Analyzing the Temporal Sequences for Text Categorization

Xiao Luo, A. Nur Zincir-Heywood
Dalhousie University, Faculty of Computer Science, NS, Canada
luo@cs.dal.ca , zincir@cs.dal.ca

Abstract – *This paper describes a text categorization approach that is based on a combination of a newly designed text representation with a kNN classifier. The new text document representation explored here is based an unsupervised learning mechanism – a hierarchical structure of Self-Organizing Feature Maps. Through this architecture, a document can be encoded to a sequence of neurons and the corresponding distances to the neurons, while the temporal sequences of words as well as their frequencies are kept. Combining this representation with the power of kNN classifier achieved a good performance (Micro average F1-measure 0.855) on the experimental data set. It shows that this architecture can capture the characteristic temporal sequences of documents/categories which can be used for various text categorization and clustering tasks.*

1. Introduction

Text categorization is one of the significant tasks of content-based document management. Research has been performed in this area since early '60s; but it became a subfield of the information systems discipline [9] in the early '90s. Text representation is a necessary and important step for text categorization. The most popular technique for representing a text document is Vector Space Model (VSM). The basic VSM was introduced in 1975 by Salton et al [8]. In this model, a document is represented by a vector. The number of dimensions of the vector is the number of different words in the corpus. Each entry of the vector is indexed by a specific individual word, and the components of the vector are formed by a given weight of the term. However, many researches show that, in a linguistic sense, individual words could not be expressed as a textual unit because they have a larger degree of ambiguity than phrases [11]. Nevertheless, attempts to introduce more sophisticated text representation methods are not ceasing. These include selected n-grams representation [3], Natural Language Processing [1,6], Bag-Of-Words [4,7,11]. However, with the increasing size of the document corpus, each document usually includes only a small fraction of it. Either n-gram or VSM faces statistical sparseness and high dimensionality problems. Many researchers have generally relied on feature selection and generation to solve the problem. In some contexts, the use of appropriate feature selection could improve the performance. Hence, it is important to explore a wide range of levels of feature selection. However, this is more complex and not easy to use. Moreover, neither of the representations above considers the significant sequences of words or phrases in the documents. Especially, word sequences or position information is very important to a document when the document is fairly short and the words in each of the documents are very similar.

In this work, our objective is to explore a new way of representing a text document for text categorization by keeping information regarding the temporal sequences of words, as well as their frequencies. The new way of representation is based on hierarchical Self Organizing Feature Maps (SOMs) architecture. This architecture was employed to encode those pertinent features (character probabilities) of a document. Then the encoded information can be used to measure the similarity between the characteristics of any given document. Specifically, a hierarchical Self Organizing Feature Maps (SOMs) architecture is employed to encode the original information in a hierarchy by first considering the relationships between characters, then words, and finally word co-occurrences. After the SOMs training process, each document is represented by a sequence of *best matching unit*s (*BMU*s) on the third level SOM and the Euclidean distances to the corresponding neurons (*BMU*s*)*. To this end, the machine learning categorization algorithm *k*-Nearest Neighbour (kNN) is employed for the stage of categorization. The results show that this encoding system can capture the characteristic temporal sequences for documents and categories. The sequence information can be utilized for document categorization. The results turned out good on the returned Micro F1-measure (0.855).

This encoding mechanism has several advantages. First, this representation naturally solves the high dimensionality and statistic sparse problem, which occur with the conventional representation for the high volume corpus. Second, it implicitly considers word correlations and position information. Finally, this encoding mechanism can encode both textual and non-textual data. It can also be utilized in analyzing other data where sequence information is significant. From the categorization results, we conclude that this data representation mechanism can

be used efficiently for many information retrieval systems, especially business or medical information systems in which sequence information is very important.

The rest of the paper is organized as follows. Section 2 presents the hierarchical SOMs encoding system and the SOM learning algorithm. The document representation and the categorization method are described in Section 3. Section 4 gives the experiments performed and the results. Finally, conclusions are drawn and future work is discussed in section 5.

2. Hierarchical SOMs Encoding System and the Learning Algorithm

In this session, a three-level hierarchical SOM architecture for the process of encoding documents is described. Each of the three levels of the SOM hierarchy is employed to discover the patterns of characters, words, and word co-occurrences.

Indeed, pre-processing of data is employed before the encoding process. As usual, all the tags and non-textual data were removed from the original document. Then a simple part-of-speech (POS) tagging algorithm [2] is used to choose nouns from the document after which special nouns are removed. The steps of pre-processing are shown in Figure 2.

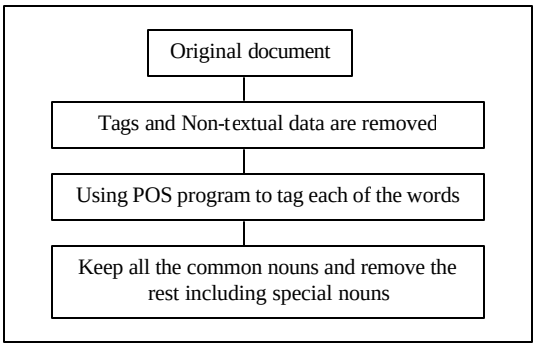

Figure 1: An overview of data pre-processing for the proposed approach

2.1 Encoding characters, words, and word co-occurrences - a three level hierarchical SOM architecture

The core of our approach is to automate the identification of typical category characteristics by analyzing the temporal sequence information of the documents in the corpus. As mentioned before, pattern discovery employs a three level hierarchical SOM architecture (characters, words, and word co-occurrences).

Each word in the documents is pre-processed to provide a numerical representation for each character. An SOM may now be used to identify a suitable character encoding, then word encoding, and finally, word co-occurrence encoding. By doing so, the authors of this paper aim to minimize the amount of *a priori* knowledge required to overcome the vocabulary differences. The hierarchical nature of the architecture is shown in Figure 2.

1) Input for the First-Level SOMs: In order to train an SOM to recognize patterns in characters, the document data must be formatted in such a way as to distinguish characters and highlight the relationships between them. Characters can easily be represented by their ASCII representations. However, for simplicity, we enumerated them by the numbers 1 to 26, i.e. no differentiation between upper and lower case. The relationships between characters are represented by a character's position, or time index, in a word. For example, in the word "*news*": "*n*" appears at time index 1, "*e*" appears at time index 2, "*w*" appears at time index 3, and "*s*" appears at time index 4. It should be noted that it is important to repeat these words as many times as they occur in the documents. In other words, for a particular document, the most frequently occurring words may occur a combined total of 10 times. Therefore, a list of 10 words is formed with the words remaining in the same order as they appear in the document. The overall pre-processing process for the first-level SOM is therefore:

- Convert the word's characters to numerical representations between 1 and 26.
- Give the time index to the characters in a word. It is the actual time index times 2.

The indices of the characters are altered in this way so that when the list is input to an SOM, both data features (enumerated characters and indices) are spread out over a close range. The assumption at this level is that the SOM forms a code-book for the patterns in characters that occur in a specific document category.

2) Input for the Second-Level SOMs: When a character and its index are run through a trained first-level SOM, the closest neurons (in the Euclidian sense), or *Best Matching Units* (*BMU*s), are used to represent the input space. The following inter-stage processing is therefore used for defining word level inputs to the second-level SOMs:

- For each word, *k*, that is input to the first-level SOM of each document,
  - Form a vector of size equal to the number of neurons (*r*) in the first-level SOM
  - For each character of *k*,
    - Observe which neurons $n_1, n_2, ..., n_r$ are affected the most (the first *3 BMU*s).
    - Increment entries in the vector corresponding to the first 3 *BMU*s by 1/j, 1 = j = 3.

Hence, each vector represents a word through the sum of its characters.

3) Input for the Third-Level SOMs: In the context of this architecture, word co-occurrence is simply a group of consecutive words in a document. Thus, the third-level SOMs are used to identify such patterns in the input space. The input space of the third-level SOMs is formed in a similar manner to that in the second-level, except that the third-level input vectors are built using *BMU*s resulting from word vectors passed through the second-level SOMs. Furthermore, in order to form a more meaningful input space for the third-level SOMs, it should be noted that the input vectors represent consecutive words only from a single document with a sliding window of size three. The result given by the hierarchical three level SOMs is: clusters of word co-occurrences on the third level SOM.

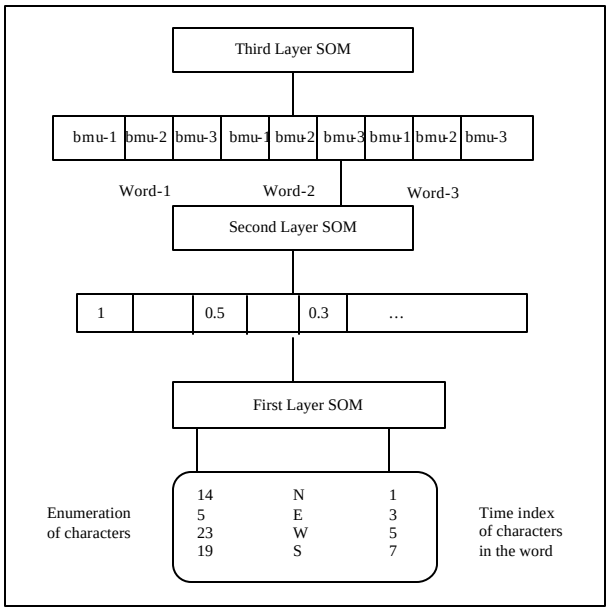

Figure 2: An overview of the hierarchical SOMs encoding architecture

2.2 Training SOM – the Learning Algorithm

The algorithm responsible for the formation of the SOM involves three basic steps after initialization: sampling, similarity matching, and updating. These three steps are repeated until formation of the feature map has completed [5]. The algorithm is summarized as follows:

- Initialization: Choose random values for the initial weight vectors $w_j(0)$, $j=1,2,\ldots,l$, where l is the number of neurons in the map.
- Sampling: Draw a sample x from the input space with a uniform probability.
- Similarity Matching: Find the best matching neuron $i(x)$ at time step n by using the minimum distance Euclidean criterion:

$$i(x) = \arg\min_j ||x(n) - w_j||, \qquad j=1,2,\ldots,l \qquad (1)$$

- Updating: Adjust the weight vectors of all neurons by using the update formula:

$$w_j(n+1) = w_j(n) + \eta(n) h_{j,i(x)}(n)(x(n) - w_j(n)) \qquad (2)$$

- Continuation: Continue with sampling until no noticeable changes in the feature map are observed.

The sizes of the maps shown in Table 1 are chosen empirically according to the observed weight changes of neurons on the SOMs. Hence, we considered the balance between the computational cost and the weight change in choosing the size of a map.

Table 1. Size of the maps

	Size of the map
Level-1	7 by13
Level-2	8 by 8
Level-3	20 by 20

3. Document Representation and Categorization Method

The text categorization scheme we explored is composed of two components: representation of documents by using the encoded information through the hierarchical SOMs described in section 2 and a kNN classifier-learning algorithm. In this section, we describe both components.

3.1 Document Representation

After training the three levels of SOMs, we analyzed the hit histogram and the *BMU* sequences on the third level SOM for categories and documents. We find that documents from the same category always hit the some same parts of the third level SOM, and also they have some common parts of the *BMU* sequences to each other. As an example, *BMU* sequences of two documents from category "Earn" are shown in Figure 3 and 4. Moreover, top frequent *BMU* sequences of different categories are analyzed. Those common *BMU* sequences between the documents (in Figure 3 and 4) belong to the top frequent *BMU* sequences of their corresponding category ("Earn") as shown in Figure 5. It also shows that different categories have different top frequent *BMU* sequences. Top frequent *BMU* sequences of category "Grain" is shown in Figure 6. Thus, it demonstrates that using this architecture to encode the data can capture the characteristic sequences for documents and categories. To this end, we hypothesize that the more temporal sequences they share the more similar the documents are. Based on the information, we proposed the document representation by using a sequence of *BMU*s on the third level SOM and distances to the corresponding *BMU*s as Figure 7.

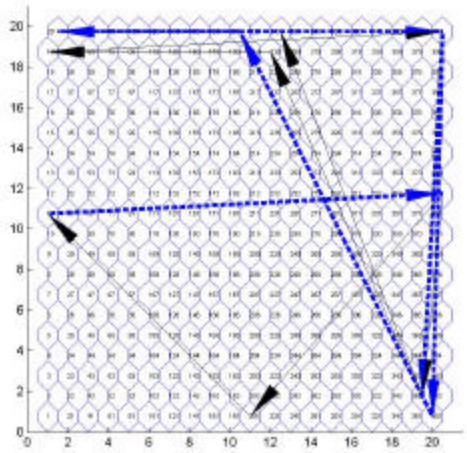

Figure 3. *BMU* sequences of a document (A) from category "Earn"

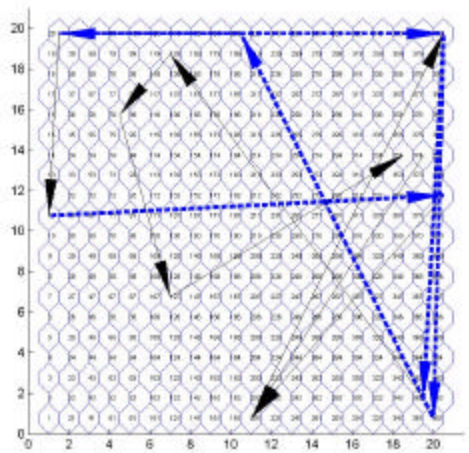

Figure 4. *BMU* sequences of a document (B) from category "Earn"

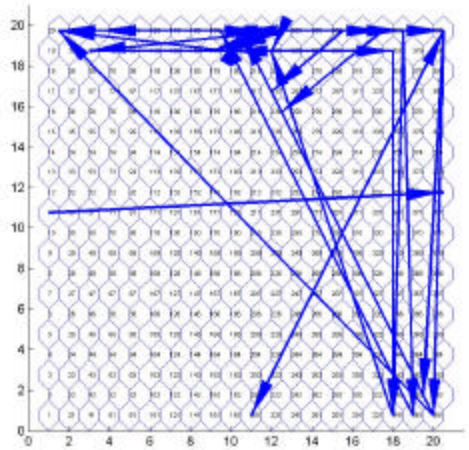

Figure 5. Top frequent *BMU* sequences of category "Earn" with frequency > 150

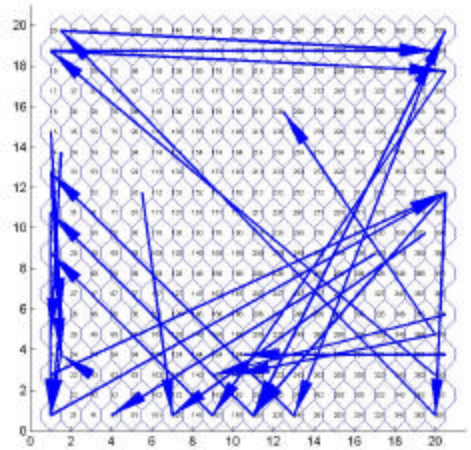

Figure 6. Top frequent *BMU* sequences of category "Grain" with frequency > 20

Index of the *BMU*s on the third level SOM (*B*)	392	201	11	392	362	239	19	400 ……
Euclidean distance to the corresponding *BMU* (*W*)	12.6	10.7	11.9	11.1	9.9	7.8	8.8	10.1 ……

Figure 7. Sequential representation of the document

3.2 k-Nearest Neighbour Classifier-Learning Algorithm

kNN stands for k-Nearest Neighbour (kNN) classification. It has been studied extensively for text categorization by Yang and Liu [10]. The kNN algorithm is quite simple: To classify a test document, the k-Nearest Neighbour classifier algorithm finds the k nearest neighbours among the training documents, and uses the category labels of the k nearest training documents to predict the category of the test document. The similarity score of each neighbour document to the test document is used as the weight of the categories of the neighbour document [10]. If there are several training documents in the k nearest neighbour that share a category, the category gets a higher weight and most likely the test document will to be classified to that category. In general, Euclidean distance, or cosine distance is used to measure the similarity between the documents. However, those distance calculations are for vector space representation. In this work, we designed a similarity measurement as (3), which fits to this sequential data representation.

$$Sim(D_i, D_j) = \sum_{k=1}^{n} \frac{100}{1 + dist(W_{ik}, W_{jk})} \times n \qquad (3)$$

D_i : A test document to be categorized.
D_j : A document in the training set.
n : The total number of *BMU*s in common *BMU* sequences of D_i and D_j.
$dist(W_{ik}, W_{jk})$: The Euclidean distance between the W (defined in Figure 7) of the corresponding *BMU* in the common *BMU* sequences of D_i and D_j.

4. Experimental Setup and Categorization Results

The practical experiments have been conducted by using the famous data set Reuters-21578 and combined with a kNN classifier described above. The description of the data set and the corresponding categorization results are given in the following subsections.

4.1 Experimental Data Set

In this work, we used the well-known multi-class, multi-labeled document set - Reuters-21578[1], which is a large research-oriented corpus to evaluate the approaches. There are a total of 12612 news stories in this collection. These stories are in English, where 9603 of them are in the training data set, and 3299 are in the test set. In our experiments,

[1] Reuters data set, http://www.daviddlewis.com/resources/testcollections/reuters21578/

we make use of all 9603 training files which belong to 118 categories. After pre-processing as Figure 1, they are input one by one to the SOMs encoding system. Finally, *BMU* sequence of the third level SOM is constructed to represent each of the documents.

In general, when a document has more than one label, those category labels are very similar or strongly related to each other. Thus, to automatically explore the relationship of those similar categories becomes a challenge for text categorization. In order to analyze the efficiency of the representation for the multi-class, multi-labeled document categorization, we analyzed complex relationships and overlap between the top 10 categories of this data set. Based on the information we got, we chose 6 of them to test the categorization performance. These categories are "Earn", "Money-fx", "Interest", "Grain", "Wheat" and "Corn". The size and relationship between these categories in training set are shown in figure 7, and is the same as their relationship in the test set. We could see that "Grain", "Wheat" and "Corn" are strongly related to each other, as are "Money-fx" and "Interest". "Earn", the biggest category in the whole corpus, has no overlap with the other five categories. In total, there are 1503 multi-labeled and uni-labeled documents in the test set.

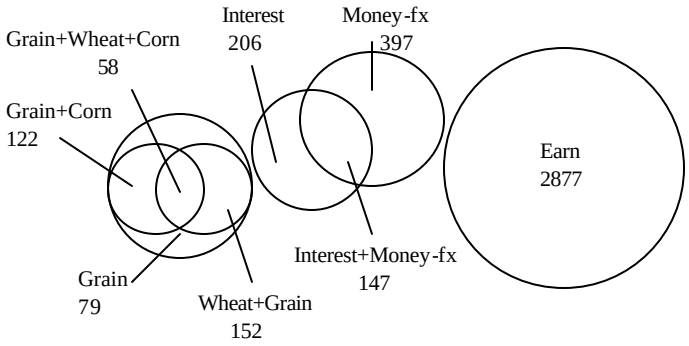

Figure 7. Size and Relationship of the Six Categories in Training Set

4.2 Categorization Results and Discussion

In our experiments, we set the number of nearest neighbors - k = 3, 5, 10 and 15. It turns out by experiments that k = 5 gives the best performance in terms of the Micro F1-measure score.

Facing a multi-labeled (*N* labels) test document, we first calculate the similarity between the test document and the training documents in the selected categories using formula (3). After ranking those similarities, we select 5 of them and weight the categories they belong to. Finally, we classify the test document to the top *N* weighted categories which corresponding to the number of the labels of the test document.

The classical effectiveness measurements of multi-labelled text categorization: Recall(R), Precision (P) and F-measure (F) are used to measure the categorization performance.

$$R = \frac{TP}{TP + FN} \quad (4) \qquad P = \frac{TP}{TP + FP} \quad (5)$$

TP : Positive examples, classified to be positive.
FN : Positive examples, classified to be negative.
FP: Negative examples, classified to be positive.

$$F1 - measure = \frac{2RP}{R + P} \quad (6)$$

Table 2-3 summary the results obtained through the experiments.

Table 2. Categorization results of all six categories

Category	Size in the test set	Recall	Precision	F1-measure
Earn	1087	0.958	0.956	0.957
Money-fx	179	0.646	0.662	0.654
Interest	131	0.539	0.738	0.623
Grain	149	0.741	0.708	0.724
Wheat	71	0.721	0.671	0.695
Corn	56	0.623	0.702	0.660
Micro Average F1-measure:		**0.855**		

Table 3. Categorization results of multi-labelled documents

Category	Size in the test set	Recall	Precision	F1-measure
Money-fx+Interest	43	0.658	0.892	0.810
Grain+Corn	34	0.500	0.708	0.586
Grain+Wheat	49	0.776	0.809	0.792
Grain+Corn+Wheat	22	0.579	1	0.733

From the achieved performance above, this encoding architecture capture the characteristic sequences for documents and categories. Good performance is achieved by utilizing the sequences information for categorization. It also offers a good trade-off between recall and precision. However, the results show that it works better for some categories, especially the category "Earn". We conclude the reasons behind this are: first, this data representation is based on the machine-learning algorithm to capture the characteristic word co-occurrence for categories, so the more frequent the word co-occurrence is, the more easily it can be caught and represented by the neurons of the third level SOM. In the corpus, category "Earn" is the biggest category; there are a large number of word co-occurrences from this category. This results that enough neurons are well trained to represent the characteristic word co-occurrences. Second, after looking into each of the documents in "Earn", we found that most of the documents are fairly short. There is less variety of word co-occurrences in the category "Earn". All of them can be captured and represented well by the SOM neurons. On the other hand, for some categories with more variety of word co-occurrences, because of the size of SOM, some word co-occurrences of them may not be represented well enough by the neurons on the SOM. The characteristic word co-occurrences from different categories may mix together on the same neuron. This impresses the performance for those categories.

From the analysis above, it is noted that the size of the SOMs has an important effect on the final performance of the categorization. The bigger the SOM's size, the more characteristic word co-occurrences are captured. In this work, in order to show that this architecture works well on analysis of temporal sequences for text categorization, we selected the proper size of SOM once it converges. Other sizes of SOMs will be tested and more sophisticated method to determine the size of a SOM will be investigated in the future.

5. Conclusion and Future Work

Through this work, we explored a new way of data representation specifically designed for text representation. The results (in section 4) show that this architecture works well for capturing the characteristics of documents/categories using temporal sequences of word co-occurrences. The performance of this new data representation has been tested for document categorization by using a kNN classifier on top of it. We end up with the Micro average F1-measure for the selected data set at 0.855.

This sequential data representation naturally solves the high dimensionality (sparseness) problems of the VSM for the large data sets. Moreover, since the order and the frequency of words or phrases are maintained as they appear before inputting to the encoding architecture, the hierarchy of SOMs can automatically capture the significant characteristics of the data without additional feature selection techniques.

The efficiency of this new representation presented in this paper is still far from being completely elaborated, so we definitely consider this as a work in progress. Future work will include performing experiments on utilizing the temporal sequences analysis for extensive document categorization as well as classification of data for other applications such as medical and/or business information systems in which the analysis of temporal sequences of information is very important. And other classifiers which fit more to the sequences representation will also be analyzed and utilized in the future.

References

[1] R. Basili, A. Moschitti, and M. T. Pazienza. Language-sensitive text classification. In Proceedings of 6th International Conference "Recherche d'Information Assistee par Orinateur", 331-343, 2000.

[2] E. Brill. A simple rule-based part of speech tagger. In Proceedings of 3rd Conference on Applied Natural Language Processing, 152-155, 1992.

[3] M.F. Caropreso, S. Matwin, and F. Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. Text Databases and Document Management: Theory and Practice, 78-102, 2001.
[4] S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In Proceedings of 7th ACM International Cconference on Information and knowledge management, 148-155, 1998.
[5] S. Haykin. Neural Networks - A comprehensive foundation, Chapter-9: Self-organizing maps, Second Edition, Prentice Hall, 1999, ISBN 0-13-273350-1.

[6] P.S. Jacobs. Joining statistics with NLP for text categorization. In Proceedings of the Third conference on Applied Natural Language Processing, 178-185, 1992.
[7] T. Joachims. Text Categorization with support vector machines: learning with many relevant features. In Proceedings of ECML'98, 10th European Conference on Machine Learning, 137-142, 1998.
[8] G. Salton, A. Wang, and C.S. Yang. A vector space model for information retrieval. Journal of the American Society for information Science, 18, 613-620, 1975.
[9] F. Sebastiani. Machine learning in automated text categorization. ACM Computing Surveys, 34(1), pp. 1-47, 2002.
[10] Y. Yang, X, Liu. A re-examination of text categorization methods. In Proceedings of SIGIR'99, 42-49 1999
[11] S. M. Weiss, C. Apte, F. J. Damerau, D. E. Johnson, F. J. Oles, T. Goetz, and T. Hampp. Maximizing text-mining performance. IEEE Intelligent Systems, 14(4):63-69, 1999.