Expanding Domain-Specific Lexicons by Term Categorization

Henri Avancini
ISISTAN-UNCPBA
7000 Tandil, Argentina
henri@exa.unicen.edu.ar

Alberto Lavelli
ITC-irst
38050 Trento, Italy
lavelli@itc.it

Bernardo Magnini
ITC-irst
38050 Trento, Italy
magnini@itc.it

Fabrizio Sebastiani
ISTI-CNR
56124 Pisa, Italy
fabrizio@iei.pi.cnr.it

Roberto Zanoli
ITC-irst
38050 Trento, Italy
zanoli@itc.it

ABSTRACT

We discuss an approach to the automatic expansion of domain-specific lexicons by means of *term categorization*, a novel task employing techniques from information retrieval (IR) and machine learning (ML). Specifically, we view the expansion of such lexicons as a process of learning previously unknown associations between terms and *domains*. The process generates, for each c_i in a set $C = \{c_1, \ldots, c_m\}$ of domains, a lexicon L_1^i, bootstrapping from an initial lexicon L_0^i and a set of documents θ given as input. The method is inspired by *text categorization* (TC), the discipline concerned with labelling natural language texts with labels from a predefined set of domains, or categories. However, while TC deals with documents represented as vectors in a space of terms, we formulate the task of term categorization as one in which terms are (dually) represented as vectors in a space of documents, and in which terms (instead of documents) are labelled with domains.

Keywords

Term categorization, lexicon generation, WordNet

1. INTRODUCTION

The generation of *domain-specific lexicons* (i.e. lexicons consisting of terms pertaining to a given domain or discipline) is a task of increased applicative interest, since such lexicons are of the utmost importance in a variety of tasks pertaining to natural language processing and information access. Unfortunately, the manual generation of domain-specific lexicons is expensive, since it requires the intervention of specialized manpower, i.e. lexicographers and domain experts working together. Many applications also require that the lexicons be not only domain-specific, but also tailored to the specific data tackled in the application.

In this paper we propose a methodology for the automatic expansion of domain-specific lexicons from a corpus of texts. This methodology relies on *term categorization*, a novel task that employs a combination of techniques from information retrieval (IR) and machine learning (ML). Specifically, we view the generation of such lexicons as a process of learning previously unknown associations between terms and *domains* (i.e. disciplines, or fields of activity). The process generates, for each c_i in a set $C = \{c_1, \ldots, c_m\}$ of predefined domains, a lexicon L_1^i, bootstrapping from a lexicon L_0^i given as input. Associations between terms and domains are learnt from a set of textual documents θ (hereafter called *corpus*); iterating this process allows to enlarge the lexicon as new corpora become available for learning. The process builds the lexicons $L_1 = \{L_1^1, \ldots, L_1^m\}$ for all the domains $C = \{c_1, \ldots, c_m\}$ in parallel, from the same corpus θ. The only requirement on θ is that at least some of the terms in each of the lexicons in $L_0 = \{L_0^1, \ldots, L_0^m\}$ should occur in it (if none among the terms in a lexicon L_0^j occurs in θ, then no new term is added to L_0^j).

This methodology is inspired by *text categorization*, the activity of automatically building, by means of machine learning techniques, automatic text classifiers, i.e. programs capable of labelling texts with (zero, one, or several) thematic categories from a predefined set $C = \{c_1, \ldots, c_m\}$ [8]. While the purpose of text categorization is that of classifying documents represented as vectors in a space of terms, the purpose of *term* categorization, as we formulate it, is (dually) that of classifying terms represented as vectors in a space of documents. In this task terms are thus items that may belong, and must thus be assigned, to (zero, one, or several) domains belonging to a predefined set. In other words, starting from a set Γ_0^i of preclassified terms, a new set of terms Γ_1^i is classified, and the terms in Γ_1^i which are deemed to belong to c_i are added to L_0^i to yield L_1^i. The set Γ_0^i is composed of lexicon L_0^i, acting as the set of "positive examples" of c_i, plus a set of terms known not to belong to c_i, acting as the set of "negative examples" of c_i.

For input to the learning device and to the term classifiers

SAC 2003, Melbourne, Florida USA

that this will eventually build, we use "bag of documents" representations for terms, dual to the "bag of terms" representations commonly used in text categorization. As the learning device we adopt AdaBoost.MH^{KR} [9], a variant of the AdaBoost.MH^{R} algorithm proposed in [5].

We have chosen a boosting approach not only because of its state-of-the-art effectiveness, but also because it naturally allows for a form of "data cleaning", which is useful in case a lexicographer wants to check the results and edit the newly generated lexicon. That is, in our term categorization context it allows the lexicographer to easily inspect the classified terms for possible misclassifications, since the algorithm, apart from generating the new lexicon L_1^i, ranks the terms in L_0^i in terms of their "hardness", i.e. how successful have been the generated classifiers at correctly recognizing their label. Since the highest ranked terms are the ones with the highest probability of having been misclassified [1], the lexicographer can examine this list starting from the top and stopping where desired, removing the misclassified examples. The process of generating a domain-specific lexicon may then become an iteration of generate-and-test steps.

This paper is organized as follows. In Section 2 we describe how we represent terms by means of a "bag of documents" representation. Section 3 discusses how to combine the indexing tools introduced in Section 2 with the boosting algorithm, and describes the role of the lexicographer in the lexicon expansion phase. Section 4 describes the results of our experiments, in which we attempt to expand (in parallel) 42 domain-specific lexicons by using a corpus of more than 800,000 documents. Section 5 concludes, pointing to avenues for improvement.

2. REPRESENTING TERMS IN A SPACE OF DOCUMENTS

In text categorization applications, the process of building internal representations of texts is called *text indexing*. In text indexing, a document d_j is usually represented as a vector of term *weights* $\vec{d}_j = \langle w_{1j}, \ldots, w_{rj} \rangle$, where r is the cardinality of the *dictionary* and $0 \leq w_{kj} \leq 1$ represents, loosely speaking, the contribution of t_k to the specification of the semantics of d_j. Usually, the dictionary is equated with the set of *terms* that occur at least once in at least α documents of Tr (with α a predefined threshold, typically ranging between 1 and 5). Different approaches to text indexing may result from different choices (i) as to what a term is and (ii) as to how term weights should be computed. A frequent choice for (i) is to use single words (minus stop words, which are usually removed prior to indexing) or their stems. Different "weighting" functions may be used for tackling issue (ii), either of a probabilistic or of a statistical nature; a frequent choice is the (normalized) $tfidf$ function, which provides the inspiration for the "term indexing" function we are going to use in this work.

Text indexing may be viewed as a particular instance of *abstract indexing*, a task in which "objects" are represented by means of "features", and whose underlying metaphor is, by and large, that the semantics of an object corresponds to the *bag of features* that "occur" in it. In order to illustrate an example of abstract indexing, let us define a *token* τ to be a specific occurrence of a given feature $f(\tau)$ in a given object $o(\tau)$, let T be the set of all tokens occurring in any of a set of objects O, and let F be the set of features of which the tokens in T are instances. Let us define the *feature frequency* $ff(f_k, o_j)$ of a feature f_k in an object o_j as

$$ff(f_k, o_j) = |\{\tau \in T \mid f(\tau) = f_k \ \wedge \ o(\tau) = o_j\}|$$

We next define the *inverted object frequency* $iof(f_k)$ of a feature f_k as

$$iof(f_k) = \log \frac{|O|}{|\{o_j \in O \mid \exists \tau \in T : f(\tau) = f_k \ \wedge \ o(\tau) = o_j\}|}$$

and the *weight* $w(f_k, o_j)$ of feature f_k in object o_j as

$$w_{kj} = w(f_k, o_j) = \frac{ff(f_k, o_j) \cdot iof(f_k)}{\sqrt{\sum_{s=1}^{|F|} (ff(f_s, o_j) \cdot iof(f_s))^2}}$$

We may consider the $w(f_k, o_j)$ function of this last equation as an *abstract indexing function*; that is, different instances of this function are obtained by specifying different choices for the set of objects O and set of features F. The previously mentioned *text* indexing function $tfidf$ is obtained by equating O with the training set of documents and F with the dictionary; T, the set of occurrences of elements of F in the elements of O, thus becomes the set of term occurrences. Dually, a *term* indexing function may be obtained by switching the roles of F and O, i.e. equating F with the training set of documents and O with the dictionary; T, the set of occurrences of elements of F in the elements of O, is thus again the set of term occurrences [7].

This approach to term representation is very elegant, in that it is based on a minimal set of assumptions (namely, the "extensional" assumption that objects can be represented as bags of features, and the assumption that occurrence can be used as "featurehood") and can be instantiated by means of any indexing technique (here we have used normalized $tfidf$), either from the tradition of text indexing or not. Note also that any program or data structure that implements a text indexing function may be used straightaway, with no modification, for term indexing: one needs only to feed the program with the terms in place of the documents and viceversa.

3. GENERATING DOMAIN-SPECIFIC LEXICONS BY SUPERVISED LEARNING

3.1 Operational methodology

We are now ready to describe the overall process that we will follow for the expansion of domain-specific lexicons. We start from a set of domain-specific lexicons $L_0 = \{L_0^1, \ldots, L_0^m\}$, one for each domain in $C = \{c_1, \ldots, c_m\}$, and from a corpus θ. We index the terms that occur in θ by means of the term indexing technique described in Section 2; this yields, for each term t_k, a representation consisting of a vector of weighted documents, the length of the vector being $r = |\theta|$. We then generate m classifiers $\Phi = \{\Phi^1, \ldots, \Phi^m\}$ by means of AdaBoost.MH^{KR} with $L_0 = \{L_0^1, \ldots, L_0^m\}$ as training set. While generating the classifiers, AdaBoost.MH^{KR} also produces, for each domain c_i, a ranking of the terms in L_0^i in terms of how hard it was for the generated classifiers to classify them correctly, which basically corresponds to their probability of being misclassified examples. The lexicographer can then, if desired, inspect L_0 and remove the misclassified examples, if any (possibly rerunning AdaBoost.MH^{KR} on the "cleaned" version of L_0). At this point, the terms occurring in θ that AdaBoost.MH^{KR} has classified under c_i

are added (possibly, after being checked by the lexicographer) to L_0^i, yielding L_1^i.

This process can be further iterated, by using new text corpora from which to "extract" new terms. In this case an alternative approach is to involve the lexicographer only after the last iteration, and not after each iteration.

3.2 Experimental methodology

The process we have described in Section 3.1 is the one we would apply in an operational setting. In an experimental setting, instead, we are also interested in evaluating the effectiveness of our approach on a benchmark. The difference with the process outlined in Section 3.1 is that at the beginning of the process the lexicon L_0 is split into a training set and a test set; the classifiers are learnt from the training set, and are then tested on the test set by checking how good they are at extracting the terms in the test set from the corpus θ.

We comply with standard text categorization practice in evaluating term categorization effectiveness by a combination of *precision* (π), the percentage of positive categorization decisions that turn out to be correct, and *recall* (ρ), the percentage of positive, correct categorization decisions that are actually taken. Since most classifiers can be tuned to emphasize one at the expense of the other, only combinations of the two are usually considered significant. Following common practice, as a measure combining the two we will adopt their harmonic mean, i.e. $F_1 = \frac{2\pi\rho}{\pi+\rho}$. When effectiveness is computed for several categories, the results for individual categories must be averaged in some way; we will do this both by microaveraging and macroaveraging, defined in the usual ways (see e.g. [8, Section 7]) and indicated by the "μ" and "M" superscripts, respectively. Microaveraging rewards classifiers that behave well on *frequent categories* (i.e. categories with many positive test examples), while classifiers that perform well also on infrequent categories are rewarded by macroaveraging. Whether one or the other should be adopted obviously depends on the application requirements.

4. EXPERIMENTS

In order to test our approach according to the methodology of Section 3.2 we need two types of resources: (i) a corpus θ, which provides the "implicit" representation for terms, and (ii) a set of domain-specific lexicons $L_0 = \{L_0^1, \ldots, L_0^m\}$.

As the corpus θ we have used Reuters Corpus Volume I (RCVI), a set of documents recently made available by Reuters[1] for text categorization experimentation and consisting of 806,812 news stories. Note that, although the texts of RCVI are labelled by thematic categories, we have not made use of such labels.

As the domain-specific lexicons we have used an extension of WordNet. WordNet [3] is a large, widely available, domain-generic, monolingual, machine-readable dictionary in which sets of synonymous words are grouped into synonym sets (or *synsets*) organized into a directed acyclic graph. WordNetDomains is an extension of WordNet [4] in which each synset has been labelled with one or more from a set of 164 thematic categories, called *domains*[2].

[1] http://www.reuters.com/

[2] From the point of view of our term categorization task, the fact that more than one domain may be attached to the same synset means that ours is a *multi-label* categorization task [8, Section 2.2].

For the purpose of the experiments reported in this paper, we have used a simplified variant of WordNetDomains, called WordNetDomains(42). This was obtained from WordNetDomains by considering only 42 highly relevant domains, and tagging by a given domain c_i also the synsets that, in WordNetDomains, were tagged by the domains immediately related to c_i in a hierarchical sense (that is, the parent domain of c_i and all the children domains of c_i). This restriction to the 42 most significant domains allows to obtain a good compromise between the conflicting needs of avoiding data sparseness and preventing the loss of relevant semantic information.

4.1 The results

Figure 1 reports several experiments run for different choices of the subset of RCVI chosen as the corpus θ. We first describe the structure of a generic experiment, and then go on to describe the sequence of different experiments we have run. In our experiments so far we have considered only nouns (there are 32,509 of them in WordNetDomains).

In each experiment, we perform a *document filtering* phase by discarding all documents that do not contain any term from the training lexicon Tr, since they do not contribute in representing the meaning of training terms, and thus could not possibly be of any help in building the classifiers. Next, we perform a *term filtering* phase, in which we discard (i) all "empty" training terms, i.e. training terms that are not contained in any document of θ, since they could not possibly contribute to learning the classifiers; (ii) empty test terms, since no algorithm that extracts terms from corpora could possibly extract them; (iii) terms that occur in θ but belong neither to the training set Tr nor to the test set Te.

We then lemmatize all remaining documents and annotate the lemmas with part-of-speech tags; we also use the WordNet morphological analyzer in order to resolve ambiguities and lemmatization mistakes. The final set of terms that results from this process is randomly divided into a training set Tr (consisting of two thirds of the entire set) and a test set Te (one third). As negative training examples of category c_i we choose all the training terms that are not positive examples of c_i. We have repeated each experiment several times by considering only training and test terms occurring in at least x documents, so all curves in Figure 1 plot F_1 as a function of x.

4.1.1 Experiment 1: Using a subset of RCVI

In our first experiment (see Figure 1(top) – "one month") we have used only a subset of the RCVI corpus, corresponding to the news stories produced in an entire month (01.11.1996 to 30.11.1996 – 67,953 documents), with the purpose of getting a feel for the dimensions of the problem that need investigation; for the same reason, in the same experiment we have used only a small number of boosting iterations (500). There are 6,636 terms in WordNetDomains after the term filtering phase described above.

The low values of F_1 are mostly the result of low recall values, while precision tends to be much higher; for instance, the F_1^μ value of .203, obtained for $x = 60$, is the result of the values $\pi^\mu = .760$ and $\rho^\mu = .117$. One way of improving F_1 could be tuning ADABOOST.MH^{KR} so as to increase recall at the expense of precision, since F_1 is maximized when precision equals recall. Although this would be easy (e.g.

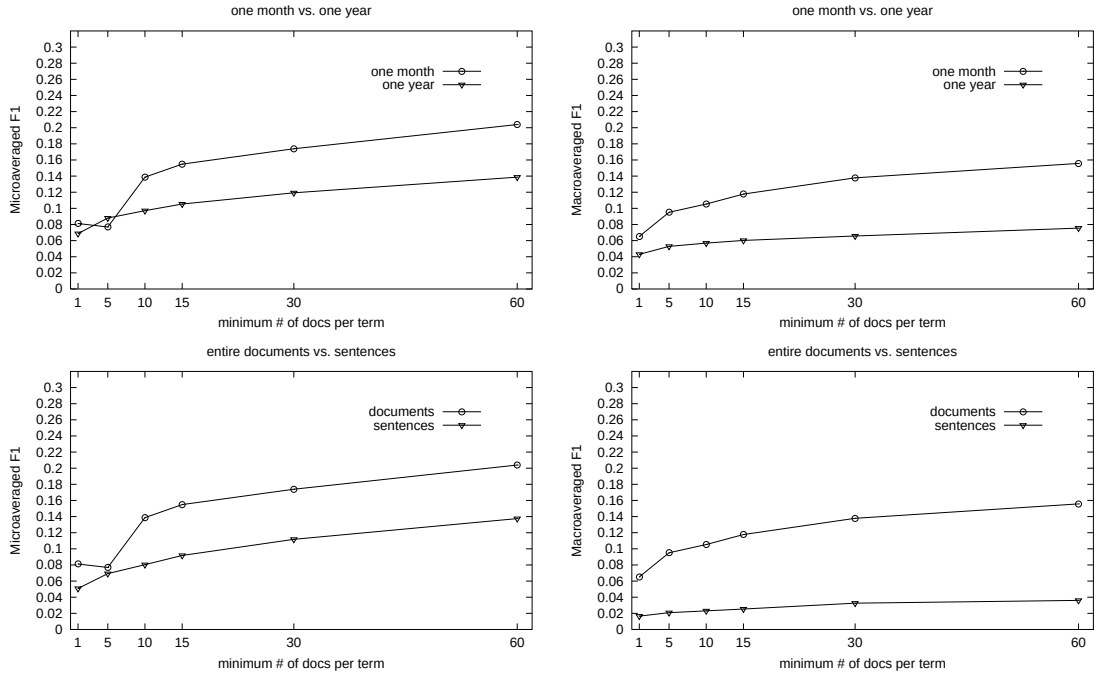

Figure 1: Results obtained on the automated lexicon expansion task. Plots report micro-averaged F_1 (leftmost) and macro-averaged F_1 (rightmost) as a function of x, representing the minimal number of documents in which training and test terms must occur in order to be taken into consideration.

by using the simple technique described in [6]), we have not pursued this line of work, for the simple reason that this work would not bring interesting insights into the problem.

The results in Figure 1(top) show a constant and definite improvement when higher values of x are used, despite the fact that, as we have found, higher levels of x mean a higher degree of polysemy, i.e. a higher average number of labels per term (e.g. this increases from 1.66 for $x = 1$ to 2.25 for $x = 60$), which tends to confuse a learning device. This behaviour is not surprising, since when a term occurs e.g. in one document only, this means that only one entry in the vector that represents the term is non-null (i.e. significant). This is in sharp contrast with text categorization, in which the number of non-null entries in the vector representing a document equals the number of distinct terms contained in the document, and is usually at least in the hundreds.

4.1.2 Experiment 2: Using the full RCVI

In our second experiment we have run our system on the entire set of 806,812 RCVI documents (this means 27,048 terms left in WordNetDomains after term filtering), since we wanted to test whether performance can be improved by increasing significantly the number of documents from which terms have to be "extracted". That this should happen would be a plausible hypothesis, since more documents mean, on average, a higher number of occurrences per term (since there is a finite number of terms in WordNetDomains(42)), hence a more reliable indication of the typical contexts in which a given term occurs. The results of this experiment (reported in Figure 1(top) – "one year") indicate that this is not the case, since in going from 67,953 documents to 806,812 documents, performance deteriorates. One likely explanation of this fact is that this move produces a sharp decrease of the ratio between the number of objects and the number of features that describe these objects, a ratio that is conceptually akin to the one between the constraints of a problem and the number of its variables.

We have then run a small experiment (not reported in Figure 1, since we have run just the $x = 1$ case) aimed at verifying whether this somehow disappointing result might also be due to RCVI containing too many documents that were not significant enough in determining the "meaning" of our terms. This experiment consisted in first applying a pass of *feature selection* aimed at selecting the documents that are the best discriminators between the presence and the absence of a category. Following common text categorization practice, we scored each of the 806,812 RCVI documents by the *information gain* function (globalized by means of the f_{max} method – see e.g. [8, Section 5.4]), and selected the 8% best scoring documents. The performance improvement was not terribly significant, i.e. we obtained only a 5% improvement over the F_1^{μ} value obtained without feature selection on the full RCVI. This is in line with the result obtained in the text categorization experiments of [10].

4.1.3 Experiment 3: Using sentences instead of documents

In our third experiment we reverted to the original set of documents of Experiment 1 and tested whether sentences can be better features than documents in term categorization. We obtained this by segmenting each of the 67,953 documents into sentences (i.e. using the full period as the separator) and considering each of the resulting 714,352 sentences as a "document". That this could be the case would

be a plausible hypothesis, since it seems intuitive that a sentence could have higher "domain coherence" (i.e. less uniform distribution of domains) than an entire document.

The results show a performance deterioration in moving from documents to sentences, as can be seen in Figure 1(bottom). Again, a possible explanation of this fact is the decrease of the ratio between the number of objects and the number of features that describe these objects (see Experiment 2).

4.1.4 Experiment 4: Augmenting the number of iterations

In our fourth experiment we have reverted to the full RCVI with no feature selection and to using documents instead of sentences, and have tested whether augmenting the number of AdaBoost.MH^{KR} iterations could improve the performance significantly. The results of this experiment have been fairly encouraging, since for $x = 1$ the value of F_1^{μ} increases from .068 (for 500 iterations) to .099 (1500 iterations) to .116 (2500 iterations). This improvement is due to a sharp increase in recall, while precision stays basically constant. We plan to explore this dimension of the problem more thoroughly in our next experiments.

5. CONCLUSION

We have reported an approach to the automatic expansion of domain-specific lexicons by the combination of (i) a dual interpretation of IR-style text indexing theory and (ii) a supervised learning approach. The advantages of our method are that it does not require pre-existing semantic knowledge, and that it is particularly suited to the situation in which several domain-specific lexicons need to be extended in parallel, and to the situation in which no labelled text corpora are available.

Our experiments suggest that the approach is viable, although large margins of improvement still exist: F_1 values are still low, at least if compared to the F_1 values that have been obtained in *text* categorization research on the same corpus [2], so work is still needed in tuning this approach in order to obtain significant categorization performance.

Anyway, the very fact that the F_1 scores are much lower than the ones usually obtained in text categorization by the same kind of "extensional representation + supervised learning" approach proves that term categorization is a harder task than text categorization. Why this is so is not entirely clear, since the metaphor according to which the meaning of a text "coincides" with the terms it contains (a metaphor that has proven so successful in text categorization) is in principle no more powerful or intuitive than the dual metaphor according to which the meaning of a term coincides with the texts it is contained in. We conjecture that the substantial difference in performance between the two cases might be due to the fact that, while in text categorization there often are features with very high discriminative power, this does not seem to be the case for term categorization. For instance, in the Reuters-21578 collection there are categories that can be almost perfectly discriminated by means of a single term (i.e. the term is present in most positive examples and absent in most negative ones); conversely, it seems hardly thinkable that a document (be it in Reuters-21578 or not) might be composed of only and all the terms belonging to a given domain. A more direct proof of this fact is the difference in the values that the information gain function has in term categorization and text categorization: in our term categorization experiments on the full RCVI corpus our highest scoring features had information gain values roughly 20 times lower than the information gain values of the highest scoring features in a Reuters-21578 text categorization experiment we have run separately. Since information gain is a direct measure of the discriminative power of a feature, this result alone indicates how much harder is term categorization than text categorization.

6. REFERENCES

[1] S. Abney, R. E. Schapire, and Y. Singer. Boosting applied to tagging and PP attachment. In *Proceedings of EMNLP-99, 4th Conference on Empirical Methods in Natural Language Processing*, pages 38–45, College Park, MD, 1999.

[2] T. Ault and Y. Yang. kNN, Rocchio and metrics for information filtering at TREC-10. In E. M. Voorhees, editor, *Proceedings of TREC-10, 10th Text Retrieval Conference*, Gaithersburg, US, 2001. National Institute of Standards and Technology, Gaithersburg, US.

[3] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, US, 1998.

[4] B. Magnini and G. Cavaglià. Integrating subject field codes into WordNet. In *Proceedings of LREC-2000, 2nd International Conference on Language Resources and Evaluation*, pages 1413–1418, Athens, GR, 2000.

[5] R. E. Schapire and Y. Singer. BoosTexter: a boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[6] R. E. Schapire, Y. Singer, and A. Singhal. Boosting and Rocchio applied to text filtering. In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 215–223, Melbourne, AU, 1998. ACM Press, New York, US.

[7] P. Schäuble and D. Knaus. The various roles of information structures. In O. Opitz, B. Lausen, and R. Klar, editors, *Proceedings of the 16th Annual Conference of the Gesellschaft für Klassifikation*, pages 282–290, Dortmund, DE, 1992. Published by Springer Verlag, Heidelberg, DE, 1993.

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

[9] F. Sebastiani, A. Sperduti, and N. Valdambrini. An improved boosting algorithm and its application to automated text categorization. In A. Agah, J. Callan, and E. Rundensteiner, editors, *Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management*, pages 78–85, McLean, US, 2000. ACM Press, New York, US.

[10] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In D. H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 412–420, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.