Title: Text Categorization for an Online Tendering System

Authors: Y. Wang, H. Zhang, B. Spencer, Y. Yan

Y. Wang, H. Zhang
Faculty of Computer Science
University of New Brunswick
P.O.Box 4400, Fredericton
NB, Canada
yue.wang@nrc.ca, hzhang@unb.ca

B. Spencer, Y. Yan
Institute for Information Technology
National Research Council of Canada
46 Dineen Drive, Fredericton
NB, E3B 9W4, Canada
{bruce.spencer,yuhong.yan}@nrc.ca

Abstract: This paper investigates the application of text categorization (TC) in a setting exhibiting a large number of target categories with relatively few training cases, applied to a real-life online tendering system. This is an experiment paper showing our experiences in dealing with a real-life application using the conventional machine learning approaches for TC, namely, the Rocchio method, TF-IDF (term frequency-inverse document frequency), WIDF (weighted inverse document frequency), and naïve Bayes. In order to make the categorization results acceptable for industrial use, we made use of the hierarchical structure of the target categories and investigated the semi-automated ranking categorization.

Keywords: text categorization, machine learning, the Rocchio method, TF-IDF, WIDF, weighted inverse document frequency, naïve Bayes classifier, ranking categorization.

Email address of contact author: yue.wang@nrc.ca

Phone number of contact author: (506)444-0485

1 Introduction

Knowledge organized hierarchically in ontologies is central to the design of the Semantic Web. Much Semantic Web research is focussed on what can be done with such organized knowledge. However, there are also interesting problems surrounding the organizing of real world items into ontologies. This paper outlines an eBusiness application, an online tendering system, which has an ontology of product types at its core.

We first give the definitions of a few tendering terminologies that are relevant to our discussion. A *vendor* is a unit who sells goods or services. A *purchasing agent* is a buyer of any goods or service. A *tender* is a document that a purchasing agent publishes to announce his request for certain goods or services. The Goods and Services Identification Numbers (GSIN) is a hierarchically arranged set of codes that can classify a huge range of products. This GSIN ontology is used in our application as a means to classify both tenders and vendors in terms of what they buy and sell respectively. It is via this ontology that effective match making between vendors and tenders is achieved. Currently, the incoming tenders are assigned a GSIN code by a human expert familiar with the GSIN system. We have completed a pilot study in automatically assigning a GSIN code to the tenders using previously coded tenders as training data. We have taken a few conventional machine learning approaches for TC in order to better understand our categorization problem. Our initial results are encouraging, especially for top-level categories and given that we have the option of presenting a ranked list of categories to a human indexer.

This paper is organized as follows. In Section 2 we give an introduction to the New Brunswick Opportunities Network (NBON), the target tendering system of this paper. In Section 3 we examine four TC techniques, namely, the Rocchio method, TFIDF (term frequency-inverse document frequency), WIDF (weighted inverse document frequency), and naïve Bayes. In Section 4 we show the experiment results on the NBON data using the four classifiers. In Sections 5 we discuss future and related work, conclude in Section 6.

2 The NBON System

New Brunswick Opportunities Network [15] is the official online tendering system of the province of New Brunswick, which went live in May 2002. The motivation of the system is to help vendors and purchasing agents find each

N66	Instruments and Laboratory Equipment
N6665	Hazard-Detecting Instruments and Apparatus
N6665A	Detectors, Hazard
N6665AB	Detectors, Hazard, Concealed Weapon

Table 1: An example of GSIN codes and categories

other online through GSIN codes. The NBON system has a hierarchical structure for its categories and their corresponding GSIN codes. It consists of three super-level groups: goods, services, and construction services. Under the super-level groups, there are currently 15,913 categories grouped in up to four levels, each corresponding to a unique GSIN code. Table 1 shows a typical example of a group of goods categories on the same path of the hierarchical structure and their corresponding GSIN codes.

The whole tendering and bidding procedure is similar to that of a traditional one, i.e. a purchasing agent submits a tender to the government (now represented by the NBON system), which notifies the potential vendors, who can choose to bid. Then the purchasing agent chooses and announces the successful bidder. A vendor needs to electronically register with the NBON system to be recognised as a potential bidder. For this purpose, he needs to navigate the hierarchical structure of the categories of the NBON system to identify a category that properly describes his expertise. A vendor may choose to classify himself on any level of the structure, except for the super-level. A purchasing agent submits to the government a tender, which is then assigned a GSIN code by a human indexer, either the purchasing agent himself or an expert in the government. If there is more than one item in the tender, each item will be treated as a separate request, and is given a separate GSIN code.

Since we do not want to leave out any potential vendor, the notification strategy of the NBON system is to notify all the registered vendors that share a branch with the tender on the tree of GSIN codes, meaning the parent, children and siblings of the category node under which the tender is classified. The main reason behind such a strategy is that the manual categorization of the NBON system is not always reliable, therefore the system has to retrieve more vendors to be on the safe side. Recall that the vendors are self-classified, and since there are altogether 15,913 categories in the NBON system, the vendors may self-classify too high in the hierarchical structure, and thus the categorization result is less precise. The same thing happens to the human indexers, who, although more familiar with the categories, may not read the tender carefully or thoroughly enough to give each item the correct GSIN code.

The main problem with the current NBON system is the manual categorization of the tenders and the vendors, which both decreases the usability of the registration procedure of the system and increases the unreliability of notification. The motivation of this paper is to make the categorization process (semi-)automated using a machine learning approach, i.e. to learn from the historic data of the NBON system and use the learned knowledge to classify a new document. We start with the categorization of the tenders, because we can easily obtain the training data necessary for the learning approach that we took in this paper: the tenders and their manually assigned categories from the past years are suitable training examples that we can feed to a computer program to build a classifier. In this paper, only the goods tenders are selected as the sample data. The goods group is more representative than the other two in terms of both overall quantity (there are 14,783 goods categories out of the total 15,913 categories) and number of levels. Therefore we chose the goods group for our initial investigation.

3 Text categorization of the tenders

The NBON sample data used in this paper consists of the goods tenders published in the system during the two years from 1 January 2001 to 30 November 2002, which was the most up-to-date data that we could obtain during the production of this paper. In the data set, there are altogether 4,822 goods tenders classified under 1,269 categories. The biggest tender contains 826 words, and the smallest only 1 word. The average number of words per tender is 31. Recall that a tender may have multiple items and thus is assigned multiple GSIN codes, one for each item. Having realized the potential mistakes in human indexing, the government only provided us with the first items of the multi-item tenders, which have a better chance to be correct, so as to improve the quality of the data. This means that we are dealing with a single-label TC problem in this paper, because a tender is assigned exactly one GSIN code.

Table 2 reflects some statistical facts about the NBON sample data after preprocessing. We use $|Categories|$ to denote the number of categories, $|Vocabulary|$ to denote the number of distinct words , $|c|$ to denote the num-

	Schema I	Schema II
Total $\|Categories\|$ in NBON system	14,783	72
$\|Categories\|$ in the sample data	1,269	70
$\|Vocabulary\|$	12,266	12,266
average $\|c\|$	72	1,318
Total $\|Tenders\|$	4,822	4,822
$F(c) \geq 200$	1/337	4/1,945
$F(c) \geq 100$	4/697	13/3,123
$F(c) \geq 50$	8/1,006	23/3,743
$F(c) \geq 10$	70/2,107	56/4,763
$F(c) < 10$	1,199/2,715	14/59
$F(c) \leq 3$	984/1,554	7/12
$F(c) = 1$	552/552	4/4

Table 2: Statistics of the NBON sample data

ber of words in a category c, $|Tenders|$ to denote the number of tenders, and $F(c)$ to denote the frequency of the category c, which equals the number of tenders classified under c. The pairs in the $F(c) \geq x$ rows represent the number of categories c that meet such criteria and the total number of tenders classified under these categories. We define two category schemas, *Schema I* denotes the categories at all levels as in a flat structure, and *Schema II* denotes the top-level categories in the hierarchical structure. Take *Schema I* for example, we see in Table 2 that there are 14,783 goods categories in the NBON system, in which 1,269 ones were observed in the sample data. There are 12,266 (preprocessed) distinct words in the data set, and the average number of distinct words per category is 72. There are 4,822 goods tenders in the data set, in which 1 category has no less than 200 tenders manually classified under it, and the actual number of tenders is 337; 4 categories have no less than 100 tenders, and the total number of tenders is 697; etc. In the last three rows of Table 2, we see that 1,199 categories have less than 10 tenders, 984 categories have less than 3 tenders, and 552 categories have exactly 1 tender.

We can identify three data sparseness problems in the sample NBON data in *Schema I*. Firstly, the number of categories is large, which is 1,269. It is intuitively obvious that the difficulty of a TC problem increases with the number of categories. Secondly, the number of the "sparse" categories (the ones with few tenders) is proportionally large. Such “sparse” categories are those in the last three rows of the table, which is $\frac{1199}{1269} = 94.48\%$. And finally, the proportion of the categories that are not covered in the sample data is large. Out of all the 14,783 goods categories of the NBON system, only 1,269 are covered by the sample data, which means the coverage of the sample data in terms of categories is only 8.58%. This will be a problem when the system under development is used in real life, because there is a great chance that it will encounter some categories that are not covered by the “learned knowledge” of the classifier.

Since the categories are organized in a hierarchical structure, we can classify a tender in a hierarchical manner, i.e. level by level. By doing so, we address the three above mentioned problems. First of all, the number of categories on each level is dramatically less than that of the total number of categories. Take the top-level for an example, which is *Schema II* in Table 2, there are only 72 categories in total, which is far more manageable than 14,782 as in *Schema I.* Second, the number of tenders in most categories is increased, because each category includes the tenders classified under their children categories. Third, the number of uncovered categories is dramatically decreased. The coverage rate is $\frac{70}{72} = 97.22\%$ in *Schema II.*

As an initial investigation, our efforts in this paper were focused on the top-level categories. For now, such an approach is fine-grained enough for the TC task of the NBON system, whose notification strategy makes it sufficient to classify a tender high in the category hierarchy. It would be meaningless to blindly go further down the hierarchical category structure before we have a high confidence in the categorization results achieved on the higher level of the structure, because the categorization mistakes made on higher levels can not be recovered on lower levels [5], and thus the predicted category ends up being further away from the correct one on the tree of GSIN codes as the classifier goes lower on the hierarchical levels.

In this paper, we investigate four classifiers, namely, the Rocchio method, TFIDF, WIDF, and naïve Bayes.

3.1 The Rocchio method

The Rocchio method originated from the relevance feedback of information retrieval (IR), and was first adapted to TC in 1994. It has since been used by many researchers as a baseline classifier [19]. We also use the Rocchio method as a baseline algorithm in this paper.

The Rocchio method is a linear classifier consisting of a “profile” vector for each category in the form $\vec{c} = (R_1, ..., R_{|c|})$, with each R_i representing the weight of a word w in c as defined in Equation (1).

$$Rocchio(w, c) = \beta \sum_{d_i \in POS_c} \frac{TF(w, d_i)}{|POS_c|} - \gamma \sum_{d_i \in NEG_c} \frac{TF(w, d_i)}{|NEG_c|}, \tag{1}$$

where $TF(w, d_i)$ is the frequency of w in document d_i, $|POS_c|$ is the number of documents with label c, and $|NEG_c|$ is the number of documents with labels other than c. In this paper, we followed [4] to set the weight parameters to $\beta = 1$ and $\gamma = 0$ [1], meaning that the function is simply a weight averaged over the positive documents.

In TC, a document is usually represented as a term vector $\vec{d} = (d_{w_1}, ..., d_{w_{|d|}})$, with terms typically identified by the words in the document, and d_{w_i} denotes the frequency of each distinct word in d. When classifying a document d, which is a tender in our case, similarity is compared between the vector $\vec{d}$ and the profile vector $\vec{c}$ of each category c, using a linear similarity function $Sim(d, c)$ (see Equation (2)), which is the dot product of the two vectors [12]. And the classifier assigns d to “the most similar” category.

$$\begin{aligned} Sim(d, c) &= \vec{d} \cdot \vec{c} \\ &= \sum_{i=1}^{|d|} d_i c_i \\ &= \sum_{w \in d} d_w Rocchio(w, c) \end{aligned} \tag{2}$$

3.2 TF-IDF classifier

Term frequency-inverse document frequency (TF-IDF) is a traditional algorithm originally used in IR that measures the relevance by the product of TF and IDF. The weight of a word in a document is jointly determined by its frequency in the document and its frequency in all the documents. The IDF term can be viewed as a penalty to the ubiquitous words that do not tell much about any document.

The TF-IDF weight of a word w in a document d is given in Equation (3).

[1]We also followed other researchers to deemphasize the role of negative documents by setting $\beta = 16$ and $\gamma = 4$ [2, 7, 8], and the categorization results were worsened. Variations of β and γ values were tried out on the NBON sample data, and the best results were achieved by simply ignoring the negative documents.

	d_1	d_2	d_3	d_4	d_5
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
w_x	2	50	3	2	4
w_y	3	2	3	2	3
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 3: A TF-IDF example with a 5-document data set [20]

$$\begin{aligned} TFIDF(w,d) &= TF(w,d)IDF(w) \\ &= TF(w,d)log(\frac{|D|}{DF(w)}), \end{aligned} \quad (3)$$

where $TF(w,d)$ denotes the frequency of w in d, $|D|$ denotes the total number of the training documents, and $DF(w)$ is the number of documents that contain w.

When adapted to TC, the IDF terms are typically computed on categories, which are represented as *bags of documents* that are labelled with the same category [20]. The categorization criterion is similar to that of the Rocchio method in Equation 2, with *Rocchio*(w,c) being replaced by $TFIDF(w,c)$.

3.3 WIDF classifier

Tokunaga and Iwayama [20] pointed out a problem with $IDF(w)$, which is that if a word is contained in all the documents, no matter what its frequency distribution is like, the weight of such a word in all the documents will be zero, because the $IDF(w)$ value is zero. The improperness of such an approach is illustrated in the example shown in Table 3 [20], where both $IDF(w_x)$ and $IDF(w_y)$ are equal to 0, because they both occur in all the documents, and thus both words will get a zero weight for all the documents, although the frequency distributions of w_x and w_y are very different. The reason for this situation is that IDF considers only whether or not a word is contained in a document, but not its frequency in the document.

To address this problem, Tokunaga and Iwayama developed a weighted inverse document frequency (WIDF) classifier which is defined as in Equation (4). Since WIDF has already taken $TF(w,d)$ into account, the weight of a word w in a document d is simply its $WIDF$ value.

$$WIDF(w,d) = \frac{TF(w,d)}{\sum_{i=1}^{n} TF(w,d_i)} \tag{4}$$

Now, the $WIDF$ values of the two words in the above example would be different. For instance, according to Table 3,

$$WIDF(w_x,d_2) = \frac{50}{2+50+3+2+4} = \frac{50}{61} \approx 0.82, \tag{5}$$

and

$$WIDF(w_y,d_2) = \frac{2}{3+2+3+2+3} = \frac{2}{13} \approx 0.15. \tag{6}$$

For categorization, new documents are compared with the category representations using the linear similarity function in Equation (2), with $Rocchio(w,c)$ being replaced by $WIDF(w,c)$.

3.4 Naïve Bayes classifier

Naïve Bayes is a popular technique for TC tasks owing to its simplicity and effectiveness [14]. It is a probabilistic classifier, in which the relevance of a category c to a document d is based on its conditional probability given d, and d is assigned to "the most probable" category based on this probability value.

The probability of a category c given a document d is computed according to Bayes rule as in Equation (7).

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)} \tag{7}$$

If the words in d are mutually independent, Equation (7) can be rewritten as Equation (8), where the $P(d)$ term is dropped because it is a constant.

$$P(c|d) \propto P(c) \prod_{w \in d} P(w|c) \tag{8}$$

Naïve Bayes assumes such independent relations between words in a text (and thus the name "naïve"). Although this assumption is rarely true, naïve Bayes performs surprisingly well for TC tasks [14].

In Equation (8), $P(c)$ can be easily obtained by the proportion of the number of tenders assigned category c divided by the total number of tenders, and we compute $P(w|c)$ terms as in Equation (9), which is Lidstone's law of succession [1].

$$P(w|c) = \frac{F(w,c) + \lambda}{|c| + \lambda|Vocabulary|}, \tag{9}$$

where $F(w,c)$ denotes the frequency of word w in category c, and λ is a variable, whose value was set to be 0.1 in this paper, because it was optimal for the NBON sample data by experimentation. The reason of introducing a λ variable is to smooth out the *zero counts* (i.e. $F(w,c) = 0$). Unlike the previous three algorithms which sum up the weights, naïve Bayes multiplies $P(w|c)$ terms, and thus a single zero count will cause $P(w|c)$ to be zero, and so is $P(c|d)$ in turn, if no smoothing is applied. There are two types of zero counts, one is that w was not observed in the category c, and this case is cared by Lidstone's law of succession; the other is that w was not observed in any category in the training data, this is also referred to as a *missing value* or *unknown*, and we follow [9] to simply ignore the missing values in this paper.

4 Experiments on the NBON data

4.1 Objective

In this section, we show the categorization results of the four classifiers on the NBON data. Our objective is to achieve a result that is acceptable for industry uses.

4.2 Method

The tenders were preprocessed using a stop word list from the Defense Virtual Library [6] consisting of 456 words and the Porter's stemming algorithm [17]. We also tried using smaller stop word lists and a restrictive version of Porter's stemmer, and the differences on the categorization results are insignificant.

Precision and recall are two standard measures of effectiveness in IR. Precision is the proportion of the retrieved documents that are relevant, and recall is the proportion of the relevant documents that are retrieved. Precision and recall were adapted to TC, and their computations are based

	Yes is correct	No is correct
Assigned Yes	TP	FP
Assigned No	FN	TN

Table 4: A 2×2 contingency table

on a *contingency table* (see Table 4) that reflects the categorization decisions made by a classifier and the expert judgements on the decisions. The categorization results are denoted by TP (number of *true positive*), FP (number of *false positive*), TN (number of *true negative*) and FN (number of *false negative*).

Precision and recall are defined in Equations (10) and (11) respectively. We took the *microaveraging* approach in [19] for the computation, which sums over the individual TP, FP and FN counts of each category to obtain a global measure. Precision and recall are two conflicting measures, therefore classifiers are usually evaluated by a combination of the two. An exception is the single-label TC, in which precision and recall are dependent on each other, therefore either can be used alone for evaluation [19].

In this paper, we use recall to measure the effectiveness of the classifiers. This naturally follows our discussion in Section 2 that leaving out a potential vendor is much more serious than a false notification of a tender, i.e. FN is less desirous than FP, meaning that a high recall is more important to the system. Precision and recall are equivalent at this stage, because as a single-label problem, each tender is assigned by human indexers to exactly one category, which the classifier tries to retrieve. If the classifier succeeds in retrieving the correct one, both precision and recall shall be 1; and 0 otherwise. However, precision and recall differ in a ranking categorization system that is to be discussed soon, therefore we use recall here to keep the consistency in later comparisons.

$$precision = \frac{TP}{TP + FP}, \tag{10}$$

$$recall = \frac{TP}{TP + FN}. \tag{11}$$

We used 20-fold cross-validation to train and test the classifiers. The 4,822 tenders in the NBON sample data were partitioned into 20 subsets of equal length, with 19 subsets served as the training data and 1 subset

	Rocchio	TF-IDF	WIDF	Naïve Bayes
Schema I	20.27%	22.07%	32.70%	34.59%
Schema II	43.46%	46.70%	59.17%	63.24%

Table 5: Recalls of the four classifiers on the two NBON category schemas

was left out for testing. The system ran 20 times so that each subset got a chance to be tested, and the final recall was averaged over the 20 results.

4.3 Results

Table 5 shows the categorization results obtained on the two category schemas of NBON. We can see that, as we expected, the WIDF and naïve Bayes classifiers achieved the higher recalls, and a significant improvement of *Schema II* (the top-level categories) over *Schema I* (categories on all levels in a flat structure) is accomplished. Recall from Section 3 that *Schema I* suffers serious sparseness problems, and by working with *Schema II*, we circumvented these problems. Whereas a recall of 30 percentage points on *Schema I* is of no use, that of 60 percentage points on *Schema II* is close to a reasonable result for the research purpose. Although the categorization results on *Schema II* are less precise than those on *Schema I*, according to our discussions in Sections 2 and 3, this is fine-grained enough because the notification strategy of the NBON system makes it sufficient to classify a tender high in the hierarchy.

Unfortunately, a recall of 60 percentage points is still unacceptable for industrial use. Therefore we developed an interactive ranking system to return the N best categories ranked high by the classifiers, and allow the users to choose the target out of them. Table 6 shows the improvement on *Schema II* achieved by returning the "N-best" categories. We see that at the level of 10-best, the recall of each classifier is increased from 20 to 37 percentage points. We stopped our experiments at the level of 10-best because some studies of searching behavior have shown that users tend to give up if a target is not found in the first ten hits [22].

	Rocchio	TF-IDF	WIDF	Naïve Bayes
1 Best	43.46%	46.70%	59.17%	63.24%
3 Best	64.83%	64.79%	71.91%	73.84%
5 Best	71.93%	72.93%	78.20%	78.15%
10 Best	80.50%	82.32%	84.83%	83.88%

Table 6: Recalls of the four classifiers on *Schema II* for the ranking categorization

4.4 Discussion

It certainly would be preferable to have a fully automated categorization procedure, but when the "1-best" prediction of a classifier is not very reliable, such automation is fairly meaningless. In such a case, it pays off to put the final decision in the hands of a human user.

In our case, this will also give users a chance to classify their tenders under multiple categories, which is desirable in some cases, such as when they have various requests in one tender, and/or when they want to know about other potentially relevant categorizations for their tenders. A major advantage of involving users in the categorization procedure is to increase the recall on the top-level and thus allow the finer grained categorization in future phases.

5 Related work

The recalls achieved in this paper were based on the top-level categories. Although we have assumed that it is sufficient to classify tenders high in the hierarchy, it is preferable to classify both tenders and vendors as precisely as possible. In addition, by classifying tenders and vendors low in the hierarchy, we will have more flexibility in modifying the notification strategy when necessary. In the next step, we will investigate the hierarchical categorization on all levels.

Classifying documents into a hierarchical structure of categories have been previously studied [3, 5, 10, 11, 13, 16, 18, 21]. Most research results showed limited improvement of hierarchical categorizations over the flat ones, except the work done by McCallum et al. [13], who showed a significant improvement of 29%. They used a naïve Bayes classifier, and adopted the statistical *shrinkage* technique to smooth the parameter estimates of sparse children categories with their parents. Dumais and Chen's work [5] was also interesting in the sense that they have shown that higher-level categorization mistakes cannot be recovered on lower levels by simply combining the estimates of the parent and the child category nodes. What we have learned from the previous work is that, in order to benefit from the hierarchical structure of categories, intrinsic relations between categories on the same branch of the hierarchy that reflect the distribution of data must be discovered and reflected in the computations, instead of doing the computations as though the categories are independent of one another and then simply combining the results of the parent and the child categories.

6 Conclusion and future work

New Brunswick's online tendering system organizes tenders into a set of categories arranged in a hierarchy of goods and services. Each category has a unique GSIN code. Whenever a new tender is open for bidding, a code is selected for it. Vendors select codes of interest, and they are notified electronically only if their selected code shares a branch with the code selected for the new tender. Thus selecting appropriate codes is vital to both the purchasing agent and the vendor. Our TC system (semi-)automates code selection for tenders, using historical data.

This work is significant for its effect in the application domain. Currently the live NBON system does not incorporate a text categorizer, and so end-users need to select from among the 15,913 different categories. While experienced users can select codes quickly, non-experts including members of the general public, must navigate the hierarchy to select an appropriate code. If text categorization methods can reduce this to a selection from among 10 choices, much human time can be saved every day.

Our work is ongoing, but so far we have found that the naïve Bayes approach applied to text categorization is appropriate for this task. We plan to do more experiments with Bayesian networks, especially when it is applied to hierarchical TC. Besides describing our experiences with a real world application, this paper also serves to bring together ideas from different spheres: naïve Bayes applied to the Semantic Web, and naïve Bayes applied to e-business applications. We also plan to investigate other TC techniques, such as k nearest neighbors and support vector machines.

In this paper, we were working with goods tenders only. In the next phase, we plan to expand the applicability of the current work to services as well as goods, and to vendors as well as tenders.

References

[1] Agrawal, R., R. Bayardo and R. Srikant. 1999. *Athena: Mining-based Interactive Management of Text Databases.* Research Report RJ10753, IBM Almaden Research Center, San Jose, CA95120.

[2] Cohen, W.W. and Y. Singer. 1996. Context-sensitive learning methods for text categorization. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval.*

[3] D'Alessio, S., M. Murray, R. Schiaffino and A. Kershenbaum. 1998. Category Levels in Hierarchical Text Categorization. In *Proceedings of EMNLP-3, 3rd Conference on Empirical Methods in Natural Language Processing.*

[4] Dumais, S., J. Platt, D. Heckerman and M. Sahami. 1998. Inductive Learning Algorithms and Representations for Text Categorization. In *Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management*, 148-155.

[5] Dumais, S. and H. Chen. 2000. Hierarchical Classification of Web Content, In *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, Athens, Greece, 256-263.

[6] Defense Virtual Library / Verity Stop Word List. http://dvl.dtic.mil/stop_list.html

[7] Ittner, D.J., D.D. Lewis and D.D. Ahn. 1995. Text categorization of Low Quality Images. In *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, NV, 301-315.

[8] Joachims, T. 1997. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In *Proceedings of ICML-97, 14th International Conference on Machine learning*, Nashvile, TN, 143-151.

[9] Kohavi, R., B. Becker and D. Sommerfield. 1997. Improving Simple Bayes. In *Proceedings of the European Conference in Machine Learning.*

[10] Koller, D. and M. Sahami. 1997. Hierarchically Classifying Documents Using Very Few Words. In *Proceedings of the 14th International Conference on Machine Learning*, Nashville, TN, 170-178.

[11] Larkey, L. 1998. Some Issues in the Automatic Classification of U.S. Patents. In *Working Notes for the AAAI-98 Workshop on Learning for Text Categorization.*

[12] Lewis, D.D., R.E. Schapire, J.P. Callan and R. Papka. 1996. Training Algorithms for Linear Text Classifiers. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*, Zürich, Switzerland, 298-306.

[13] McCallum, A., R. Rosenfeld, T. Mitchell and A.Y. Ng. 1998. Improving Text Classification by Shrinkage in a Hiearchy of Classes. In *Proceedings 15th International Conference on Machine Learning*, San Francisco, CA, 359-367.

[14] Mitchell, T.M. 1996. *Machine Learning.* The McGraw-Hill Companies, Inc., New York.

[15] New Brunswick Opportunities Network. https://nbon-rpanb.gnb.ca

[16] Ng, H.T., W.B. Goh and K.L. Low. 1997. Feature Selection, Perceptron Learning and a Usability Case Study for Text Categorization. In *Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information Retrieval*, Philadelphia, PA, 67-73.

[17] Porter, M.F. 1980. The Porter Stemming Algorithm.
http://www.tartarus.org/∼martin/PorterStemmer

[18] Ruiz, M.E. and P. Srinivasan. 1999. Hierarchical Neural Networks for Text Categorization. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, Berkeley, CA, 281-282.

[19] Sebastiani, F. 2002. Machine Learning in Automated Text Categorization. ACM Computing Surveys, Vol.34, No.1, March 2002, pp.1-47.

[20] Tokunaga, T. and M. Iwayama. 1994. *Text Categorization Based on Weighted Inverse Document Frequency.* Technical Report 0918-2802, Department of Computer Science, Tokyo Institute of Technology, Tokyo, Japan

[21] Weigend, A.S., E.D. Wiener and J.O. Pedersen. 1999. Exploiting Hierarchy in Text Categorization. *Information Retrieval, l(3)*, 193-216.

[22] Woods, W.A., L.A. Bookman, A. Houston, R.J. Kuhns, P. Martin and S. Green. 2000. *Linguistic Knowledge can Improve Information Retrieval.* Technical Report No. TR-99-83, Sun Microsystems Laboratories, Burlington, MA.