Concept Indexing for Automated Text Categorization

José María Gómez[1], José Carlos Cortizo[2], Enrique Puertas[1], and Miguel Ruiz[2]

[1] Universidad Europea de Madrid
Villaviciosa de Odón, 28670 Madrid, Spain
jmgomez@uem.es, epuertas@uem.es

[2] AINet Solutions
Fuenlabrada, 28943, Madrid, Spain
jccp@ainetsolutions.com, ethan113@hotmail.com

Abstract. In this paper we explore the potential of concept indexing with WordNet synsets for Text Categorization, in comparison with the traditional bag of words text representation model. We have performed a series of experiments in which we also test the possibility of using simple yet robust disambiguation methods for concept indexing, and the effectiveness of stoplist-filtering and stemming on the SemCor semantic concordance. Results are not conclusive yet promising.

1 Introduction

Automated Text Categorization (ATC) – the automated assignment of documents to predefined categories – is a text classification task that has attracted much interest in the recent years. Many types of documents (reports, books, Web pages, email messages, etc.) can be classified according to different kinds of categories, most often thematic or subject oriented: the Library of Congress Subject Headings, the categories directories like Yahoo!, etc. (see [1] for a survey).

While it is possible to build an ATC system by manually writting sets of classification rules, the most popular approach nowadays consist of using Information Retrieval (IR) and Machine Learning (ML) techniques to automatically induce a classification system called a *classifier* [1]. Such a classifier is obtained by: (1) representing (indexing) a set of manually classified documents (the *training collection*) as term weight vectors – as in the Salton's Vector Space Model [2], and (2) learning a classification function that can be a set of rules, a decision tree, etc. This approach is quite effective for subject-based ATC, producing accurate classifiers provided enough training data is available.

An important decision in this learning-based model is the definition of what a term is. Most often, terms are defined as stoplist filtered, stemmed words. This may be sufficient for accurate learning, since individual words carry an important part of the meaning of text. However, this does not always hold, due to polysemy and synonymy of words. So, a number of term definitions have been tested in the bibliography, including word phrases obtained by statistical

or linguistic analysis (e.g. [3, 4]), Information Extraction patterns (as in [5]), and WordNet sets of synonyms or *synsets* (e.g. [6, 7]).

The latter indexing term definition is specially promising, according to some results obtained for IR tasks (see e.g. [8]). In this paper, we focus in using WordNet sysnets as indexing units, in order to address the problems of synonymy and polysemy that are faced by text classification tasks. In the literature, a number of approaches have been tested, with mixed results [6, 9–11, 7]. To the date, however, there is no an in-depth study using a wide range of text representation options, feature selection methods, learning algorithms and problem definitions. This paper attempts to be such a study, in which we report experiments to test the hipothesis that concept indexing using WordNet synsets is a more effective text representation than using stoplist filtered, stemmed words as indexing terms. In our experiments, we have tested a number of text representations for a representative range of ML algorithms. Also, and since indexing with WordNet synsets involves a word sense disambiguation process, several disambiguation strategies have been tested.

We have organized this paper the following way. First, we introduce the most popular text representation approaches used in the literature. Then, we focus in conceptual indexing using WordNet synsets. The design of our experiments is after described, including the text representations and ML algorithms tested, the benchmark text collection, and the evaluation metrics. Then we present and analyze the results of our experiments, and finally our conclusions and future work are described.

2 Text Representation Approaches for Automated Text Categorization

Text classification tasks, such as IR, ATC, Text Filtering and Routing, and others, share the necessity of representing or indexing documents. It is not guaranteed that a representation method will be equally effective for all of them, but an improvement in text representation for TC may be of benefit for other tasks. In fact, some researchers support that TC is an excellent vehicle to studying different text representations. In words of Lewis ([4], page 72): "(...) We mentioned a number of problems with comparing text representations on a standard text retrieval test collection. (...) Text categorization as a vehicle for studying text representation addresses all (...) of this problems. (...) Of text classification tasks, text categorization seems best suited to studying text representation."

In TC, documents are usually represented as term weight vectors, as in the Vector Space Model for IR [1]. In this model, called the *bag of words* representation in the literature, terms are most often defined as words stems, after filtering by using a stoplist, and aplying a stemming algorithms like Porter's. Weights can be binary (1 representing that a word stem occurs in the document, and 0 in the other case), TF (Term Frequency, the number of times that a word stem occurs in the document), or TF.IDF (in which IDF stands for Inverse Document Frequency, usually defined as $\log_2 (n/df(t))$, being n the number of documents used

for learning, and $df(t)$ the number of documents in which the term t occurs). This term-weight (or in the vocabulary of ML, attribute-value) representation allows learning algorithms to be applied, obtaining a classification model called a classifier. Depending on the selected learning method, it can be needed to select a subset of the original terms[3] according to some quality metric, like Information Gain or χ^2 (see [1] for others).

The definition of term, or in other words, the indexing unit, is critical in TC. Terms must be good from a semantic point of view (that is, they must capture as much as possible the meaning of the texts), and from a learning point of view (that is, the must allow efficient and effective learning). A number of alternative definitions of term have been tested in the bibliography, including: (1) Character n-grams, that are short sequences of alphanumeric characters [12], appropiate for documents with OCR errors; (2) statistical and linguistic phrases (see e.g. [3, 4, 7]), that is, multi-word expressions, either built using coocurrence statistics in documents, or by shallow Natural Language Processing; its usage has not proven definitely superior to word stem based indexing, although it seems promising [3]; and (3) Information Extraction patterns, used in conjunction with the Relevancy Signatures algorithm by Riloff and others [5]; the patterns, called *signatures*, are $\langle$*word, semantic_node*$\rangle$ pairs in which words act as semantic node triggers, and these are defined for the domain at hand. This latter approach guarantees high precision TC, yet recall can be hurt. Some other approaches have been tested, but none has been clearly proven better than the bag of words representation, over a range of Machine Learning algorithms and a variety of application domains.

3 Concept Indexing with WordNet synsets

The popularity of the bag of words model is justified by the fact that words and its stems carry an important part of the meaning of a text, specially regarding subject-based classification. However, this representation faces two main problems: the synonymy and the polysemy of words. These problems are addressed by a concept indexing model using WordNet synsets.

3.1 The Lexical Database WordNet

WordNet sysnets are excellent candidates to index terms in text classification tasks, and allow addressing these problems. WordNet is a Lexical Database that accumulates lexical information about the words of the English language [13]. WordNet uses Synonym Sets or synsets as basic information and organization units. A synset contains a number of synsets that define a concept, which is one of the possible senses of the words or collocations in the synset. WordNet also stores information about lexical and conceptual relations between words, and concepts, including hyponymy or IS-A links, meronymy or HAS-A links, and others. This kind of information in WordNet makes it more a very populated semantic net

[3] This process is often called "feature selection" in the literature.

and ontology than an electronic dictionary or thesaurus. Current contents of WordNet (1.7.1) include more than 146,000 words and collocations and 111,000 synsets for nouns, verbs, adjetives and adverbs of the English language.

3.2 WordNet synsets as indexing units for TC

The wide coverage of WordNet and its free availability has promoted its utilization for a variety of text classification tasks, including IR and TC[4]. While WordNet usage for text classification has not proven widely effective (see e.g. [7, 14]), some works in which WordNet synsets are used as indexing terms for IR and TC are very promising [8, 6, 15, 11]; see [16] for an in-depth discussion.

The basic idea of concept indexing with WordNet synsets is recognizing the sysnets to which words in texts refer, and using them as terms for representation of documents in a Vector Space Model. Synset weights in documents can be computed using the same formulas for word stem terms in the bag of words representation. This concept based representation can improve IR, as commented by Gonzalo *et al.* [8]: "(...) using WordNet sysnets as indexing space instead of word forms (...) combines two benefits for retrieval: one, that terms are fully disambiguated (this should improve precision); and two, that equivalent terms can be identified (this should improve recall)."

Experiments focused on concept indexing with WordNet synsets for TC have mixed results. On one side, lack of disambiguation has led to loss of effectiveness in some works. On the other, it is not clear that full disambiguation is absolutely required to obtain a document representation more effective than the bag of words model. We discuss three works specially relevant.

Scott and Matwin [7] have tested a text representation in which WordNet synsets corresponding to the words in documents, and their hypernyms, were used as indexing units with the rule learner Ripper on the Reuters-21578 test collection. The results of the experiments were discouraging, probably due to the fact that no disambiguation at all is performed, and to the inability of Ripper to accurately learn in a highly dimensional space.

Fukumoto and Suzuki [6] have performed experiments extracting synonyms and hypernyms from WordNet nouns in a more sophisticated fashion. First, sysnets are not used as indexing units; instead, words extracted from synsets whose words occur in the documents are used. Second, the heigth to which the WordNet hierarchy is scanned is dependent on the semantic field (location, person, activity, etc.), and estimated during learning. These experiments were performed with Support Vector Machines on the Reuters-21578 test collection, and their results are positive, with special incidence on rare (low frequency) categories. Notably, no sense disambiguation was performed.

Petridis *et al.* [11] used WordNet synsets as indexing units with several learning algorithms on the Semcor text collection. In this collection, all words and collocations have been manually disambiguated with respect to WordNet synsets.

[4] See WordNet home page (`http://www.cogsci.princeton.edu/~wn/`) for a long bibliography.

The lazy learner k-Nearest Neigbors, the probabilistic approach Naive Bayes, and a Neural Network were tested on several text representations. The concept indexing approach performed consistently better than the bag of words model, being the Neural Network the best learner.

The work by Scott and Matwin suggests that some kind of disambiguation is required. The work by Fukumoto and Suzuki allows to suppose that no full disambiguation is needed. Finally, the work by Petridis *et al.* demonstrates that perfect disambiguation is effective, over a limited number of learning algorithms and an correctly disambiguated text collection.

However, to the date no experiments have been performed to test concept indexing over a representative range of learning algorithms and feature selection strategies. Also, the effect of of disambiguation remains to be tested. More in detail, current technologies are still far of perfect disambiguation (see the SENSEVAL conferences results [17]), and the discussion about the level of accuracy required for disambiguation to help IR [16]. However, there are simple but robust disambiguation strategies not yet tested in TC: disambiguation by part of speech – if a word has two or more possible part of speech categories (e.g. noun, verb, etc), only the senses for the right part of speech are taken[5]; and disambiguation by frequency – the most frequent sense (on a reference corpus[6]) for the correct part of speech is taken.

Our experiments are so focused on testing the potential of WordNet synsets as indexing units for TC, considering a representative range of feature selection strategies ans learning algorithms. Also, we compare several concept indexing strategies (with perfect disambiguation, disambiguation by part of speech, and by frequency) and bag of words representations.

4 The design of experiments

In this section, we describe our experiments setup, with special attention to the benchmark collection and evaluation metrics used in out work, and the text representation approaches and learning algorithms tested.

4.1 The SemCor benchmark collection

The SemCor text collection [18] is a Semantic Concordance, a corpus tagged with WordNet senses in order to supplement WordNet itself (for instance, for researching or showing examples of sense usage). However, SemCor has been adapted and used for testing IR by Gonzalo *et al.* [8], and used for evaluating TC by Petridis *et al.* [11]. Moreover, there are not other collections tagged with conceptual information in depth, and so, indexing with "perfect" disambiguation can hardly tested without SemCor.

[5] Note that several senses, and thus, synsets, are selected.

[6] E.g. The corpus from which the senses were taken. WordNet senses are ordered according to their frequency in the reference corpus used building it.

SemCor is a subset of the Brown Corpus and the "The Red Badge of Courage" of about 250,000 words, in which each word or collocation is tagged with its correct WordNet sense using SGML. It is important to note that available information in SemCor allows both sense and concept indexing. As sense indexing, we understand using word senses as indexing units. For instance, we could use the pair (car, sense 1) or "car_s1" as indexing unit. Concept indexing involves a word-independent normalization that allows recognizing "car_s1" and "automobile_s1" as occurrences of the same concept, the noun of code 02573998 in WordNet (thus addressing synonymy and polysemy simultaneously).

SemCor needs not to be adapted for testing TC. It contains 352 text fragments obtained from several sources, and covering 15 text genres, like Press: Reportage, Religion, or Fiction: Science (see the full list in Table 1). We have used the first 186 text fragments (fully tagged) as testing documents, and the 15 genres as target categories. We must note that classes are not overlapping, being each document in exactly one category. Also, the classification problem is closer to genre than subject-based TC, although some genre differences are also related to the topics covered by the texts. Finally, the number of documents in each category is variable, from 2 (Press: Editorial) to 43 (Learned).

Given that classes are not overlapping, the TC problem is a multi-class problem (select the suitable category among $m = 15$ for a given document). However, some of the learning algorithms used in our work (e.g. Support Vector Machines) only allow binary class problems (classifying a document in a class or not). The SemCor multi-class TC problem can be transformed into $m = 15$ binary problems, in which a classifier is build for each class[7]. We have performed experiments on the binary problems for all algorithms, and in the multi-class only for those allowing it, as described below.

4.2 Evaluation Metrics

Effectiveness of TC approaches is usually meassured using IR evaluation metrics, like recall, precision and F_1 [1]. We have used F_1, more suitable for TC than others used in related work (like accuracy in [11]). This metric is an average of recall and precision, and can be averaged on the categories by macro- and micro-averaging. The first kind of average gives equal importance to all categories, while the second gives more importance to more populated categories. It is sensible to compute both. Macro- and micro-averaged F_1 values are noted as F_1^M and F_1^m respectively.

Often, TC test collections are divided into two parts: one for learning, and one for testing (the most prominent example is the Reuters-21578 test collection, with three consolidated partitions). When there is not a reference partition, or training data is limited, a k-fold cross validation process is used to estimate the evaluation metrics values. In short, the data is randomly divided in k groups (preserving class distribution), and k experiments are run using $k-1$ groups for

[7] This kind of transformation is called "one against the rest" in the literature [11].

Table 1. Number of documents and indexing units per class (binary problem), for the seven text representation approaches tested, and selection with IG > 0. The last row shows the number of potential units.

Name	# docs	CD	CF	CA	BNN	BNS	BSN	BSS
Press: Reportage	7	845	536	1363	1133	840	1025	745
Press: Editorial	2	363	275	482	371	249	362	241
Press: Reviews	3	404	306	603	530	396	495	369
Religion	4	406	314	644	406	267	383	251
Skill and Hobbies	14	1487	1052	1879	1354	766	1258	683
Popular Lore	19	1540	1046	1455	1585	973	1494	895
Belles-Lettres	18	1482	1051	1604	1527	959	1428	876
Miscellaneous	12	1071	766	1658	1071	691	899	531
Learned	43	995	815	3091	1062	861	909	713
Fiction: General	29	714	596	2183	738	625	624	515
Fiction: Mystery	11	895	605	1288	822	567	738	484
Fiction: Science	2	296	240	415	326	230	280	194
Fiction: Adventure	10	1180	788	1791	1140	826	1031	722
Fiction: Romance	6	558	415	875	535	359	499	324
Humor	6	682	544	862	853	615	788	560
Initial (NOS)	186	25867	18780	37038	25728	16679	24188	15327

learning, and 1 for testing. The values obtained on each experiment are averaged on the k runs. Our test have been performed with 10-fold cross validation.

4.3 Text representation approaches

We have tested two kinds of text representation approches for TC: a bag of words model, and a concept indexing model. Given that the TC problem is more oriented to genre than to subject, we have considered four possibilities: using a stoplist and stemming, using a stoplist, using stemming, and finally using words (without stoplist and stemming). These approaches are coded below as BSS, BSN, BNS, and BNN respectively. The motivation of considering these four approaches is that words ocurring in a stoplist (e.g. prepositions, etc.) and original word forms (e.g. past suffixes as "-ed") can be good indicators of different text genres (see e.g. [19]).

The three concept indexing approaches considered in our experiments are those regarding the level of disambiguation, which are: using the correct word sense (CD), using the first sense given the part of speech (CF), and using all the senses for the given part of speech (CA). We have weighted terms and synsets in documents using the popular TF.IDF formula from the Vector Space Model[8].

Usually in TC, a feature selection process is applied in order to detect the most informative features (indexing units), and to decrease the problem dim-

[8] The indexing process have been performed with the experimental package `ir.jar` by Mooney and colleagues (`http://www.cs.utexas.edu/users/mooney/ir-course/`).

Table 2. Number of indexing units per representation (multi-class problem), for each selection strategy.

Selection	CD	CF	CA	BNN	BNS	BSN	BSS
NOS	25867	18780	37038	25728	16679	24188	15327
S00	31	34	196	64	66	20	27

mensionality allowing efficient learning. An effective approach is selecting the top scored indexing units according to a quality metric, as Information Gain (IG). We have tested for selection approaches: no selection, selecting the top 1% terms or concepts, selecting the top 10% terms or concepts, and selecting those indexing units with IG score over 0. These approaches are coded below as NOS, S01, S10 and S00 respectively. The percentages are justified by other's work [20].

In the Table 1, we show the resulting numbers of indexing units for each text representation approach and category, with S00 (which is class-dependent in the binary problems), and the original number of units (NOS) for each representation. Note there is not a strong correlation between the number of documents and units per class. For instance, the most populated classes (Learned and Fiction: General) have less indexing synsets than other less populated classes (e.g. Skill and Hobbies). Also, and as expected, the number of total indexing units is such as CA > CD > CF, and BNN > BSN > BNS > BSS.

Also, we report the number of documents per class. In the Table 2, we show the number of indexing units per representation in the multi-class problems. It is remarkable that the number of units when S00 (IG > 0) is applied, is quite smaller than in the binary problems; This suggests that it is harder to find e.g. a word able to inform (decide) on all classes as a whole, than in a binary classification problem.

4.4 Machine Learning algorithms

It is important for our hypothesis to test a representative range of high performance learning algorithms. From those tested in the literature [1], we have selected the following ones[9]: the probabilistic approach Naive Bayes (NB); the rule learner PART [21]; the decision tree learner C4.5; the lazy learning approach k-Nearest Neighbors (kNN), used $k = 1, 2, 5$ neighbors in our experiments; the Support Vector Machines kernel method; and the AdaBoost meta-learner applied to Naive Bayes and C4.5. In our experiments, we have made use of the WEKA[10] learning package, with default parameters for all algorithms except for kNN, where we selected weighted voting instead of majority voting, and AdaBoost, where we tried 10 and 20 iterations.

[9] We exclude some references for brevity. Please find a sample at [1].

[10] See `http://www.cs.waikato.ac.nz/ml/weka/`.

Table 3. Summary of best binary classification results, obtained with $k = 1$ for kNN, 10 iterations for ABNB and 20 for ABC4.5, S00 for NB and ABNB, and S01 for the rest of learning algorithms. Best F_1^M and F_1^m scores per algorithm are shown in boldface.

		NB	kNN	C4.5	PART	SVM	ABNB	ABC4.5
CD	macro	.631	.128	.253	.258	.502	**.638**	.262
	micro	.739	.224	.375	.403	**.773**	.759	.425
CF	macro	.554	.166	.184	.267	.461	.539	.228
	micro	.690	.304	.311	.431	.699	.680	.434
CA	macro	.611	**.245**	.219	.238	**.567**	.587	.226
	micro	.665	**.409**	.351	.427	.690	.680	.439
BNN	macro	**.635**	.130	**.270**	**.308**	.482	**.638**	**.367**
	micro	**.750**	.287	.382	.460	.730	**.760**	.526
BNS	macro	.536	.194	.245	.277	.538	.522	.290
	micro	.694	.347	**.431**	**.479**	.709	.682	.509
BSN	macro	.587	.116	.248	.262	.451	.577	.346
	micro	.722	.241	.351	.422	.685	.717	**.531**
BSS	macro	.494	.171	.198	.254	.518	.502	.296
	micro	.646	.325	.393	.459	.671	.649	.511

5 Results and Analysis

Our experiments involve computing a pair (F_1^M, F_1^m) per algorithm, feature selection approach, kind of indexing unit, and classification problem. Given that we consider three k values for kNN, 10 and 20 iterations for AdaBoost applied to 2 algorithms, and that SVM is only applied in the binary classification problems, these experiments produce $10 \times 4 \times 7 = 280$ binary classification F_1 pairs, and $9 \times 4 \times 7 = 252$ multi-class F_1 pairs. For brevity, we have summarized our experiments in Tables 3 and 4: the pair (F_1^M, F_1^m) for each learning algorithm/indexing approach is presented, according to the best selection metric.

Also, we have compared the number of times an indexing approach was better than others, focusing on answering: (a) Is conceptual indexing better than the bag of words model? (b) Are the simple disambiguation methods effective? and (c) Which is the impact of using stemming and a stoplist on effectiveness, in the bag of words model? We have compared the F_1 pairs for each algorithm and selection approach, leading to $11 \times 4 = 44$ comparisons for binary classification, and $10 \times 4 = 40$ comparisons for the multi-class problem. The results of these comparisons are summarized in the Table 5.

Examining these tables and the omitted results, and focusing on our research goals, we find that there is not a dominant indexing approach over all algorithms. Effectiveness of learning algorithms and feature selection methods is quite variable, partly motivated by the small size of training data. Remarkably, Naive Bayes, Support Vector Machines and AdaBoost with Naive Bayes perform comparably on the binary case, which is the most general and interesting one.

Table 4. Summary of best multi-class classification results, obtained with $k = 1$ for kNN, 20 iterations for AB*, S00 for kNN, S10 for ABC4.5, and S01 for the rest of learning algorithms. Best F_1^M and F_1^m scores per algorithm are shown in boldface.

		NB	kNN	C4.5	PART	ABC4.5	ABNB
CD	macro	.319	**.346**	.261	.248	.339	.265
	micro	.446	**.414**	.403	**.419**	.468	.425
CF	macro	.274	.290	.244	.234	.282	.247
	micro	.368	.330	.373	.368	.427	.373
CA	macro	.232	.271	.199	.181	.297	.216
	micro	.317	.349	.290	.253	.441	.414
BNN	macro	**.329**	.281	.273	**.254**	.343	**.321**
	micro	**.484**	.366	.376	.403	.462	**.489**
BNS	macro	.291	.228	.224	.243	.277	.276
	micro	.425	.344	.323	.355	.419	.414
BSN	macro	.296	.292	**.373**	.244	**.396**	.277
	micro	.409	.349	**.425**	.333	**.473**	.446
BSS	macro	.253	.313	.281	.211	.297	.266
	micro	.355	.344	.301	.306	.414	.387

When comparing concept indexing and the bag of words model, we find that perfect disambiguated concept indexing (CD) outperforms all the bag of words approaches (B*) on 20 of 84 times. Also, the highest F_1^M value are obtained with ABNB+CD or BNN (binary) and ABC45+BSN (multi-class), and the highest F_1^m are obtained with SVM+CD(binary) and ABNB+BNN (multi-class). In general, there is not a strong evidence of that concept indexing outperforms the bag of words model on the SemCor collection. The lack of training data, the skewness of class distribution, and the fact that other semantic relations in WordNet have been not used in the experiments, can explain these results.

Focusing on the disambiguation problem, we find that perfect disambiguation improves the two other approaches tested on 57 of 84 times, and rarely (except on kNN) has worst results than any of them on Tables 3 and 4. We believe that errors in disambiguation can have an important impact on performance, perhaps more than those reported by Gonzalo *et al.* [8]. Also, when testing the bag of words model on this genre-oriented classification task, we find that *not* using stoplist and stemmer is better than other combinations 20 of 84 times, and the behavior on the figures reported on Tables 3 and 4 is rather erratic. We consider that a more in depth study of the application of genre oriented features (including e.g. tense, punctuation marks, etc.) is needed in this collection.

Regarding other's work, the experiments by Petridis *et al.* [11] are the most similar to ours, but far from comparable. This is due to the fact that they evaluate the learning algorithms on a different subset of Semcor, and by 3-fold cross validation, showing their results for a the less suitable evaluation meassure: accuracy. However, their work has partially inspired ours.

Table 5. Results of comparisons between (a) perfect disambiguation conceptual indexing vs. indexing with the two other disambiguation methods (CD > C*), (b) the bag of words model without stemming and stoplist vs. the other combinations (BNN > B*), and (c), the perfect disambiguation conceptual indexing vs. the bag of words model for all stoplist/stemming combinations (CD > B*).

	F_1^M	F_1^m	Both	Of
# CD > C* (Bin)	13	16	22	44
# CD > C* (MC)	31	33	35	40
# CD > C* (All)	44	49	57	84
# BNN > B* (Bin)	9	20	12	44
# BNN > B* (MC)	14	19	19	40
# BNN > B* (All)	23	39	31	84
# CD > B* (Bin)	6	12	9	44
# CD > B* (MC)	14	21	16	40
# CD > B* (All)	20	33	25	84

6 Conclusions

In general, we have not been able to prove that concept indexing is better than the bag of words model for TC. However, our results must be examined with care, because of at least two reasons: (1) the lack of enough training data: stronger evidence is got when the number of documents increases; also, the behavior of some algorithms is surprising (Naive Bayes, kNN); and (2) the genre identification problem is such that meaning of used words is not more relevant than other text features, including structure, capitalization, tense, puntuation, etc. In other words, this study needs to be extended to a more populated, subject-oriented TC test collection (e.g. Reuters-21578). The work by Petridis *et al.* [11] adds evidence of concept indexing outperforming the bag of words model on the SemCor collection, specially with SVM, and Fukumoto and Suzuki [6] work with them on Reuters-21578 allow to say that such an study is weel motivated and promising.

We also plan to use semantic relations in WordNet in our future experiments on the Semcor and Reuters test collections. The results published in literature (e.g. [6]) suggest that using a limited amount of this information can improve our results in concept indexing.

ATC is a different task to IR. We think that, being TC categories more static than IR queries, and given that it is possible to apply ML feature selection strategies in TC, it is more likely that using concept indexing in TC will outperform the bag of words model, against Voorhees findings in IR [14].

References

1. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34** (2002) 1–47

2. Salton, G.: Automatic text processing: the transformation, analysis, and retrieval of information by computer. Addison Wesley (1989)
3. Caropreso, M., Matwin, S., Sebastiani, F.: A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In: Text Databases and Document Management: Theory and Practice. Idea Group Publishing (2001) 78–102
4. Lewis, D.D.: Representation and learning in information retrieval. PhD thesis, Department of Computer Science, University of Massachusetts, Amherst, US (1992)
5. Riloff, E.: Using learned extraction patterns for text classification. In: Connectionist, statistical, and symbolic approaches to learning for natural language processing, Springer Verlag (1996) 275–289
6. Fukumoto, F., Suzuki, Y.: Learning lexical representation for text categorization. In: Proceedings of the NAACL 2001 Workshop on WordNet and Other Lexical Resources. (2001)
7. Scott, S.: Feature engineering for a symbolic approach to text classification. Master's thesis, Computer Science Dept., University of Ottawa, Ottawa, CA (1998)
8. Gonzalo, J., Verdejo, F., Chugur, I., Cigarrán, J.: Indexing with WordNet synsets can improve text retrieval. In: Proceedings of the COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems. (1998)
9. Junker, M., Abecker, A.: Exploiting thesaurus knowledge in rule induction for text classification. In: Proceedings of the, 2nd International Conference on Recent Advances in Natural Language Processing. (1997) 202–207
10. Liu, J., Chua, T.: Building semantic perceptron net for topic spotting. In: Proceedings of 37th Meeting of Association of Computational Linguistics. (2001)
11. Petridis, V., Kaburlasos, V., Fragkou, P., Kehagias, A.: Text classification using the σ-FLNMAP neural network. In: Proceedings of the 2001 International Joint Conference on Neural Networks. (2001)
12. Cavnar, W., Trenkle, J.: N-gram-based text categorization. In: Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, US (1994) 161–175
13. Miller, G.A.: WordNet: A lexical database for English. Communications of the ACM **38** (1995) 39–41
14. Voorhees, E.: Using WordNet for text retrieval. In: WordNet: An Electronic Lexical Database. MIT Press (1998)
15. Mihalcea, R., Moldovan, D.: Semantic indexing using WordNet senses. In: Proceedings of ACL Workshop on IR and NLP. (2000)
16. Stokoe, C., Oakes, M.P., Tait, J.: Word sense disambiguation in information retrieval revisited. In: Proceedings of the 26th ACM International Conference on Research and Development in Information Retrieval. (2003)
17. Kilgarriff, A., Rosenzweig, J.: Framework and results for english SENSEVAL. Computers and the Humanities **34** (2000) 15–48
18. Miller, G.A., Leacock, C., Tengi, R., Bunker, R.: A semantic concordance. In: Proc. Of the ARPA Human Language Technology Workshop. (1993) 303–308
19. Kessler, B., Nunberg, G., Schütze, H.: Automatic detection of text genre. In: Proceedings of ACL-97, 35th Annual Meeting of the Association for Computational Linguistics, Madrid, ES (1997) 32–38
20. Yang, Y., Pedersen, J.: A comparative study on feature selection in text categorization. In: Proc. Of the 14th International Conf. On Machine Learning. (1997)
21. Frank, E., Witten, I.H.: Generating accurate rule sets without global optimization. In Shavlik, J., ed.: Machine Learning: Proceedings of the Fifteenth International Conference, San Francisco, CA, Morgan Kaufmann Publishers (1998)