A NOVEL SUPERVISED TEXT CLASSIFIER FROM A SMALL TRAINING SET

Fabio Clarizia, Francesco Colace, Massimo De Santo, Luca Greco and Paolo Napoletano
Department of Electronics and Computer Engineering, University of Salerno
Via Ponte Don Melillo 1, 84084 Fisciano, Italy

Keywords: Text classification, Term extraction, Probabilistic topic model.

Abstract: Text classification methods have been evaluated on supervised classification tasks of large datasets showing high accuracy. Nevertheless, due to the fact that these classifiers, to obtain a good performance on a test set, need to learn from many examples, some difficulties may be found when they are employed in real contexts. In fact, most users of a practical system do not want to carry out labeling tasks for a long time only to obtain a better level of accuracy. They obviously prefer algorithms that have high accuracy, but do not require a large amount of manual labeling tasks.
In this paper we propose a new supervised method for single-label text classification, based on a mixed *Graph of Terms*, that is capable of achieving a good performance, in term of accuracy, when the size of the training set is 1% of the original. The mixed *Graph of Terms* can be automatically extracted from a set of documents following a kind of *term clustering* technique weighted by the probabilistic topic model. The method has been tested on the top 10 classes of the ModApte split from the Reuters-21578 dataset and learnt on 1% of the original training set. Results have confirmed the discriminative property of the graph and have confirmed that the proposed method is comparable with existing methods learnt on the whole training set.

1 INTRODUCTION

The problem of *text classification* has been extensively treated in literature where metrics and measures of performance have been reported (Christopher D. Manning and Schtze, 2009), (Sebastiani, 2002), (Lewis et al., 2004). All the existing techniques have been demonstrated to achieve high accuracy (mainly assessed through the F_1 measure) when employed in supervised classification tasks of large datasets.

Nevertheless, it has been found that only 100 documents could be hand-labeled in 90 minutes and in this case the accuracy of classifiers (amongst which we find Support Vector Machine based methods), learnt from this reduced training set, could be around 30%. This makes, most times, a classifier unfeasible in a real context. In fact, most users of a practical system do not want to carry out labeling tasks for a long time only to obtain better level of accuracy. They obviously prefer algorithms that have high accuracy, but do not require a large amount of manual labeling tasks (McCallum et al., 1999)(Ko and Seo, 2009). As a consequence, we can affirm that, in several application fields we need algorithms to be fast and with a good performance.

Here we propose a linear single label supervised classifier that is capable, based on a *vector of features* represented through a mixed *Graph of Terms* ($m\mathcal{GT}$), of achieving a good performance, in terms of accuracy, when the size of the training set is 1% of the original and comparable to the performances achieved by existing methods learnt on the whole training set.

The *vector of features* can be automatically extracted from a set of documents following a kind of *term clustering* technique weighted by the probabilistic topic model. The graph learning procedure is composed of two stages and leads us to a two level representation. Firstly, we group terms with a high degree of pairwise semantic relatedness so obtaining several groups, each of them represented by a cloud of *words* and their respective centroids that we call *concepts*. In this way, we obtain the lowest level, namely the *word level*. Later, we compute the second level, namely the *conceptual level*, by inferring semantic relatedness between centroids, and so between *concepts*.

To confirm the discriminative property of the graph we have evaluated the performance through a comparison between our term extraction methodol-

Clarizia F., Colace F., De Santo M., Greco L. and Napoletano P..
A NOVEL SUPERVISED TEXT CLASSIFIER FROM A SMALL TRAINING SET.
DOI: 10.5220/0003661105370545
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (SSTM-2011), pages 537-545
ISBN: 978-989-8425-79-9

ogy and a term selection methodology which considers the *vector of features* formed of only the list of concepts and words composing the graph and so where relations have not been considered. The results, obtained on the top 10 classes of the ModApte split from the Reuters-21578 dataset, show that our method, independently of the topic, is capable of achieving a better performance.

2 PROBLEM DEFINITION

Following the definition introduced by (Sebastiani, 2002), a supervised *Text Classifier* may be formalized as the task of approximating the unknown target function $\Phi : \mathcal{D} \times \mathcal{C} \rightarrow \{T,F\}$ (namely the expert) by means of a function $\hat{\Phi} : \mathcal{D} \times \mathcal{C} \rightarrow \{T,F\}$ called the *classifier*, where $\mathcal{C} = \{\mathbf{c}_1,...,\mathbf{c}_{|\mathcal{C}|}\}$ is a predefined set of *categories* and $\mathcal{D}$ is a (possibly infinite) set of *documents*. If $\Phi(\mathbf{d}_j,\mathbf{c}_i) = T$, then $\mathbf{d}_j$ is called a positive example (or a member) of $\mathbf{c}_i$, while if $\Phi(\mathbf{d}_j,\mathbf{c}_i) = F$ it is called a negative example of $\mathbf{c}_i$. Moreover, the categories are just symbolic labels: no additional knowledge (of a procedural or declarative nature) of their meaning is usually available, and it is often the case that no metadata (such as e.g. publication date, document type, publication source) is available either. In these cases, classification must be accomplished only on the basis of knowledge extracted from the documents themselves.

In practice we consider an initial corpus $\Omega = \{\mathbf{d}_1,\ldots,\mathbf{d}_{|\Omega|}\} \subset \mathcal{D}$ of documents pre-classified under $\mathcal{C} = \{\mathbf{c}_1,...,\mathbf{c}_{|\mathcal{C}|}\}$. The values of the total function Φ are known for every pair $(\mathbf{d}_j,\mathbf{c}_i) \in \Omega \times \mathcal{C}$.

We consider the initial corpus to be split into two sets, not necessarily of equal size:

1. *training* set $Tr = \{\mathbf{d}_1,\ldots,\mathbf{d}_{|Tr|}\}$. The classifier Φ for categories is inductively built by observing the characteristics of these documents;
2. *test* set $Te = \{\mathbf{d}_{|Tr|+1},\ldots,\mathbf{d}_{|\Omega|}\}$, used for testing the effectiveness of the classifiers.

We assume that the documents in Te cannot participate in any way in the inductive construction of the classifiers.

Here we consider the case of *single-label* classification, also called *binary*, in which, given a category $\mathbf{c}_i$, each $\mathbf{d}_j \in \mathcal{D}$ must be assigned either to $\mathbf{c}_i$ or to its complement $\bar{c}_i$. In fact, it has been demonstrated that the binary case is more general than the multi-label (Sebastiani, 2002; Christopher D. Manning and Schtze, 2009). It means that we consider the classification under $\mathcal{C} = \{\mathbf{c}_1,...,\mathbf{c}_{|\mathcal{C}|}\}$ as consisting of $|\mathcal{C}|$ independent problems of classifying the documents in $\mathcal{D}$ under a given category $\mathbf{c}_i$, and so we have $\hat{\phi}_i$, for $i = 1,\ldots,|\mathcal{C}|$, classifiers. As a consequence, the whole problem in this case is to approximate the set of function $\Phi = \{\phi_1,\ldots,\phi_{|\mathcal{C}|}\}$ with the set of $|\mathcal{C}|$ classifiers $\hat{\Phi} = \{\hat{\phi}_1,\ldots,\hat{\phi}_{|\mathcal{C}|}\}$.

2.1 Data Preparation

Texts cannot be directly interpreted by a classifier or by a classifier-building algorithm. Because of this, an indexing procedure that maps a text $\mathbf{d}_j$ into a compact representation of its content needs to be uniformly applied to the training and test documents. In fact, each document is represented as a vector of term *weights* $\mathbf{d}_i = \{w_{1j},\ldots,w_{|\mathcal{T}|j}\}$, where $\mathcal{T}$ is the set of *terms* (sometimes called *features*) that occur at least once in at least one document of Tr, and $0 \leq w_{kj} \leq 1$ represents how much term t_k contributes to a semantics of document $\mathbf{d}_j$. A typical choice is to identify terms with words, that is the *bags of words* assumption, and in this case $t_k = v_k$, where v_k is one of the words of a vocabulary. Usually to determine the weight w_{kj} of term t_k in a document $\mathbf{d}_j$, the standard *tfidf* function is used , defined as:

$$tfidf(t_k,\mathbf{d}_j) = N(t_k,\mathbf{d}_j) \cdot \log \frac{|Tr|}{N_{Tr(t_k)}} \qquad (1)$$

where $N(t_k,\mathbf{d}_j)$ denotes the number of times t_k occurs in $\mathbf{d}_j$, and $N_{Tr(t_k)}$ denotes the document frequency of term t_k, i.e. the number of documents in Tr in which t_k occurs.

In order for the weights to fall in the $[0,1]$ interval and for the documents to be represented by vectors of equal length, the weights resulting from $tfidf$ are often normalized by cosine normalization, given by:

$$w_{kj} = \frac{tfidf(t_k,\mathbf{d}_j)}{\sqrt{\sum_{s=1}^{|\mathcal{T}|}(tfidf(t_s,\mathbf{d}_j))^2}} \qquad (2)$$

Before indexing, we have performed the removal of function words (i.e. topic-neutral words such as articles, prepositions, conjunctions, etc.) and we have performed the stemming procedure[1] (i.e. grouping words that share the same morphological root).

2.2 Data Reduction

Usually, due to computational problems and to the problem of *overfitting*, a *dimensional reduction* of the dataset is applied. We distinguish between *local* and

[1] Stemming has sometimes been reported to hurt effectiveness, the recent tendency is to adopt it, as it reduces both the dimensionality and the stochastic dependence between terms.

global methods, if we apply the reduction to each document or to the whole repository respectively. Another distinction may be considered, that is between the *term selection* and *term extraction* reduction techniques (Sebastiani, 2002; Christopher D. Manning and Schtze, 2009):

1. *Term Selection*: $\mathcal{T}'$ is a subset of $\mathcal{T}$. Examples of this are methods that consider the selection of only the terms that occur in the highest number of documents, or the selection depending on the observation of information-theoretic functions, among which we find the *DIA association factor*, *chi-square*, *NGL coefficient*, *information gain*, *mutual information*, *odds ratio*, *relevancy score*, *GSS coefficient* and others.
2. *Term Extraction*: the terms in $\mathcal{T}'$ are not of the same type as the terms in $\mathcal{T}$ (e.g. if the terms in $\mathcal{T}$ are words, the terms in $\mathcal{T}'$ may not be words at all), but are obtained by combinations or transformations of the original ones. Examples of this are methods that consider generating, from the original, a set of "synthetic" terms that maximize effectiveness based on *term clustering*, *latent semantic analysis*, *latent dirichlet allocation* and others.

In this paper we use a *global* method for the *term extraction* based on a kind of *Term Clustering* technique (Sebastiani, 2002) weighted by the *Latent Dirichlet Allocation* (Blei et al., 2003) implemented as the *Probabilistic Topic Model* (Griffiths et al., 2007). Previous works (Berkhin, 2006; Noam and Naftali, 2001) have confirmed the potential of supervised clustering methods for term extraction.

3 PROPOSED TERM EXTRACTION METHOD

More precisely the term extraction procedure is composed of two stages and leads us to a two level representation.

Firstly, we group terms with a high degree of pairwise semantic relatedness so obtaining several groups, each of them represented by a cloud of *words* and their respective centroids that we call *concepts* (see Fig. 1(b)). In this way we obtain the lowest level, namely the *word level*. More formally, each concept r_i can be defined as a rooted graph of words v_s and a set of links weighted by ρ_{is} (see Fig. 1(b)). The weight ρ_{is} can measure how far a word is related to a concept, or in other words how much we need such a word to specify that concept. We can consider such a weight as a probability: $\rho_{is} = P(r_i|v_s)$. The probability of the concept given a parameter μ, which we call $\mathbf{c}_i$, is defined as the factorisation of ρ_{is}

$$P(r_i|\{v_1,\cdots,v_{V_\mu}\}) = \frac{1}{Z_C}\prod_{s\in S_\mu}\rho_{is} \qquad (3)$$

where $Z_C = \sum_C \prod_{s\in S_\mu}\rho_{is}$ is a normalisation constant, V_μ is the number of words defining the concept, such a number depending on the parameter μ.

After, we compute the second level, namely the *conceptual level*, by inferring semantic relatedness between centroids, and so *concepts* (see Fig. 1(a)). More formally, let us define a *Graph of Concepts* as a triple $\mathcal{G}_\mathcal{R} = \langle N, E, R\rangle$ where N is a finite set of nodes, E is a set of edges weighted by ψ_{ij} on N, such that $\langle N, E\rangle$ is an a-directed graph (see Fig. 1(a)), and R is a finite set of concepts, such that for any node $n_i \in N$ there is one and only one concept $r_i \in R$. The weight ψ_{ij} can be considered as the degree of semantic correlation between two concepts r_i *is-related*$_{\psi_{ij}}$*-to* r_j and it can be considered as a probability: $\psi_{ij} = P(r_i, r_j)$. The probability of $\mathcal{G}_\mathcal{R}$ given a parameter ν can be written as the joint probability between all the concepts. By following the theory on the factorisation of undirected graph, we can consider such a joint probability as a product of functions where each of this can be considered as the weight ψ_{ij}. We have

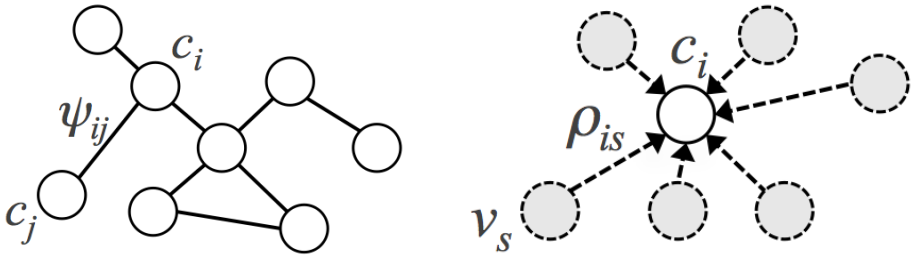
$$P(\mathcal{G}_\mathcal{R}|\nu) = P(r_1,\cdots,r_H|\nu) = \frac{1}{Z}\prod_{(i,j)\in E_\nu}\psi_{ij} \qquad (4)$$

where H is the number of concepts, $Z = \sum_{\mathcal{G}_\mathcal{R}}\prod_{(i,j)\in E_\nu}\psi_{ij}$ is a normalisation constant and the parameter ν can be used to modulate the number of edges of the graph.

The resulting structure is a mixed *Graph of Terms* ($m\mathcal{GT}$) composed of such two levels of information, the *conceptual level* and the *word level*, see Fig. 2. A $m\mathcal{GT}$ is defined by the probability $P(\mathcal{G}_\mathcal{R}|\nu)$, which defines a graph of connected H concepts and the number of edges depends on ν, and by H probabilities of the concepts, $P(r_i|\{v_1,\cdots,v_{V_{\mu_i}}\})$, where the number of edges depends on μ_i. Once each ψ_{ij} and ρ_{is} is known (*Relations Learning*), to determine the final graph we need to compute the appropriate set of parameters $\Lambda = (H, \nu, \mu_1, \cdots, \mu_H)$ (*Parameters Learning*), which establishes the final shape of the graph, that is the number of pairs and the number of both words and concepts.

3.1 Relations Learning

We consider each concept as lexically represented by a word, then we have that $\rho_{is} = P(r_i|v_s) = P(v_i|v_s)$



(a) (b)

Figure 1: 1(a) Theoretical representation of the conceptual level. 1(b) Graphical representation of the word level.

and $\psi_{ij} = P(r_i, r_j) = P(v_i, v_j)$. As a result , all the relations of the $m\mathcal{GT}$ can be represented by $P(v_i, v_j)$ $\forall i, j \in V$ which can be considered as a *word association problem* and so it can be solved through a smoothed version of the generative model introduced in (Blei et al., 2003) called latent Dirichlet allocation, which makes use of Gibbs sampling (Griffiths et al., 2007)[2].

Furthermore, it is quite important to make clear that the mixed *Graph of Terms* can not be considered as a co-occurrence matrix. In fact, the core of the graph is the probability $P(v_i, v_j)$, which we compute through the probabilistic topic model and particularly thanks to the word association problem, that solves the probability $P(v_i|v_j)$. In the topic model, the word association can be considered as a problem of prediction: given that a cue is presented, which new words might occur next in that context? It means that the model does not take into account the fact that two words occur in the same document, but that they occur in the same document when a specific topic is assigned to the document itself (Griffiths et al., 2007), in fact $P(v_i|v_j)$ is the result of a sum over all the topics.

3.2 Parameters Learning

Once each ψ_{ij} and ρ_{is} is known, we have to find a value for the parameter H, which establishes the number of concepts, a value for ν and finally values for μ_i ,$\forall i \in [1, \cdots, H]$. As a consequence, we have $H+2$ parameters which modulate the shape of the graph. If we let the parameters assuming different values, we can observe different graph $m\mathcal{GT}_t$ for each set of parameters, $\Lambda_t = (H, \nu, \mu_1, \cdots, \mu_H)_t$ extracted from the same set of documents, where t is representative of different parameter values.

As we have already discussed, term extraction attempts to generate, from the original set $\mathcal{T}$, a set $\mathcal{T}' \ll \mathcal{T}$ of "synthetic" terms that maximize effectiveness. In this case each "synthetic' term is represented by a pair of related words while the semantic relatedness between pairs, of both the conceptual and the word level, namely ψ_{ij} and ρ_{is}, that we can simply call *boost* of the term k, b_k, gives a degree of relevance to each pair. In practice, we have that each term $t_k = (v_i, v_j)$, that is not the simple bags of words assumption, and w_{kj} being the weight calculated thanks to the $tfidf$ model applied to the pairs represented through t_k, and with the addition of the *boost* b_k. Formally we have

$$w_{kj} = \frac{tfidf(t_k, \mathbf{d}_j) \cdot b_k}{\sqrt{\sum_{s=1}^{|\mathcal{T}_p|} (tfidf(t_s, \mathbf{d}_j) \cdot b_k)^2}} \quad (5)$$

Note that the boost, due to the fact that is a probability factor, is such that $0 \leq b_k \leq 1$. Moreover $|\mathcal{T}_p|$ is the number of pairs of the $m\mathcal{GT}$, considered as composed of all possible combinations of words of the initial vocabulary, that has cardinality $|\mathcal{T}|$.

The scope of this term extraction is to reduce the set $|\mathcal{T}_p|$ to a smaller set $|\mathcal{T}'_p|$, such that the corresponding set of words, composing the pairs belonging to the reduced set $|\mathcal{T}'_p|$, has dimension $|\mathcal{T}'| \ll |\mathcal{T}|$. A way of reducing the set of pairs is to change the set of parameters $\Lambda_t = (H, \nu, \mu_1, \cdots, \mu_H)_t$ until we have maximised the effectiveness, with the condition that the cardinality of the set of pairs is such that $|\mathcal{T}'_p| \ll |\mathcal{T}_p|$.

A way of saying that a $m\mathcal{GT}$, given the parameters, is the best possible for that set of documents is to demonstrate that it produces the maximum score attainable for each of the documents when the same graph is used as a knowledge base for classify a set containing just those documents which have fed the $m\mathcal{GT}$ builder, that is the training set Tr. The result of the parameter learning procedure is an explict *profile*, that is the best $m\mathcal{GT}$, namely the *classifier* $\mathbf{c}_i = \{w_{1i}, \ldots, w_{|\mathcal{T}'_p|i}\}$, belonging to the same $|\mathcal{T}'_p|$-dimensional space in which documents are also represented. In this case we have a linear classifier that,

[2]The authors reported the formulation that brings from the LDA to $P(v_i, v_j)$ in a paper that can not be cited due to the blind review.

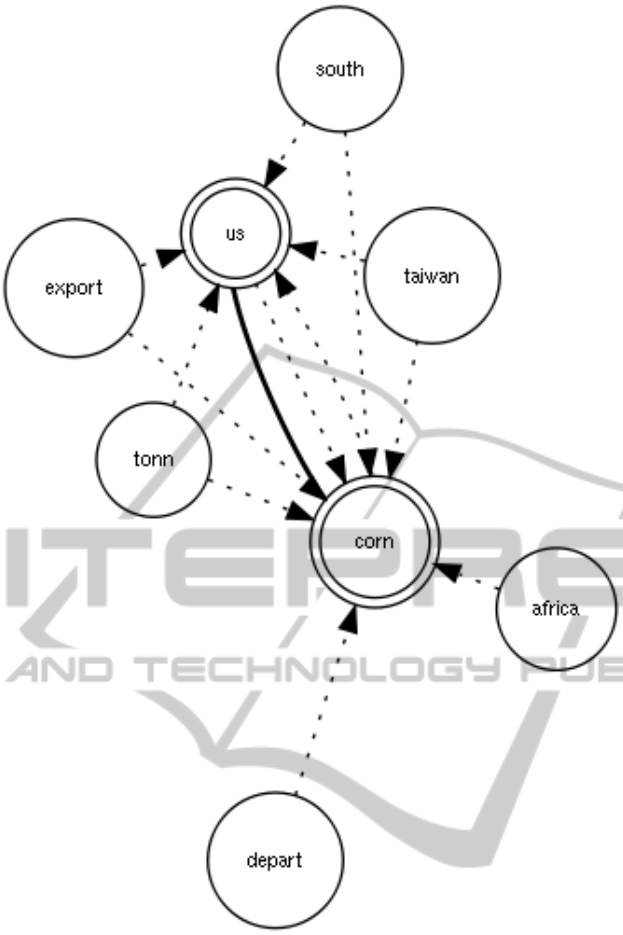


Figure 2: Part of the $m\mathcal{GT}$ learnt on the topic "Corn". We have 2 concepts (double circle) and 6 words (single circle).

thanks to the Vector Space Model theory, measure the degree of relatedness by computing (when both classifier and document weights are cosine normalized) the cosine similarity:

$$\mathcal{S}(\mathbf{c}_i, \mathbf{d}_j) = \frac{\sum_{k=1}^{|\mathcal{T}'_p|} w_{ki} \cdot w_{jk}}{\sqrt{\sum_{k=1}^{|\mathcal{T}'_p|} w_{ki}^2} \cdot \sqrt{\sum_{k=1}^{|\mathcal{T}'_p|} w_{kj}^2}} \qquad (6)$$

By performing a classification task that uses the current graph $m\mathcal{GT}_t$, represented through the set of the classifier weights, on the same repository Tr, we obtain a score for each document $\mathbf{d}_j$ and then we have $\mathbf{S}_t = \{\mathcal{S}(\mathbf{c}_i, \mathbf{d}_1), \cdots, \mathcal{S}(\mathbf{c}_i, \mathbf{d}_{|Tr|})\}_t$, where each of them depends on the set Λ_t. To compute the best value of Λ we can maximise the score value for each documents, which means that we are looking for the graph which best describes each document of the repository from which it has been learnt. It should be noted that such an optimisation maximises at same time all $|Tr|$ elements of $\mathbf{S}_t$. Alternatively, in order to reduce the number of the objectives to being optimised, we can contemporary maximise the mean value of the scores and minimise their standard deviation, which turns a multi-objectives problem into a two-objectives one. Additionally, we can reformulate the latter problem by means a linear combination of its objectives, thus obtaining a single objective function, i.e., *Fitness* ($\mathcal{F}$), which depends on Λ_t, $\mathcal{F}(\Lambda_t) = E_m[\mathcal{S}(\mathbf{q}_t, \mathbf{w}_m)] - \sigma_m[\mathcal{S}(\mathbf{q}_t, \mathbf{w}_m)]$, where E_m is the mean value of all element of $\mathbf{S}_t$ and σ_m being the standard deviation. By summing up, the parameters learning procedure is represented as follows, $\Lambda^* = \underset{t}{\text{argmax}}\{\mathcal{F}(\Lambda_t)\}$. We will see in the next section how we have performed the optimisation phase.

3.2.1 Optimisation Procedure

The fitness function depends on $H+2$ parameters, hence the space of possible solutions could grow exponentially. Due to the fact that we would not have small or too big graph (we wish that $|\mathcal{T}'_p| \ll |\mathcal{T}_p|$ which is equal to say that $|\mathcal{T}'| \ll |\mathcal{T}|$), we suppose

that the number H of concepts can vary from a minimum of 5 to a maximum of 20, and considering that it is an integer number we conclude that the number of possible values for H is 15. We have empirically chosen these values for H considering that we wish to have at most $|\mathcal{T}'_p| \approx 300$[3].

ψ_{ij} and ρ_{is} are probabilities, and so real value, we have that $\nu \in [0,1]$ and each $\mu_i \in [0,1]$. It means that if we use a step of 1% to explore the entire set $[0,1]$, then we have 100 possible values for ν and 100 for each μ_i, which makes $100 \times 100 \times H \times 15$ possible values of Λ, that is $750,000$ for $H = 5$ and $3,000,000$ for $H = 20$. To limit such a space we can reduce the numbers of parameters, for instance we can consider $\mu_i = \mu$, $\forall i \in [1, \cdots, H]$ and so obtaining $150,000$, independently of H, possible values of Λ.

Searching for the best solution is still not easy and it does not provide an accurate solution, because of the big number of possible values and due to the linear exploration strategy of the set $[0,1]$ we are employing. In fact, by analysing how the values of ψ_{ij} and ρ_{is} are distributed along the set $[0,1]$, we note that they are not uniformly distributed. It means that many values of ψ_{ij} and ρ_{is} are likely closer than 1% with the consequence that if the thresholds ν and μ are chosen thanks to that linear exploration then many values will be treated in the same way. To solve this problem one can think to reduce the step from 1% to 0.1% and so obtaining more accuracy in the exploration of the set $[0,1]$. The problem in this case is that the space of solution can grow exponentially, and so this way is not feasible. Another way to reduce the space can be the application of a clustering methods, like the *K-means* algorithm, to all ψ_{ij} and ρ_{is} values (Bishop, 2006). In this way we can have a space of possible values extracted by a no-uniform procedure directly adapted to the real numbers and not to the set which the numbers belong to. Following this approach and choosing for instance 10 classes of values for ν and μ, we obtain that the space of possible Λ is $10 \times 10 \times 15$, that is $1,500$. As a consequence, the optimum solution can be exactly obtained after the exploration of the entire space of solutions. This reduction allows us to compute a $m\mathcal{GT}$ from a repository composed of few documents in a reasonable time, for instance for 10 documents it takes about 30 seconds with a Mac OS X based computer and a 2.66 GHz Intel Core i7 CPU and a 8GB RAM. Otherwise we need an algorithm based on a random search procedure in big solution spaces, for instance Evolutionary Algorithms would be suitable for this purpose, which can be very slow.

[3]This number is usually employed in the case the Support Vector Machine, which have demonstrated to be one of the best.

4 INDUCTIVE CONSTRUCTION OF THE CLASSIFIER

The inductive construction of a ranking classifier for category $\mathbf{c}_i \in \mathcal{C}$ usually consists in the definition of a function $CSV_i : \mathcal{D} \rightarrow [0,1]$ that, given a document $\mathbf{d}_j$, returns a *categorization status value* ($CSV_i(\mathbf{d}_j)$) for it, i.e. a number between 0 and 1 that, represents the evidence for the fact that $d_j \in \mathbf{c}_i$, or in other words it is a measure of vector closeness in $|\mathcal{T}'_p|$-dimensional space. Following this criterion each documents is then ranked according to its CSV_i value, and so the system works as a document-ranking text classifier, namely a "soft" decision based classifier. As we have discussed in previous sections we need a binary classifier, also known as "hard" classifier, that is capable of assign to each document a value T or F to measure the vector closeness. A way turn a soft classifier in a hard one is to define a threshold γ_i such that $CSV_i(\mathbf{d}_j) \geq \gamma_i$ is interpreted as T while $CSV_i(\mathbf{d}_j) \leq \gamma_i$ is interpreted as F. We have adopted an experimental method, that is the *CSV thresholding* (Sebastiani, 2002), which consists in testing different values for γ_i on a sub set of the training set (the *validation* set) and choosing the value which maximizes effectiveness. Different γ_i's have been chosen for the different $\mathbf{c}_i$'s.

Table 1: $m\mathcal{GT}$ for the topic *Corn*. (see Fig. 2).

Conceptual Level		
Concept i	**Concept** j	**Relation Factor** (ψ_{ij})
corn	us	4,0
...	...	...
Word Level		
Concept i	**Word** s	**Relation Factor** (ρ_{is})
corn	south	2.0
corn	us	1.96
corn	export	1.69
corn	africa	1.0
...	...	...
us	south	1.17
us	taiwan	1.0
...	...	...

5 EVALUATION

We have considered a classic text classification problem performed on the Reuters-21578 repository. This is a collection of 21,578 newswire articles, originally collected and labeled by Carnegie Group, Inc. and Reuters, Ltd.. The articles are assigned classes from a set of 118 topic categories. A document may be

Table 2: Distribution of the ModApte split (columns "train" and "test"). Distribution of the training set employed by the proposed method ("mGTtrain" column).

class	mGTtrain	train	test
earn	29	2877	1087
acq	17	1650	719
money-fx	6	538	179
grain	5	433	149
crude	4	389	189
trade	4	369	119
interest	4	347	131
ship	2	197	89
wheat	3	212	71
corn	2	182	56
total	**76**	**7194**	**2249**

assigned several classes or none, but the commonest case is single assignment (documents with at least one class received an average of 1.24 classes). For this task we have used the ModApte split which includes only documents that were viewed and assessed by a human indexer, and comprises 9,603 training documents and 3,299 test documents. The distribution of documents in classes is very uneven and we have evaluated the system on only documents in the 10 largest classes, in table 2 the distribution of the ten largest classes is reported (Christopher D. Manning and Schtze, 2009)[4].

As discussed before we have considered the *any-of problem* and so we have learnt 10 two-class classifiers, one for each class, where the two-class classifier for class c is the classifier for the two classes c and its complement $\bar{c}$. For each of these classifiers, we have measured recall, precision, and accuracy, but we have focused on measures of accuracy named F_1 measure, and on a single aggregate measure that combines the measures for individual classifiers, that is the Macroaveraging, which computes a simple average over classes, and the Microaveraging pools per-document decisions across classes (Sebastiani, 2002; Christopher D. Manning and Schtze, 2009).

Note that the $m\mathcal{GT}$ is different from a simple list of key words because of the presence of two features: the relations between terms and the hierarchical differentiation between simple words and concepts. To demonstrate the discriminative property of such features we have to prove that the results obtained performing the proposed approach are significantly better than the results obtained by performing the same queries composed of the simple list of words extracted from the $m\mathcal{GT}$. As a result, the aim of the evaluation phase is twofold:

1. To demonstrate the discriminative property of the $m\mathcal{GT}$ compared with a method based on only the words from the $m\mathcal{GT}$ without relations (named Words List $\mathcal{WL}$);
2. To demonstrate that the $m\mathcal{GT}$ achieves good performance when 1% of the training set is employed for each class. Here comparison with well known methods trained on the whole training set will be reported.

We have randomly selected the 1% from each training set (in table 2 is reported the comparison with the original training set dimension) and moreover we have performed the selection 100 times in order to make the results independent of the particular documents selection. As a result we have 100 repositories and from each of them we have calculated 100 $m\mathcal{GT}$s by performing the parameters learning described described above. Due to the fact that each optimisation procedure brings to a different graph (from a topological point of view), we have a different number of pairs for each of them. We have calculated the average number of pairs for each topic and the corresponding average number of terms, see table 3. Note that the average size of $|\mathcal{T}'_p|$ is 120, while the average size of $|\mathcal{T}'|$ is 33 (150 and 47 respectively in the case of the best performance). The overall number of features observed by our method is, independently of the topic, less than the number considered in the case of Rocchio and Support Vector Machines, in fact they have employed a term selection process obtaining $|\mathcal{T}'|$ equals to 50 and 300 respectively.

In table 4 we have reported the best accuracy (calculate in the F_1 measure) obtained by our method and the average value obtained by all 100 graphs. It is surprising how the proposed method, even if the training set is smaller than the original ones, is capable of clustering in most of the case with an accuracy comparable to that obtained by well-known approaches (amongst which we find the worst case that is Rocchio and best that is Support Vector Machines) (Christopher D. Manning and Schtze, 2009). Note that the performance of the proposed method is, independently of the topic, better than the $\mathcal{WL}$, so demonstrating that the graph representation possesses better discriminative properties than a simple list of words. Furthermore it is surprising how $m\mathcal{GT}$ performs in the same way of SVM in the case of the topic *acq* and comparable to Rocchio and Naive Bayes for the other topics. Finally, it should be noticed that the good performances shown by $\mathcal{WL}$ are motived by the fact that the list of words is formed by the terms extracted form the $m\mathcal{GT}$.

[4]Note that considering the 10 largest classes means 75% of the training set and 68% of the test set.

Table 3: Average number of pairs and words for each topic. Values for each run and for each the best run.

Topic	Av. #pairs	Av. #words	#pairs@max	#words@max
earn	83	43	17	17
acq	75	38	162	87
money-fx	98	33	357	48
grain	127	36	204	64
crude	153	40	262	50
trade	178	48	229	80
interest	105	28	143	83
ship	113	16	54	15
wheat	124	26	18	15
corn	139	20	50	15
Average	**120**	**33**	**150**	**47**

Table 4: F_1 and *micro-avg* measure for NB, Rocchio and SVM when 100% of the training set is employed (Christopher D. Manning and Schtze, 2009). The same measures and the *macro-avg* for the $m\mathcal{GT}$ and $\mathcal{WL}$.

	NB (100%)	**Rocchio (100%)**	**SVM (100%)**	**max $\mathcal{WL}$ (1%)**	**av. $\mathcal{WL}$ (1%)**	**max $m\mathcal{GT}$ (1%)**	**av. $m\mathcal{GT}$ (1%)**
earn	96	93	98	82	69	92	83
acq	88	65	94	73	60	94	74
money-fx	57	47	75	39	30	48	35
grain	79	68	95	64	45	68	47
crude	80	70	89	60	40	71	49
trade	64	65	76	53	39	61	46
interest	65	63	78	48	34	50	45
ship	85	49	86	71	30	73	44
wheat	70	69	92	82	41	86	54
corn	65	48	90	54	30	57	47
micro-avg (top 10)	**82**	**65**	**92**	**70**	–	**80**	–
macro-avg (top 10)	–	–	–	**66**	–	**70**	–

6 CONCLUSIONS

In this work we have demonstrated that a *term extraction* procedure based on a mixed *Graph of Terms* representation is capable of achieving better performance than a method based on a simple *term selection* obtained considering only the words composing the graph. Moreover we have demonstrated that the overall performance of the method is good even if 1% of the training set has been employed. As a future work we consider to measure performances of well known methods when trained on the same, small, percentage of the training set. Furthermore, we are interested in finding an analytic method to set the suitable threshold to the *CSV*'s.

REFERENCES

Berkhin, P. (2006). A survey of clustering data mining techniques. In Kogan, J., Nicholas, C., and Teboulle, M., editors, *Grouping Multidimensional Data*, pages 25–71. Springer Berlin Heidelberg.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(993–1022).

Christopher D. Manning, P. R. and Schtze, H. (2009). *Introduction to Information Retrieval*. Cambridge University.

Griffiths, T. L., Steyvers, M., and Tenenbaum, J. B. (2007). Topics in semantic representation. *Psychological Review*, 114(2):211–244.

Ko, Y. and Seo, J. (2009). Text classification from unlabeled documents with bootstrapping and feature projection techniques. *Inf. Process. Manage.*, 45:70–83.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397.

McCallum, A., Nigam, K., Rennie, J., and Seymore, K. (1999). A machine learning approach to building domain-specific search engines. In *Proceedings of the 16th international joint conference on Artificial intelligence - Volume 2*, pages 662–667. Morgan Kaufmann.

Noam, S. and Naftali, T. (2001). The power of word clusters for text classification. In *In 23rd European Colloquium on Information Retrieval Research*.

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