / SUP_T(w_i) \quad (3)$$

$$DPF2(w_i) = SUP_P(w_i) / |C| \quad (4)$$

where $|C|$ is the number of documents in category C.

According to above discriminative power functions, the ideal discriminating feature for a category will appear in all the documents in the category and the number of documents in this category will be same with the number of documents which contains the feature in the training set. In KAN, if two function values of a feature satisfy the user specified minimum values, this feature is considered as a representative feature and plays an important role in the similarity calculation. Figure 1 shows an example of KAN for a particular category in Reuters-21578 when the minimum values for *DPF1* and *DPF2* are 0.5 and 0.2 respectively. In this example, the nodes “agriculture”, “grain”, and “wheat” are presented as the discriminating features satisfying two minimum values. These will provide most of the overall similarity score. An important factor in achieving high performance in text categorization is to remove the influence of the large number of irrelevant features that occur evenly across the categories. KAN is designed to distinguish them from the discriminating features in a category and use them by adding more weight to the small number of discriminating features.

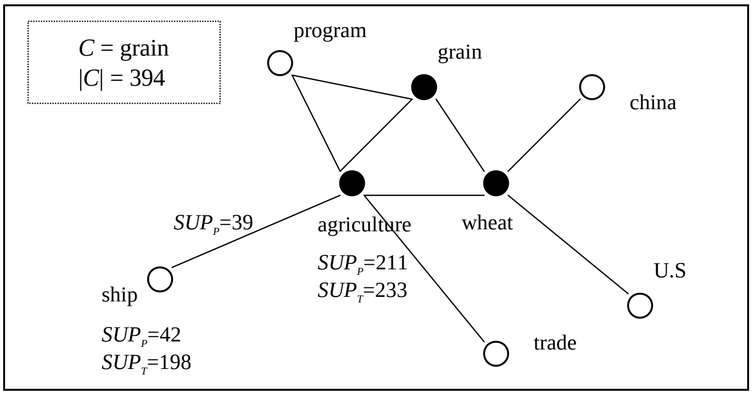


Fig. 1. An example of KAN for the grain category in Reuters-21578 dataset.

For a category C which has the unique feature set $\{W = (w_1, w_2, \ldots, w_k)\}$, the vector having the form $c = (x_1, x_2, \ldots, x_k)$ is prepared as follows: $c = \sum_{d \in C} d \times |C|^{-1}$ where d is the vector of a document calculated using the equation (1) and $|C|$ is the number of documents in the category C. The similarity score between the category C and a test document D which is represented by the vector of the form $(v_1, v_2, \ldots, v_k)$ is computed using the following function:

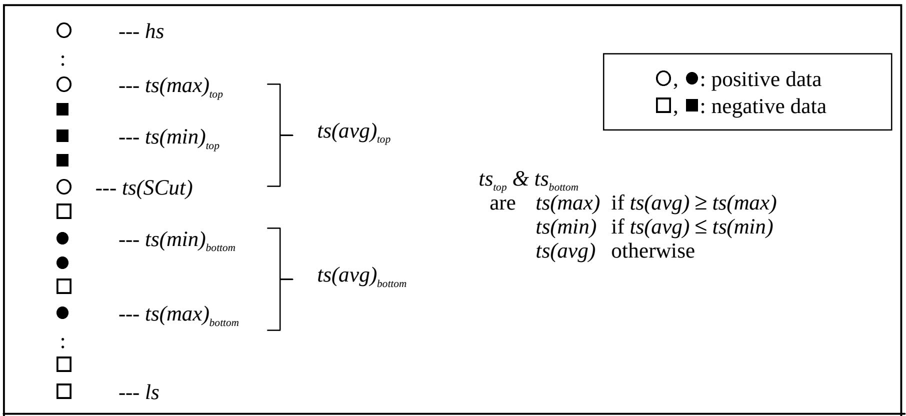
$$\text{Similarity}(D, C) = \sum_{i=1,k} [\, (x_i \times v_i) + \delta \,] \qquad (5)$$

$$\delta \text{ is } \begin{cases} \sum_{j=1,k \& j \neq i} [x_i \times CONF(w_j, w_i) \times v_i] & \text{if } w_i \in R \\ 0 & \text{otherwise} \end{cases}$$

where R is the set of discriminative features in the category C.

4 New Thresholding Strategy: RinSCut

RinSCut is designed to overcome weaknesses in both SCut and RCut. For each category, it computes the two threshold scores, ts_{top} and ts_{bottom}, as shown in Figure 2 and the range of above two threshold values is considered as the ambiguous zone. For the new documents that have the similarity scores belonging to this zone, the rank threshold, t, is used to make the final decision. The threshold, t, is predefined by optimizing the local performance of each category. As a result, unlike in RCut, each category may have different t in RinSCut. For a test document D having the similarity score, Similarity(D, C), it assigns a "YES" decision if Similarity(D, C) $\geq ts_{top}$ and a "NO" decision if Similarity(D, C) $< ts_{bottom}$. If Similarity(D, C) is between ts_{top} and ts_{bottom}, the assignment decision depends on the rank-based threshold t.



hs: highest similarity score. *ls*: lowest similarity score.
ts(SCut): optimal threshold from SCut.
$ts(max)_{top}$: $ts(SCut) + [\{hs - ts(SCut)\} \times \varphi_{max}]$ $ts(max)_{bottom}$: $ts(SCut) - [\{ts(SCut) - ls\} \times \varphi_{max}]$
$ts(min)_{top}$: $ts(SCut) + [\{hs - ts(SCut)\} \times \varphi_{min}]$ $ts(min)_{bottom}$: $ts(SCut) - [\{ts(SCut) - ls\} \times \varphi_{tmin}]$
$ts(avg)_{top}$: average score of negative data (■) having similarity scores greater than *ts(SCut)*.
$ts(avg)_{bottom}$: average score of positive data (●) having similarity scores smaller than *ts(SCut)*.
φ_{max}, φ_{min}: real-valued numbers between 0 and 1 specified by the user ($\varphi_{max} > \varphi_{min}$).

Fig. 2. Example showing how to calculate ts_{top} and ts_{bottom} in RinSCut.

5 Experiments

5.1 Data Sets

The 20-Newsgroups[1] collected by Ken Lang consists of 20,017 articles posted to 20 different Usenet discussion groups. Because each article belongs to exactly one newsgroup, RCut seems to be the optimal thresholding strategy in this data set. So, the performance of each classifier will be measured using the RCut. For the test set, 30% of the articles were randomly selected. The Reuters-21578[2] consists of 21,578 articles appeared on the Reuters newswire. We split the articles into training and test sets according to the Modified Lewis Split. Instead of analyzing all 135 categories, we choose the categories having at least 10 articles in both training and test sets. The number of selected categories is 53. This results in a corpus of 6,984 training data and 3,265 test data. In this data set, the SCut and RinSCut will be applied to each classifier, because many articles have the variable number of categories.

5.2 Experimental Setup

For the preprocessing steps, we applied a stop-list and Porter's stemming algorithm [10] to the articles. We used information gain for the feature selection and took the same number of features, 50 in these experiments, for all categories in both data sets.

[1] http://www.ai.mit.edu/people/jrennie/20_newsgroups
[2] http://www.research.att.com/~lewis/reuters21578.html

We have implemented three classifiers - Rocchio, k-NN, and KAN - with three thresholding strategies - RCut, SCut, and RinSCut. The performance was measured using the standard F_1 measure that is designed to balance recall and precision by giving them equal weight [16].

$$F_1 = 2 \times \text{Recall} \times \text{Precision} / (\text{Recall} + \text{Precision}) \qquad \textbf{(6)}$$

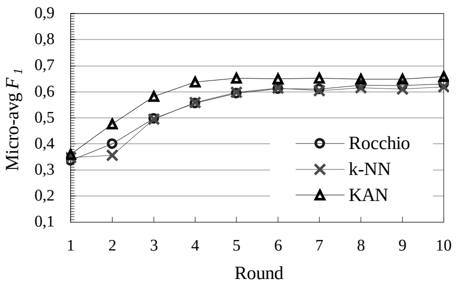
We computed the macro-averaged F_1 as well as the micro-averaged F_1 in order to analyze the performance on individual categories. We conducted the experiments by increasing the size of the training data to construct learning curves for the classifiers. Table 1 shows the amount of training data in each round. The training set for each round is a superset of the one for the previous round. For KAN, we use 0.5 as the minimum value for *DPF1* and 0.2 for *DPF2* to define the representative features in the categories. To compute the vectors of categories for the Rocchio algorithm, we use α = 16 and β = 4 as suggested in [2]. The value k used in these experiments for the k-NN is 10, 30, and 50. Then, we chose the value with the best result in each round. For RinSCut, 0.3 and 0.1 are assigned to φ_{max} and φ_{min} respectively.

Table 1. Number of unique training data in each round.

Round / Data set	1	2	3	4	5	6	7	8	9	10
Reuters	106	212	371	689	1272	2409	3696	5136	6202	6984
Newsgroups	80	160	300	580	1140	2260	3520	4920	5980	13998

5.3 Results

Figure 3 shows the micro-averaged and macro-averaged F_1 performance of each classifier with the RCut on 20-Newsgroups data set. Each classifier achieves the similar performance on both measures and it is due to the fact that articles in trainging and test sets are divided almost evenly among 20 groups. KAN gives better performance than the other classifiers although the difference is minor with the large number of training data. In Figure 4 and 5, the performance of each classifier on the Reuters-21578 data set is shown. With the SCut in Figure 4, KAN performs significantly better than k-NN and Rocchio in both micro-averaged and macro-averaged F_1. Note here on the macro-



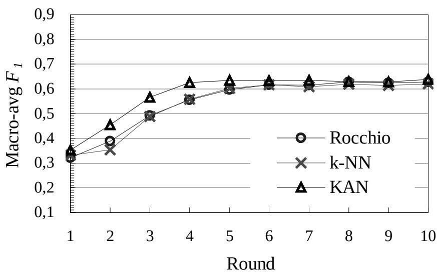


Fig. 3. Micro-averaged F_1 and Macro-averaged F_1 performance with the RCut on the 20-Newsgroups data set.

averaged measure that KAN outperforms Rocchio and k-NN after round 3. This result demonstrates that our effort to capture the characteristics of each category using a discriminative feature set is achieved, and these discriminative features in each category increase the overall macro-averaged performance. In Figure 5, RinSCut gives similar learning curves to SCut for three classifiers. On both measures, RinSCut gives better performance than SCut across all the rounds for each classifier. On the basis of the data in Figure 4-5, it seems that the best choice across techniques for the Reuters-21578 data set would be KAN with RinSCut.

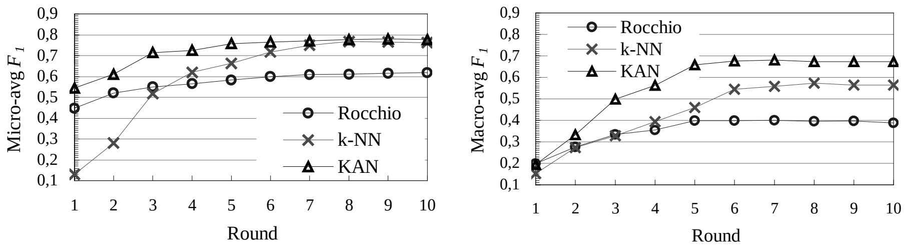


Fig. 4. Micro-averaged F_1 and Macro-averaged F_1 performance with the SCut on the Reuters-21578 data set.

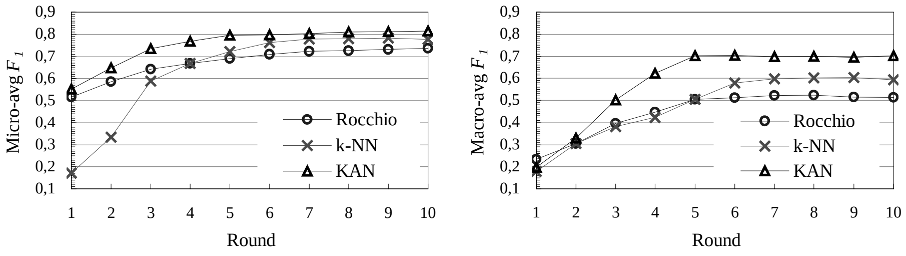


Fig. 5. Micro-averaged F_1 and Macro-averaged F_1 performance with the RinSCut on the Reuters-21578 data set.

6 Conclusions

We have explored two main research areas, statistical learning algorithms and thresholding strategies to text categorization. After outlining current techniques in both areas, we described KAN as a new representation and learning algorithm and RinSCut as a new thresholding strategy. We implemented Rocchio, k-NN, KAN with RCut, SCut, and RinSCut. Extensive experiments have been conducted on the 20-Newsgroups and Reuters-21578 data sets. Empirical results show that our new approaches outperform slightly existing other techniques. We note that the application of KAN and RinSCut in other areas could be promising, such as document routing and document filtering tasks. More research is envisaged to investigate the performance of our new approaches in these application areas.

References

1. Agrawal, A., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast Discovery of Association Rules. In: U. M. Fayyad, G. Piatetsky-Shapiro, P. Smith, R. Uthurusamy (eds). Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press (1996) 307-328
2. Buckley, C., Salton, G., Allan, J.: The Effect of Adding Relevance Information in a Relevance Feedback Environment. International ACM SIGIR Conference (1994) 292-300
3. Buckley, C., Salton, G., Allan, J., Singhal, A.: Automatic Query Expansion Using SMART: TREC 3. The Third Text Retrieval Conference (TREC-3), National Institute of Standards and Technology Special Publication 500-207, Gaithersburg, MD (1995)
4. Joachims, T.: A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In Proceedings of the 14th International Conference on Machine Learning ICML'97 (1997) 143-151
5. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In European Conference on Machine Learning (ECML-98) (1998)
6. Lewis, D.D., Ringuette, M.: Comparison of Two Learning Algorithms for Text Categorization. In Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94), Nevada, Las Vegas (1994)
7. Lewis, D.D., Schapire, R.E., Callan, J.P., Papka, R.: Training Algorithms for Linear Text Classifiers. In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1996) 298-306
8. McCallum, A., Nigam, K.: A Comparison of Event Models for Naive Bayes Text Classifiers. In AAAI-98 Workshop on Learning for Text Categorization (1998)
9. Mladenic, D., Grobelnik, M.: Feature Selection for Unbalanced Class Distribution and Naive Bayes. In Proceedings of the 16th International Conference on Machine Learning (ICML-99) (1999)
10. Porter, M.F.: An Algorithm for Suffix Stripping. Program, Vol. 14. No. 3. (1980) 130-137
11. Rocchio, J.: Relevance Feedback in Information Retrieval. In G. Salton (ed): The SMART Retrieval System: Experiments in Automatic Document Processing, Prentice-Hall (1971)
12. Ruge, G.: Experiments on Linguistically Based Term Associations. Information Processing & Management, Vol. 28. No. 3. (1992) 317-332
13. Sekine, S., Carroll, J., Ananiadou, A., Tsujii, J.: Automatic Learning for Semantic Collocation. Proceedings of the Third Conference on Applied Natural Language Processing, ACL (1992) 104-110
14. Wiener, E., Pedersen, J.O., Weigend, A.S.: A Neural Network Approach to Topic Spotting. In Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95) (1995)
15. Yang, Y.: Expert Network: Effective and Efficient Learning from Human Decisions in Text Categorization and Retrieval. In Proceedings of the 17th International ACM SIGIR Conference on Research and Development in Information Retrieval (1994) 13-22
16. Yang, Y.: An Evaluation of Statistical Approaches to Text Categorization. Journal of Information Retrieval, Vol. 1. No. 1/2. (1999) 67-88
17. Yang, Y.: A Study on Thresholding Strategies for Text Categorization. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01) (2001)
18. Yang, Y., Pedersen, J.O.: Feature Selection in Statistical Learning of Text Categorization. In 14th International Conference on Machine Learning (1997) 412-420
19. Yavuz, T., Guvenir, A.: Application of k-Nearest Neighbor on Feature Projections Classifier to Text Categorization. In Proceedings of the 13th International Symposium on Computer and Information Sciences – ISCIS'98, U. Gudukbay, T. Dayar, A. Gursoy, E. Gelenbe (eds), Antalya, Turkey (1998) 135-142