A Text Classifier based on Linguistic Processing

Roberto Basili and **Alessandro Moschitti** and **Maria Teresa Pazienza,**
University of Rome Tor Vergata,
Department of Computer Science, Systems and Production,
00133 Roma (Italy),
{basili,moschitti,pazienza}@info.uniroma2.it

Abstract

In this paper[1] a simple classification model based on a linguistic processing that produces syntactic information, i.e. grammatical categories of words in documents, is described. Moreover an experimental set-up able to dynamically support several tests of different approaches has been realized in order to get large scale empirical evidences. The evidences for this are discussed with comparative evaluation against other alternative more complex models.

1 Definition of the Problem

Thematic text classification (TC) is assuming an increasingly relevant role, given the need of software systems capable of intelligently accessing and using internal or external available information resources. The classification problem can be formulated as follows:

- Given a set of (possibly evolving) user needs expressed as a structure of topics/subtopics classes
- Given a variety of existing examples of these classes (also referred as training data set (Tr)),
- Build a decision function able to upgrade the existing example repository with suitable new incoming texts

Classes ($C = \{C_1, ..., C_n\}$) are used to represents topics/areas of interest. The decision function is asked to map newly incoming documents (d) in one (or more) class(es), according to their content.

The target classes are very heterogeneous according to the different user needs. As an example, in the set of Reuters categories we can find quite specific (e.g. corporate industrial *Ownership Changes* or *Advertising/Promotion*) together with highly general classes (e.g. *Domestic Politics* or *Sport*). It is often the case that documents dealing with specific information are also useful to other (distinct) general topics. The relevance of this information is a matter of judgment depending on the providers of information as well as on the user himself.

Two main approaches to the construction of a non-parametric classifier have been proposed and experimented in literature [Lewis *et al.*, 1996]. *Profile-based* (or linear) classifiers derive a description of each target class (C_i) in terms of a profile, usually a vector of weighted terms. These vectors are extracted from the training documents pre-categorized under C_i. This approach can be referred as *category-centred* classification. Classification is thus the evaluation of similarity between the incoming document d and the different profiles (one for each class). Early profile-based classifier made use of the vector space model ([Salton and Buckley, 1988]) to define similarity, by switching from an information retrieval to a text classification task.

Example-based are other types of classifiers, in which the incoming document d is used as a query against the training data (Tr). Similarity between d and the documents in Tr is evaluated. The categories under which the training documents with the highest similarity are categorized, are considered as promising classification candidates for d. This approach is also referred as *document-centred* categorization. An example in this class is the k-NN algorithm used in the ExpNet system [Yang, 1994].

Main differences among different approaches to TC relate to the representation as well as to the classification issue. The deeper is the information used in the representation (from simple structural, morphological features to semantic information), the more complex are the functionalities needed for extracting them from target documents. It is in this area that natural language processing can give its major contribution.

In this paper a profile-based classifier making use of syntactic information of words is described. It is an extension of existing models as it includes a first level of linguistic information (i.e. part-of-speech tags). In the presented model an original similarity score as well as a novel probabilistic inference method for the selection of suitable classes are also introduced. Section 2 presents the required formal definitions and discuss the original aspects of the method. Section 3 discuss the results of large scale experiments over a traditional test case (i.e. Reuters).

2 A language-sensitive classification model

The major role that linguistic information can play in improving the accuracy of a statistical classification

[1]The research described in this paper has been partially funded under the TREVI ESPRIT project, n. EP23311

model has been repeatedly stressed (e.g. [Lewis and Jones, 1996]). *Normalization*, *Semantic sensitivity* and *Intelligent Clustering* are among the recognized benefits. Note that the rapidly evolving research in NLP made available in the recent years a variety of methods and tools for large scale language processing: part-of-speech (POS) tagging ([Brill, 1992],[Church, 1988]) for example is nowadays effectively used over raw texts with a reasonable accuracy (about 3-4% error rate). Although large scale NL semantic understanding is still an open issue, the application of available linguistic methods to text classification is worth to be systematically studied.

For these reasons, we tested the classification accuracy of a system where a linguistic processor (i.e. a parser) is firstly applied to documents and then a statistical classification model is used to decide. A set of training documents has been used as a basis for the development of the statistical classifier: parsing is first carried out on the learning data and the corresponding linguistic evidence is used for training the classifier. In particular, the adopted parser is CHAOS ([Basili *et al.*, 1998]), a robust parser for Information Extraction that derives syntactically annotated dependency graphs where nodes are POS tagged words and links express the main grammatical relations (e.g. subject, objects). Instead of modeling a document in a vector of words (possibly pruned by stopword lists), the availability of syntactic graphs supports representation in terms of:

- *canonical word forms* or *lemmas*, i.e. morphologically normalized words
- *syntactically categorized words*, i.e. couples `(lemma,POS_tag)` expressing the different grammatical roles of lemmas in the source documents
- *phrasal information*, like sentence fragments expressing relevant concepts (e.g. compound words like *artificial satellites*).
- *facts*, in terms of subgraphs representing verbal phrases, i.e. events specific to the application domain.

In the current tests only the first two forms of linguistic information (i.e. `(lemma,POS_tag)`) have been used, although all of them are made available within the current integrated system. The experiments described in section 3.1 are thus based only on a subset of the available information and aim to shed some light on the study of the impact of NLP in TC. In fact, the quality and degree of improvement due to linguistic features are still under discussion (e.g. [Lewis and Jones, 1996]) as their impact is still to be fully demonstrated.

2.1 A simple profile-based classifier

The adopted model is made essentially of the following distinct phases:

- *Training*, that uses a subset of the available data to develop a synthetic description (i.e. profile) of a given class
- *Evaluation*, that takes a new document and estimates similarity with profiles
- *Classification*, that decides the suitable classes for the incoming document according to the similarity scores.

2.2 The training model

The training method proceeds by

- finding a representation $\vec{d}$ of a document d by means of the set of terms t that appears in d. Weights of those terms are also included in the representation
- finding a representation of a class C_i that summarizes the representation $\vec{d}$ of all documents that are positive instances of C_i (i.e. those d such that $d \in C_i$).

It is worth noticing that the selected representations for classes (C_i) and documents (d) should easily support the modeling of a similarity measure, able to express adherence of a document to a class. Given a generic document $d_h \in C_i$, let $< t, \bar{\omega}_t^h >$ be the couple of the word $t \in d_h$ and its weight $\bar{\omega}_t^h$ expressing the "relevance" that t assumes in d_h. We can represent the document d_h with the following vector

$$\vec{d_h} = << t_1, \bar{\omega}_{t_1}^h >, ..., < t_{k_h}, \bar{\omega}_{t_{k_h}}^h >> \qquad (1)$$

where t_k ($k = 1, ..., k_h$) are all the words in the h-th document.

The definition of the weights $\bar{\omega}_t^h$ is the following:

$$\bar{\omega}_t^h = (log\frac{N}{F_t})^2 f_t^h \qquad (2)$$

where

- N is training set size (i.e. the sum of occurrences of all words in the training set)
- F_t are the occurrences of word t in the training set
- f_t^h are the occurrences of word t in the document d_h

In analogy with the representation of a document, the following vector is used to represent a class C_i:

$$\vec{C_i} = << t_1, \omega_{t_1}^i >, ..., < t_{k_i}, \omega_{t_{k_i}}^i >> \qquad (3)$$

where t_k is a generic word appearing in at least one document of class C_i, i.e. t_k is such that a document $d_h \in C_i$ exists with $t_k \in d_h$. The maximum value of this index (i.e. k_i) clearly depends on the class C_i: some larger classes have significantly higher values for k_i and this has to be taken into account in the definition of the metrics. Weights ω_t^i are defined by

$$\omega_t^i = (log\frac{N}{F_t})^2 F_t^i \qquad (4)$$

where F_t^i is the occurrences of word t in class i.

2.3 The evaluation model

The evaluation phase can be stated as follows:

- Given
  - a document d_h, and its representation $\vec{d_h}$
  - the set of classes $C = \{C_1, ..., C_n\}$
  - a representation of the C's classes, i.e. the vectors $\vec{C_i}$ associated to the classes C_i

- Build, by means of a function $f(d_h)$, the vector of real values $< x_1, ..., x_n >$ in which x_i expresses the strength of the membership, i.e. how much d_h *"belongs to"* C_i:

It is worth noticing that the value x_i depends on both the document d_h and the class C_i. Therefore to emphasize such dependence it will be denoted by s_{ih}. In the vector space model [Salton and Buckley, 1988], membership degree of d_h to C_i is usually estimated by:

$$s_{ih} = cos(\angle(\vec{C_i}, \vec{d_h})) = \frac{\sum_t \omega_t^i \bar{\omega}_t^h}{|\vec{C_i}||\vec{d_h}|} \quad (5)$$

The cosine of the angle between the representation vector of document d_h and class C_i is used as the "membership" score.

As the numerator of the scalar product is zero for each word t that does not appear in the document, the corresponding addends can be omitted. Moreover, the norm of C_i tends to penalize larger classes. Therefore we redefine the C_i norm as follows:

$$s_{ih} = \frac{\sum_{t \in (d_h \cap C_i)} \omega_t^i \bar{\omega}_t^h}{|\vec{C_i}|_{d_h} |\vec{d_h}|_{C_i}} \quad (6)$$

where $|\vec{C_i}|_{d_h}$ is the norm in the subspace of words t that appears in the document d_h. Equation 6 will be hereafter used as the membership score of d_h in class C_i, and is the i-th component of the $f(d_h)$ function.

2.4 Classification via Relative Difference Score

In order to select the suitable classes for a document usually thresholding over s_{ih} is the widely adopted empirical criteria (see [Lewis, 1992] for a comparative evaluation). We defined a thresholding policy based on a probabilistic treatment (empirical estimation from training data) of the *differences between membership scores*. Instead of the s_{ih} scores directly, a stochastic variable m_i, expressing the average difference between the score of the correct (i-th) class and the remaining classes, i.e.

$$m_i = \frac{\sum_{j=1}^{n} s_{hi} - s_{hj}}{n-1} \quad (7)$$

is used to control the decision. For each class, the mean and standard deviation, denoted respectively as $E(m_i)$ and $StdDev(m_i)$ of m_i are estimated over all documents d_h in the training set. Given the vector $f(d_h) =< s_{h1}, ..., s_{hn} >$, we assign d_h to C_i if its corresponding m_i has the following property:

$$m_i > E(m_i) - \alpha_i StdDev(m_i) \quad (8)$$

where each α_i is a threshold (empirically determined to optimize recall and precision over the test data). Hereafter, this kind of inference will be referred as *Relative Difference Score* (RDS). Note that the RDS classification model is better suited to deal with those *odd* documents d_h that are not similar (i.e. have low s_{hi} values for each i) because they are quite "different" from the training documents. In the assumption that target classes represent a closed world, the above method suggests a *"try always to classify"* principle: it is expected to improve significantly the recall of the system keeping satisfactory precision. Details on the evaluation are in Section 3.1.

2.5 Main features of the method

Our model introduces two major differences with respect to the traditional weighting strategy employed in SMART ([Salton, 1991]). First, in equation 2, the *Inverse Word Frequencies* ($IWF = log(\frac{N}{F_t})$) is used in place of IDF. Its meaning is similar to IDF, as both tend to penalize high-frequency (and less meaningful) terms (e.g. *be*, *have*, ...). Another significant difference with respect to SMART is the IWF squaring in equations 2 and 4. In fact, the product $IDF\dot{F}_t^h$ was too biased by the (global) frequency term F_t^h: in order to balance the IWF contribution its square is preferred in eq. 4. A similar adjustment technique is proposed in [Hull, 1994]. The result in our case was a 4% improvement of the breakeven point.

A distinctive feature of our model relates to the thresholding policy (Eq. 8). The tree main approaches to thresholding are *probability-based*, *fixed* and *proportional thresholding*. Lewis has shown [Lewis, 1992] that none of this is a clearly superior policy.

The RDS method we propose produces an improvement of the breakeven point with respect to the policies discussed in [Lewis, 1992]. It is in fact to be seen as an extension of *proportional thresholding* policy as it is estimated over the training data. RDS is independent from the document stream (i.e. the overall set of incoming data) as it applies individually to documents.

RDS is expected to improve (and in fact it does) the system recall, keeping the same precision if compared with other policies. RDS is not influenced by the average membership scores of documents in the training set (it is thus less biased by the training data). RDS does not fix the number of classes (k) to be retained for a document. RDS has been shown more effective with respect to categories with different specificity. We experimentally observed significantly different thresholding rules (suggested by $E(m_i)$ and $StdVar(m_i)$ in Eq. 8) for classes with different specificity. RDS pushes for a sort of *closed world* assumption: any document should be classified in at least one of the target classes. Note that it is also true for other policies. This trend is also coherent with the application scenario where categories represent the exhaustive set of user needs.

Equations 2, 4, 6, together with the inference rule 8, characterize the proposed model. As the model has been tested over a vector representation making use of linguistic information (i.e. POS tags and lemmas), it will be referred hereafter as the NL/RDS method. The employed syntactic categories for representing documents and classes are only verbs, nouns and adjectives. The result is a method where a number of words (i.e. functional words like prepositions or conjunctions) are removed from the representation.

3 Experimental Evaluation and Discussion

The above model has been employed within the TREVI (Text Retrieval and Enrichment for Vital Information) system. TREVI , is a system for Intelligent Text Retrieval and Enrichment. TREVI ([Basili *et al.*, August 1998])is a distributed system for text classification and enrichment, designed and developed by a European consortium under the TREVI ESPRIT project EP23311. Reuters is a member of the Consortium and has been used as a main "*User Case*" for the released prototype. A specific subset of the classes (judged particularly meaningful to the Reuters customer service) is currently managed by the prototype. It includes 30 classes in different levels of the Reuters classification tree. For these categorization task, we received $29,026$ manually classified documents. Cross validation has been run using 90% of the overall data as training and testing on the remaining portion. Precision, Recall and the Breakeven point (when significant for comparative purpose) have been used as performance indexes.

3.1 Models and Experiments

In Section 2.5 we outlined that the NL/RDS classifier introduces three major changes with respect to the SMART model: the use of syntactic categories for lemma, a weighting factor (squared IWF) and the RDS classification rule. For these reasons we used SMART as a baseline of our model. Two different tests have been run to evaluate the performance of the presented model focusing on the newly introduced features.

In Table 1 the breakeven points of the NL/RDS classifier with respect to the SMART model is reported. Both statistical models have been run over the output of the linguistic processor (i.e. POS tagged lemma found detected in document). The SMART model has been run using two different classification rules: SMART+k-best adopts the fixed thresholding policy, while SMART+RDS has been implemented by the relative difference score (Eq. 8). The only difference between NL/RDS and SMART+RDS is the weighting factor (see Eq. 2 and 4).

Table 1: Classification Accuracy

	SMART+k-best	SMART+RDS	NL/RDS
Breakeven Point	63%	72%	76%

We can observe that the squared IWF improves performance of 4% on the breakeven point, due to the square operation that gives more relevance to the IWF (first weight factor) than to term frequency (second weighting factor). IWF is used in place of IDF because is more easily computed from the corpus.

The first column in Table 1 reports the SMART+k-best model. Comparison with the second column suggests that the RDS classification rule brings an increment of about 9%.

It is worth noticing that, as the RDS rule depends on the different classes C_i, the exact measure of the breakeven point requires a $30 \times M$ experiment matrix: M is the number of samples required to detect the breakeven. Values in Table 1 have been derived by polynomial approximation over a smaller number of runs.

Finally, Table 1 suggests that the NL/RDS method, whose implementation is very simple if compared to other classifiers (e.g. [Yang, 1994]) has a relatively good performance. In order to better evaluate it, a second test has been carried out focusing on the effects of syntactic information on the classification accuracy.

In Table 2 the NL/RDS model is run in four different ways. RDS is the model based on just lemmas, without taking into account the syntactic categories. NL/RDS+*Adj* and NL/RDS-*Adj* are runs where only adjectives and, respectively, only nouns and verbs are used. The second column NL/RDS reports precision and recall of the overall method without any missing syntactic information. According to the TREVI requirements, the empirical threshold (α_i) have been selected in order to optimize recall rather than precision. False negatives are in fact more dangerous than false positives for the customers of an information provider.

Table 2: Syntactic Information vs. Classification Accuracy

	RDS	NL/RDS	NL/RDS +*Adj*	NL/RDS -*Adj*
Rec.	83.02%	83.70%	59.18%	83.01%
Prec.	70.56%	70.86%	51.89%	70.83%

The result shows that syntactic categories seem not to influence too much the accuracy of the system (only .7% improvement on the recall and .3% on precision). In order to study this phenomenon the distribution of lemmas and their syntactic categories has been studied in the training data. Results are reported in Table 3. N, V and A describe the number of `(lemma,POS_tag)` couples where `POS_tag` is one of the classes noun, verb and adjective respectively. *Total* is the number of couples and *Lemma* is the number of different lemmas in the training corpus, obtained by disregarding the POS information. As a result the number of really ambiguous lemmas (e.g. nouns that appear in the training data also as verbs or adjectives) is only $4,089$.

Table 3: Syntactic Information in the training data

N	V	A	Total
24,428	8,869	7,861	41,158

Lemmas	Ambiguous Lemmas
37,069	4,089

However, the role of syntactic information is outlined by the comparison of the second column in Table 2 with the third and fourth. The performances of the model that uses only adjectives (i.e. NL/RDS+*Adj*) is very poor: this suggests that different syntactic categories bring very different contributions to the classification accuracy. When adjectives are not used for classification the precision and the recall are very near to the NL/RDS case. Note that the breakeven point of NL/RDS-*Adj* (in Table 1) is near to the value of the NL/RDS model, as a further evidence of the low contribution of this syntactic class of words. Intuitively adjectives do not capture relevant information, as they appear uniformly distributed among all the texts. The result in the third column of Table 2 is a dramatic reduction of performance (about 20%). Note, however, that adjectives are only the 19% of the corpus lemmas.

4 Conclusions and Open Problems

In this paper, a simple and efficient profile-based classifier has been described. The main features of the proposed approach are the use of linguistic information (i.e. lemmatization and part-of-speech (POS) tagging) as source information for document and class representation, a specific weighting model and an original technique for the classification inference (*Relative Difference Score*, RDS). The experimentation has been carried out within an integrated prototype (called TREVI) over the Reuters classification case. The overall breakeven performance of the model is 76%. The weighting policy and the classification rule have been shown to produce an improvement of about 0.13 in the breakeven point with respect to the baseline SMART system. Data show also that RDS classification alone produces an improvement of about 9% within the simple SMART model. Tests on the role of syntactic information show a little improvement in the overall accuracy (only +0.7% on the recall value when using POS categories). Although this result is not striking a significant difference is measured in the role of different syntactic categories on accuracy. This suggests that a better use of them (e.g. a more complex membership scoring function using a combination of functions over different categories) could result in a stronger impact on accuracy. Furthermore the availability of parsing information support a variety of extensions (e.g. from syntax-driven term clustering to semantic indexing).

For its simplicity and efficiency (complexity is $O(1)$) in the document classification phase, the model is promising from a user-oriented point of view: portability and robustness are key features for automatic text classification on large scale.

Further experimentations is on going to test the effects of the RDS technique within other classification models (e.g. the k-NN system). Although we do not expect a corresponding 9% improvement (as for SMART), even a little improvement for an already effective system would be relevant.

The NL/RDS model is the first step towards a truly language sensitive system making full use of the linguistic information available after parsing. Although questions about the role and effectiveness of grammatical information in classification have not been answered during this first phase of experimentation, (linguistically) richer representation forms and better statistical inference can be supported in the current integrated system and this will be a relevant research area for our future work.

Acknowledgements

The author would like to thank Michele Vindigni and Massimo Di Nanni, of the Department of Computer Science of our University as well as Luigi Mazzucchelli and Maria Vittoria Marabello of the Itaca s.r.l. group, for having made available the TREVI categorization system.

References

[Basili *et al.*, 1998] Roberto Basili, Maria Teresa Pazienza, and Fabio Massimo Zanzotto. Efficient parsing for information extraction. In *Proc. of the ECAI98*, Brighton, UK, 1998.

[Basili *et al.*, August 1998] R. Basili, M. Di Nanni, L. Mazzucchelli, M.V. Marabello, and M.T. Pazienza. Nlp for text classification: the trevi experience. In *Proceedings of the Second International Conference on Natural Language Processing and Industrial Applications, Universite' de Moncton, New Brunswick (Canada)*, August 1998.

[Brill, 1992] E. Brill. A simple rule-based part of speech tagger. In *Proc. of the Third Applied Natural Language Processing, Povo, Trento, Italy*, 1992.

[Church, 1988] K. A. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proc. of Second Conference on Applied Natural Language Processing*, 1988.

[Hull, 1994] David Hull. Improving text retrieval for the routing problem using latent semantic indexing. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 282–291, Dublin, IE, 1994.

[Lewis and Jones, 1996] David Lewis and Karen Sparck Jones. Natural language processing for information retrieval. *Communications of ACM*, 39:92–101, 1996.

[Lewis *et al.*, 1996] David D. Lewis, Robert E. Schapiro, James P. Callan, and Ron Papka. Training algorithms for linear text classifiers. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*, pages 298–306, Zürich, CH, 1996.

[Lewis, 1992] David D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval*, pages 37–50, Kobenhavn, DK, 1992.

[Salton and Buckley, 1988] G: Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[Salton, 1991] G. Salton. Development in automatic text retrieval. *Science*, 253:974–980, 1991.

[Yang, 1994] Yiming Yang. Expert network: effective and efficient learning from human decisions in text categorisation and retrieval. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 13–22, Dublin, IE, 1994.