A Semantic Kernel to Classify Texts with Very Few Training Examples

Roberto Basili, Marco Cammisa and Alessandro Moschitti
Department of Computer Science
University of Rome "Tor Vergata"
Rome,Italy
e-mail: {basili,cammisa,moschitti}@info.uniroma2.it

Keywords: kernel methods, similarity measures, support vector machines, WordNet

Received: May 12, 2005

Advanced techniques to access the information distributed on the Web often exploit automatic text categorization to filter out irrelevant data before activating specific searching procedures. The drawback of such approach is the need of a large number of training documents to train the target classifiers. One way to reduce such number relates to the use of more effective document similarities based on prior knowledge. Unfortunately, previous work has shown that such information (e.g. WordNet) causes the decrease of retrieval accuracy.
In this paper, we propose kernel functions to use prior knowledge in learning algorithms for document classification. Such kernels implement balanced and statistically coherent document similarities in a vector space by means of the term similarity based on the WordNet hierarchy. Cross-validation results show the benefit of the approach for Support Vector Machines when few training examples are available.

Povzetek: Predstavljena je kategorizacija besedil na osnovi malo primerov.

1 Introduction

The access to Web distributed information often requires the detection and filtering of the target objects (e.g. texts) before specialized automatic retrieval processes can be applied. In this perspective, text categorization (TC) is a useful approach to filter out irrelevant (or equivalently accept relevant) data.

As the Web is a dynamic source of information, a flexible and fast design of categorization systems is required. One of the most important aspects to achieve the above properties is to limit the time and effort needed to manually annotate large training data. Unfortunately, when few data is available the classification accuracy is rather unsatisfactory. The main reason for this outcome is the document representation based on *bag-of-words* along with the term matching document similarity. If few documents are available for training there is a high probability that terms in test documents will not be matched. Consequently, the document similarity results inadequate for an effective classification.

This problem has been tackled by enriching the document representation with term clustering (*term generalization*) or adding compound terms (*term specification*). Such approaches are based on the assumption that the similarity between two documents can be expressed as the similarity between pairs of complex terms or term clusters. The latter are built based on corpus term distributions, e.g. [4], or prior knowledge external to the target corpus (e.g. provided by WordNet [9]).

The main problem of term cluster representation is the unclear relationship with the one based on simple words. Although (semantic) clusters tend to improve the system recall, simple terms are, on a large scale, more accurate (e.g. [17]). To overcome this problem, hybrid spaces containing terms and clusters were experimented (e.g. [19]) but, again, the results showed that the mixed statistical distributions of clusters and terms impact either marginally or even negatively on the overall accuracy. Hence, the successful introduction of prior/external knowledge relies on the solution of this problem.

In this paper, we propose a model to introduce the semantic lexical knowledge encoded in the WN hierarchy in automatic text classification. The idea is to compute document similarity between two documents d_1 and d_2 by summing the term similarity contributions of all term pairs $\langle t_1, t_2 \rangle$ where $t_1 \in d_1$ and $t_2 \in d_2$. Each pair contribution is evaluated by considering the spatial and topological properties that the two compounding terms have in WN. Such approach has two advantages: (a) we obtain a well defined space which supports the similarity between terms of different surface forms based on external knowledge and (b) we avoid to explicitly define term or sense clusters which inevitably introduce noise.

The above document similarity is a valid kernel that can be used with kernel-based learning machine methods such as Support Vector Machines (SVMs) [25]. Moreover, as we believe that the external knowledge in TC is not very useful when a sufficient amount of training documents is available, we experimented our model in poor training conditions (e.g. 10 documents for each category). The improvement in the accuracy, observed on the classification of the well known Reuters and 20 NewsGroups corpora, shows that our document similarity model is very promising for general IR tasks: unlike previous attempts (e.g. [26, 22, 17], it makes sense of the adoption of semantic external resources (i.e. WN) in IR.

Section 2 introduces the WordNet-based term similarity whereas Section 3 defines the new document similarity measure, the kernel function and its use within SVMs. Section 4 discusses the computational aspects of such kernel. Section 5 presents the comparative results between the traditional linear and the WN-based kernels within SVMs. In Section 6 comparative discussion against the related IR literature is carried out. Finally Section 6 derives the conclusions.

2 Term similarity based on general knowledge

In IR, similarity metrics in vector space models are usually driven by lexical matching. When small training material is available, few words can be effectively used and the resulting document similarity may be inaccurate. Semantic generalizations overcome data sparseness problems as contributions from different but semantically similar words can be derived.

Methods for the induction of semantic word clusters have been widely used in language modeling and lexical acquisition tasks (e.g. [7]). The linguistic resource employed in most previous work is WordNet [9] which contains three subhierarchies for nouns, verbs and adjectives. Each hierarchy represents lexicalized concepts (or senses) organized according to an "*is-a-kind-of*" relation, where a concept s is described by a set of words $syn(s)$, called $synset$, and the words $w \in syn(s)$ are synonyms according to the sense s.

For example, the words *line*, *argumentation*, *logical argument* and *line of reasoning* describe a synset which expresses the methodical process of logical reasoning (e.g. "*I can't follow your line of reasoning*"). Each word/term may be lexically related to more than one synset depending on its senses. The word *line* is also a member of the synset *line*, *dividing line*, *demarcation* and *contrast*, as a *line* denotes also a conceptual separation (e.g. "*there is a narrow line between sanity and insanity*"). The Wordnet noun hierarchy is a direct acyclic graph[1] in which the edges establish the *direct_isa* relations between two synsets.

2.1 Problems with WordNet similarities

The automatic use of WordNet for NLP and IR tasks has shown to be very complex:

First, how the topological distance among senses is related to their corresponding conceptual distance is unclear. The pervasive lexical ambiguity is also problematic as it impacts on the measure of conceptual distances between word pairs.

Second, the approximation of a set of concepts by means of their generalization in the hierarchy implies a conceptual loss that affects the target IR (or NLP) tasks. For example, *black* and *white* are *colors* but also *chess pieces* and this impacts on the similarity score that should be used in IR applications.

Finally, similar words play different roles in IR tasks and in other NLP-based systems, e.g. machine translation, so that the equivalence between them cannot be imposed in general. It is thus difficult to decide the degree of generalization (which allows us to reduce a set of senses into single features) effective for IR.

To solve the above problems, some methods attempt to map (a priori) terms to specific generalization levels, i.e. they *cut* the hierarchy at some levels (e.g. [16, 18]) and use corpus statistics to assign weights to the resulting generalizations. For several tasks (e.g. in TC) this is unsatisfactory: different contexts of the same corpus (e.g. documents) may require different levels of generalization of the same word since they have a different impact on the document similarity.

On the contrary, the *Conceptual Density* (CD) [1, 3] is a flexible semantic similarity measure which depends on the generalizations of word senses not referring to any fixed level of the hierarchy.

2.2 The Conceptual Density

CD defines a metric according to the topological structure of WN. Intuitively, given two words, their lowest common WN hypernym determines a sub-hierarchy. This latter will suggest maximum relatedness if only few levels are used to connect the two words. The CD of such words is expressed as the ratio between the size of the minimal (*ideal*) tree connecting such words and the sub-hierarchy.

To formally define CD, we introduce some basic concepts: let $\bar{g}$ be the set of nodes of the hierarchy rooted in the synset g, i.e. $\{c \in S | c \; isa \; g\}$, where S is the set of WN synsets. By definition $\forall g \in S, g \in \bar{g}$. CD makes a guess about the proximity of the senses, s_u and s_v, of two words u and v, according to the information expressed by the minimal subhierarchy, $\bar{g}$, that includes them. Let G_u be the set of generalizations for at least one sense of the word u, i.e. $G_u = \{g \in S | \exists s \in \bar{g}, u \in syn(s)\}$. The CD of u and v is:

$$CD(u,v) = \begin{cases} 0 & \textbf{iff } G_u \cap G_v = \emptyset \\ max_{g \in G_u \cap G_v} \frac{\sum_{i=0}^{h} (\mu(\bar{g}))^i}{|\bar{g}|} & \\ & \textbf{otherwise} \end{cases} \quad (1)$$

where:

- $G_u \cap G_v$ is the set of WN shared generalizations (i.e. the common hypernyms) of u and v.

[1] As only the 1% of its nodes own more than one parent in the graph, most of the techniques assume the hierarchy to be a tree, and treat the few exception heuristically.

- $\mu(\bar{g})$ is the average number of children per node (i.e. the branching factor) in the sub-hierarchy $\bar{g}$. $\mu(\bar{g})$ depends on WordNet and in some cases its value can approach 1.

- h is the depth of the *ideal*, i.e. maximally dense, *tree* with enough leaves to cover the two senses, s_u and s_v, according to an average branching factor of $\mu(\bar{g})$. This value is actually estimated by:

$$h = \begin{cases} \lfloor log_{\mu(\bar{g})} 2 \rfloor & \textbf{iff } \mu(\bar{g}) \neq 1 \\ 2 & \textbf{otherwise} \end{cases} \quad (2)$$

When $\mu(g)$=1, h ensures a tree with at least 2 nodes to cover s_u and s_v ($height = 2$).

- $|\bar{g}|$ is the number of nodes in the sub-hierarchy $\bar{g}$. This value is statically measured on WN and it is a negative bias for the higher generalization levels (i.e. larger $\bar{g}$).

CD models the semantic distance as the density of the generalizations $g \in S_u \cap S_v$. Such *density* is the ratio between the number of nodes of the *ideal tree* and $|\bar{g}|$. The ideal tree should (a) link the two senses/nodes s_u and s_v with the minimal number of edges (isa-relations) and (b) preserve the same branching factor (*bf*) observed in $\bar{g}$. In other words, this tree contains the minimal number of nodes (and isa-relations) sufficient to connect s_u and s_v according to the topological structure of $\bar{g}$. When *bf* is 1, Eq. 1 degenerates to the inverse of the number of nodes in the path between s_u and s_v, i.e. the simple proximity measure used in [21].

Figure 1 shows a subhierarchy $\bar{g}$ of two senses s_u and s_v and the associated ideal tree. Note that the bf is the average of the node branching factors (nbfs), i.e. 2. This suggests that, in this *zone*, the topological structure has a density quantifiable by a factor equal to 2. Ideally, if two senses are *very close* they should be linked by only one node, i.e. using the ideal tree on the right of the figure. Once the ideal tree is built, the $CD(s_u, s_v)$ is computed by dividing the number of its nodes by the number of nodes in the real hierarchy, e.g. 3/7 for the example.

The final $CD(u, v)$ between two words is the maximum CD among all the word sense pairs. This means that $CD(u, v)$ is determined by the *closest lexical senses*, s_u, $s_v \in \bar{g}$: the remaining senses of u and v are irrelevant, with a resulting semantic disambiguation side effect.

As the number of word pairs is in general very high, efficient approaches to compute the document similarity are needed. The next section describes how kernel methods can make practical the use of the Conceptual Density in Text Categorization.

3 A document similarity kernel based on WordNet

Term similarities are used to design document similarities which are the core functions of most TC algorithms. The one proposed in Eq. 1 is valid for all term pairs of a target vocabulary and has two main advantages:

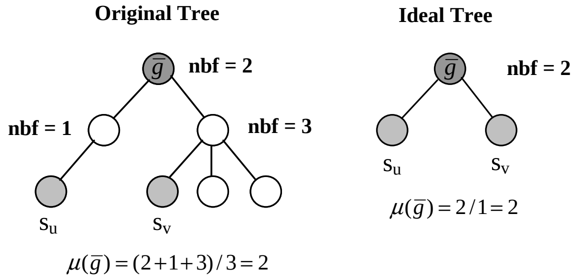

Figure 1: A subhierarchy $\bar{g}$, rooted in g, of two word senses, s_u and s_v and the corresponding ideal tree. The branching factor $\mu(\bar{g})$ is the average over the node branching factors (nbfs) of $\bar{g}$

1. the relatedness of each term occurring in the first document can be computed against *all* terms in the second document, i.e. all different pairs of similar (not just identical) tokens can contribute; and

2. if we use all term pair contributions in the document similarity, we obtain a measure consistent with the term probability distributions, i.e. the sum of all term contributions does not penalize or emphasize arbitrarily any subset of terms.

The positive aspects of the first point is quite clear and will solve data sparseness problems. Regarding the second point, we should consider that when document representation is enriched by means of some term clusters a simplification assumption about all the other possible clusters is made, i.e. a zero probability is assumed for them. As the literature on smoothing techniques has shown, this is not the best way to approach the problem. However, it should also be stated that the discriminative nature of Support Vector Machines makes them less sensitive to such smoothing aspects.

The next subsections present more formally the above ideas.

3.1 Document similarity Kernel

Given two documents d_1 and $d_2 \in D$ (the document set), we define their similarity as:

$$K(d_1, d_2) = \sum_{w_1 \in d_1, w_2 \in d_2} (\lambda_1 \lambda_2) \times \sigma(w_1, w_2) \quad (3)$$

where λ_1 and λ_2 are the weights of the words (features) w_1 and w_2 in the documents d_1 and d_2, respectively, and σ is a term similarity function, e.g. the conceptual density defined in Section 2.

The above document similarity could be used in kernel based machines if we prove that it is a valid kernel function, i.e. if it satisfies the Mercer's conditions [8]. Such conditions establish that the Gram matrix, $\boldsymbol{G} = K(d_i, d_j)$ $\forall i, j = 1, .., l$, where $d_1, .., d_l$ are the training documents, must be positive semi-definite. In order to obtain such property in [21] was adopted as term similarity the matrix $\boldsymbol{M}' \cdot \boldsymbol{M}$, where $\boldsymbol{M}$ is defined by $\sigma(w_1, w_2)$ with $w_1, w_2 \in V$ and $\boldsymbol{M}'$ is its transposed. As shown in [8], $\boldsymbol{P} = \boldsymbol{M}' \cdot \boldsymbol{M}$ as well as $K(d_i, d_j) = \vec{\lambda}_i' \boldsymbol{P} \vec{\lambda}_j$ are positive semi-definite matrices. Unfortunately, this approach does not use the original similarity matrix $\boldsymbol{M}$, i.e. the term to term similarity defined in WN, since $\boldsymbol{P} = \boldsymbol{M}^2$. Although, we can see the application of such matrix as a feature expansion techniques, we loose the direct similarity semantics of two words encoded by the matrix $\boldsymbol{M}$.

With the aim to preserve an intuitive notion of document similarity, we adopt the simple similarity term matrix given by $\boldsymbol{P} = \sigma(w_1, w_2) = CD(w_1, w_2)$ without applying the square operation. This means that (a) we exactly use Eq. 3 as a kernel function and (b) we need to prove that P is positive semi-definite. To prove that $\boldsymbol{P}$ is positive semi-definite, a general way is to show that all its eigenvalues are non negative. Thus, we run the single value decomposition algorithm and verified that such condition holds.

Additionally, in [10], it is shown that when kernel functions are not positive semi-definite, SVMs still solve a data separation problem in pseudo Euclidean spaces. The drawback is that the solution may be only a local optimum. Therefore, we can experimentally observe if the empirical results are satisfactory. Our extensive experimentation (reported in Section 5) with different corpora and many training document subsets provides some evidence that Eq. 3 is a useful function as SVMs based on Eq. 3 always convergence to a significant accuracy.

The next section shows as a similarity measure can be used within Support Vector Machines.

3.2 Kernel methods and Support Vector Machines

Given a vector space in $\mathbb{R}^\eta$ and a set of positive and negative points, SVMs classify vectors according to a separating hyperplane, $H(\vec{x}) = \vec{\omega} \cdot \vec{x} + b = 0$, where $\vec{x}$ and $\vec{\omega} \in \mathbb{R}^\eta$ and $b \in \mathbb{R}$ are learned by applying the *Structural Risk Minimization principle* [25]. From the kernel theory we have that:

$$H(\vec{x}) = \Big(\sum_{h=1..l} y_h \alpha_h \vec{x_h}\Big) \cdot \vec{x} + b = \sum_{h=1..l} y_h \alpha_h \vec{x}_h \cdot \vec{x} + b =$$

$$= \sum_{h=1..l} y_h \alpha_h \phi(d_h) \cdot \phi(d) + b =$$

$$= \sum_{h=1..l} y_h \alpha_h K(d_h, d) + b. \quad (4)$$

where y_h and α_h are the class labels and the Lagrange multipliers associated with the l training documents d_h. d is the classifying document and ϕ is the mapping which projects it and d_h in the vectors $\vec{x}$ and $\vec{x}_h$, respectively. By choosing the right ϕ, the product $K(d, d_h) = \langle \phi(d) \cdot \phi(d_h) \rangle$ will correspond to the *Semantic WN-based Kernel* (SK).

Eq. 4 shows that to evaluate the separating hyperplane in $\mathbb{R}^\eta$, we do not need to evaluate the entire vector $\vec{x_h}$ or $\vec{x}$. As it is sufficient to compute $K(d, d_h)$, we can carry out the learning with Eq. 3 in $\mathbb{R}^n$, avoiding to use the explicit representation in the $\mathbb{R}^\eta$ space. The real advantage is that we can consider only the word pairs associated with non-zero weights, i.e. we can use a sparse vector computation. Additionally, to have a uniform score across different document size, the kernel function can be normalized as follows:

$$\frac{SK(d_1, d_2)}{\sqrt{SK(d_1, d_1) \cdot SK(d_2, d_2)}}$$

4 Computational Aspects

The previous section has shown that we can apply the kernel trick to train the SVMs in the dual space. In this way we avoid to compute the huge space of all WN word pairs. However, the computational complexity of the algorithm is higher than the usual approach based on the *bag-of-words* model. It depends on two main aspects:

1. The similarity measure between two documents d_1 and d_2 requires the evaluation of all the word pairs $\langle w_1, w_2 \rangle$. This leads to a complexity of $O(|d_1| \times |d_2|)$ which is remarkably higher than the usual complexity, $O(|d_1| + |d_2|)$, of traditional approaches.

2. The conceptual density evaluation requires to navigate the WN hierarchy which includes more than 10^5 nodes. On the contrary the traditional term similarity is carried out by a fast string matching function.

Since we use an implicit document representation, we need to test all document pairs during the kernel evaluation, thus, unless we apply a feature selection in the kernel space, the complexity of point 1 cannot be improved. On the contrary, we can improve the conceptual density evaluation by pre-computing it for all WN term pairs and store them in a hash table.

In the next section we described the technical approach that we adopted.

4.1 Technical approach

To evaluate the CD between two words u and v, for each sense pairs, s_u and s_v, we need to derive: (a) the minimal subhierarchy $\bar{g}$ (with its number of nodes) which includes both of them and (b) the ideal tree associated with $\bar{g}$.

Step (a) requires to evaluate the lowest hierarchy node g that dominates s_u and s_v, i.e. we need to consider all the ISA relation paths that links s_u and s_v. To optimize this step, we pre-computed all the *transitive closures* (about 2×10^5) of the ISA relation for all WN synsets along with the number of nodes dominated by g.

The ideal tree evaluation corresponds to derive its height and the branching factor, $\mu(\bar{g})$. The former is computed by means of Eq. 2. The latter can be incrementally precomputed by navigating bottom-up the hierarchy.

The current version of WordNet package[2] makes available a set of built in libraries, written in C language, to navigate the hierarchy. These use an internal structure to store the WN information (e.g. glosses, relations,...). To make more efficient such data structures, we retain only (1) the relations between nouns and synsets, (2) the *"is-a-kind-of"* hierarchy and (3) we implemented special libraries to gather information efficiently.

Moreover, to speed-up the kernel computation, we designed hashed associative containers which store any word pair similarity requested during the learning or testing phase. Note that, although the term pairs are sparse, their similarity values are not. We observed that, for a set of about $32,000$ words (i.e. $1,024 \times 10^6$ pairs), the number of different values were about $6,500$, if we consider only similarity values higher than 2×10^{-5}. We exploit this property by replacing them with an integer index to access a dictionary of float numbers. This reduced the memory usage by a factor of 2.

Given the above optimized architecture, we carried out an extensive experimentation (as illustrated in the next section).

5 Experiments

The use of WordNet (WN) as a term similarity function introduces a prior knowledge whose impact on the Semantic Kernel (SK) should be experimentally assessed. The main goal is to compare the traditional Vector Space Model kernel against SK, both within the Support Vector learning algorithm.

The high complexity of the SK limits the size of the experiments that we can carry out in a feasible time. Moreover, we are not interested to large collections of training documents as in these training conditions the simple *bag-of-words* models are in general very effective, i.e. they seem to model well the document similarity needed by the learning algorithms. Thus, we carried out the experiments on small subsets of the 20NewsGroups[3] (20NG) and the *Reuters-21578*[4] corpora to simulate critical learning conditions.

5.1 Experimental set-up

For the experiments, we used the SVM-light software [12] (available at `svmlight.joachims.org`) with the default linear kernel on the token space (adopted for the baseline evaluations). For the SK evaluation we implemented Eq. 3 with $\sigma(\cdot, \cdot) = CD(\cdot, \cdot)$ (Eq. 1) inside SVM-light. As Eq. 1 is only defined for nouns, a part of speech (POS) tagger was applied. However, also verbs, adjectives and numerical features were included in the feature space. For these tokens a $CD = 0$ is assigned to pairs made by different strings. As the POS tagger could introduce errors, in a second experiment, any token with a successful look-up in the WN noun hierarchy was considered in the kernel. This approximation has the benefit to retrieve useful information even for verbs and capture the similarity between verbs and some nouns, e.g. *to drive* (via the noun *drive*) has a common synset with *parkway*.

For the evaluations, we applied a careful SVM parameterization: a preliminary investigation suggested that the trade off (between the training-set error and margin, i.e. c option in SVM-light) parameter optimizes the F_1 measure for values in the range [0.02,0.32][5]. We noted also that the cost-factor parameter (i.e. j option) is not critical, i.e. a value of 10 always optimizes the accuracy. Feature selection techniques and weighting schemes were not applied in our experiments as they cannot be accurately estimated from few training documents.

The classification performance was evaluated by means of the F_1 measure[6] for the single category and the MicroAverage F_1 for the final classifier pool [28]. Given the high computational complexity of SK, we selected 8 categories from the 20NG[7] and 8 from the Reuters corpus[8]

To derive statistically significant results from few training documents, we randomly selected 10 different samples from the 8 categories of each corpus. We trained the classifiers on one sample, parameterized on a second sample and derived the measures on the other 8. By rotating the training sample, we obtained 80 different measures for each model. The size of the samples ranged from 24 to 160 documents depending on the target experiment. The training of the SVMs adopting SK required about 10/20 minutes for each sample (depending on their size). Considering that the parameterization phase carries out the training of each classifier 16 times (one for each parameter values), we chose to use an 8 multiprocessor machine in which the classifiers can run independently. Even with this optimized strategy, we employed about 1 month to accomplish all the experiments.

5.2 Cross validation results

With the aim of showing the benefit of SK (Eq. 3) for text categorization, we compared it with the linear kernel which obtained the best F_1 measure in [12].

First, in Table 1, we report the results for 8 categories of 20NG on 40 training documents. They are expressed as the *Mean* and the *Std. Dev.* over 80 runs. Column 2, 3 and 4 show the F_1 for the linear kernel (*bow*), for SK

[2] Downloadable from `wordnet.princeton.edu`

[3] Available at `www.ai.mit.edu/people/jrennie/20Newsgroups/`.

[4] The Apté split available at `kdd.ics.uci.edu/databases/reuters21578/reuters21578.html`.

[5] We used all the values from 0.02 to 0.32 with step 0.02.

[6] F_1 assigns equal importance to Precision P and Recall R, i.e. $F_1 = \frac{2P \cdot R}{P+R}$.

[7] We selected the 8 most different categories (in terms of their content) i.e. *Atheism*, *Computer Graphics*, *Misc Forsale*, *Autos*, *Sport Baseball*, *Medicine*, *Talk Religions* and *Talk Politics*.

[8] We selected the 8 largest categories, i.e. *Acquisition*, *Earn*, *Crude*, *Grain*, *Interest*, *Money-fx*, *Trade* and *Wheat*.

without applying POS information and for SK with the use of POS information (SK-POS), respectively. The last row of the table shows the MicroAverage performance for the above three models on all 8 categories. We note that SK improves bow of 3%, i.e. 34.3% vs. 31.5% and that the POS information reduces the improvement of SK, i.e. 33.5% vs. 34.3%.

Second, to verify that the above results are general, we repeated the evaluation over the 8 categories of Reuters with samples of 24 and 160 documents, respectively. Table 2 illustrates that (1) again SK improves bow (41.7% - 37.2% = 4.5%) and (2) as the number of documents increases the improvement decreases (77.9% - 75.9% = 2%).

Third, the complexity of the classification task across the samples varies remarkably thus the standard deviations assume high values. Nevertheless, the high number of samples should provide reliable results. To verify the hypothesis that SK improves bow, we evaluated the Std. Dev. of the difference, d, between the MicroAverage F_1 of SK and the MicroAverage F_1 of bow over the samples. For instance, in relation to the Table 2 experiment, we obtained that the mean and the Std. Dev. of d on the 80 test samples (of 24 documents) are 4.53 and 6.57, respectively. Using the Normal Distribution, we found that at a confidence level of 99% d is in the range [2.40,6.66], thus the probability that d is negative, i.e. bow is better than SK, is very small.

Category	bow	SK	SK-POS
Atheism	29.5±19.8	32.0±16.3	25.2±17.2
Comp.Graph	39.2±20.7	39.3±20.8	29.3±21.8
Misc.Forsale	61.3±17.7	51.3±18.7	49.5±20.4
Autos	26.2±22.7	26.0±20.6	33.5±26.8
Sport.Baseb.	32.7±20.1	36.9±22.5	41.8±19.2
Sci.Med	26.1±17.2	18.5±17.4	16.6±17.2
Talk.Relig.	23.5±11.6	28.4±19.0	27.6±17.0
Talk.Polit.	28.3±17.5	30.7±15.5	30.3±14.3
MicroAvg. F_1	31.5±4.8	34.3±5.8	33.5±6.4

Table 1: Performance of the linear and Semantic Kernel with 40 training documents over 8 categories of 20NewsGroups collection.

Category	24 docs		160 docs	
	bow	SK	bow	SK
Acq.	55.3±18.1	50.8±18.1	86.7±4.6	84.2±4.3
Crude	3.4±5.6	3.5±5.7	64.0±20.6	62.0±16.7
Earn	64.0±10.0	64.7±10.3	91.3±5.5	90.4±5.1
Grain	45.0±33.4	44.4±29.6	69.9±16.3	73.7±14.8
Interest	23.9±29.9	24.9±28.6	67.2±12.9	59.8±12.6
Money-fx	36.1±34.3	39.2±29.5	69.1±11.9	67.4±13.3
Trade	9.8±21.2	10.3±17.9	57.1±23.8	60.1±15.4
Wheat	8.6±19.7	13.3±26.3	23.9±24.8	31.2±23.0
Mic.Avg.	37.2±5.9	41.7±6.0	75.9±11.0	77.9±5.7

Table 2: Performance of the linear and Semantic Kernel with 24 and 160 training documents over 8 categories of the Reuters corpus.

Next, the above findings confirm that SK outperforms the *bag-of-words* kernel in critical learning conditions as the semantic contribution of the SK recovers useful information. To confirm this hypothesis we carried out experiments with samples of different size, i.e. 3, 5, 10, 15 and 20 documents for each category. Figures 2 and 3 show the learning curves for 20NG and Reuters corpora. Each point refers to the average on 80 samples.

As expected the improvement provided by SK decreases when more training data is available. However, the SK model without POS information on 160 training documents still outperforms the baseline of about 2-3%. This suggests that the matching between noun-verb pairs still provides semantic information which is useful for topic detection. In particular, during the similarity estimation, each word shows a non-null similarity with 60.05 words on average. This is useful to increase the amount of information available to the SVMs. To confirm such hypothesis, we removed the string matching contributions from SK such that only words having different surface forms participate to the evaluation of Eq. 3. The interesting result is that SK still converged to a MicroAverage F_1 measure of 56.4% (compare with Table 2). This shows that SVMs can discern between the correct and incorrect categories by using only the WN similarity.

Finally, to provide a comparison with literature models, we experimented with SK by training on 10 random samples of 40 documents and testing on the Reuters test set, i.e. on the 2,502 documents labeled with the 8 target categories. The SK obtained a MicroAverage F_1 (averaged on the 10 runs) of 67.4% which is higher than 65,0% of the baseline outcome. In a second experiment, we used for the *Acquisition* category all the available training data from the 8 categories (i.e. 6,367 documents) obtaining a F_1 of 94.5% for the SK vs. a F_1 of 96.0% of the baseline. This shows that when the number of training documents is large, the word distributions assume a statistical significance that is more reliable than the distribution of the term pairs weighted by WN. Indeed, these latter introduce necessarily some errors due to disambiguation mistakes or incorrect (for the specific target domain) term similarities.

In summary, WN allows the learning algorithm to carry out document similarity when few or no terms can be matched. When precise terms are available with a reliable statistical distribution, string matching is more precise since it is less affected by errors.

6 Related Work

Several IR studies focus on the term similarity models to embed prior knowledge in document similarity.

In [15] a *Latent Semantic Indexing* analysis was used for term clustering. Such approach assumes that values x_{ij} in the transformed term-term matrix represents the similarity (values ≤ 0) and anti-similarity (values < 0) between terms i and j. This enables both positive and negative clusters of terms. Evaluation of query expansion techniques

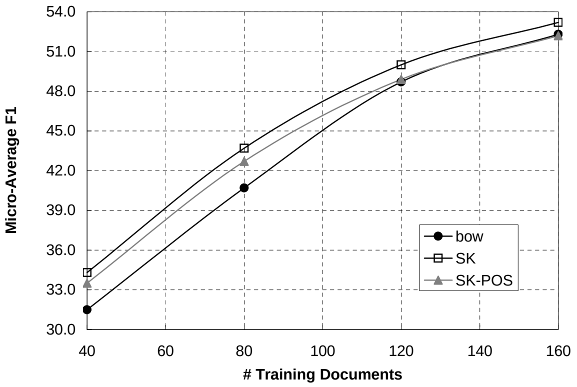

Figure 2: MicroAverage F_1 of SVMs using bow, SK and SK-POS kernels over the 8 categories of 20NewsGroups.

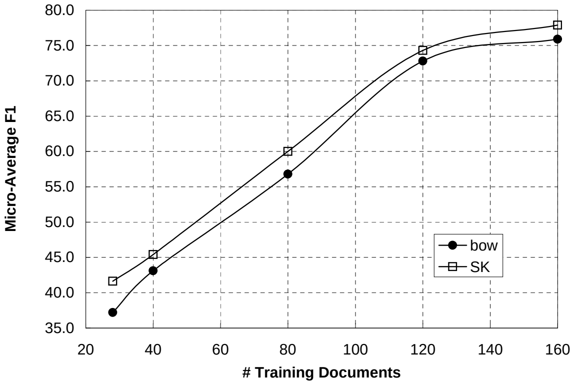

Figure 3: MicroAverage F_1 of SVMs using bow and SK over the 8 categories of the Reuters corpus.

showed that positive clusters can improve Recall of about 18% for the *CISI* collection, 2.9% for *MED* and 3.4% for *CRAN*. Furthermore, the negative clusters, when used to prune the result set, improved the precision.

In [4], a feature selection technique that clusters similar features/words, called the Information Bottleneck (IB), is applied to TC. Support Vector Machines trained over such clusters were experimented with three different corpora: *Reuters-21578*, WebKB and 20NewsGroups. Controversial results were obtained as the cluster based representation outperformed the simple *bag-of-words* only on the latter collection (>3%). This was explained as a consequence of the corpus "complexity". Reuters and WebKB corpora seem to require few features to reach optimal performance. IB can thus be adopted either to reduce the problem complexity as well as to increase accuracy by using a simpler representation space.

In [6], Latent Semantic Analysis (LSA) was applied to derive domain-specific concepts and to create semantic document representations over these concepts. Such representations (based on both terms and concepts) were used to design weak classifiers. Concepts were derived from different LSA models (i.e. LSA spaces of different dimensions). AdaBoost was applied to efficiently combine weak hypotheses and integrate term and concept based information. The experiments on two standard document collections show that conceptual features in addition to terms lead to consistent and quite substantial accuracy gains. In our own opinion such evidence is restricted to the used experimental set-up, i.e. classification models, parameterization, data pre-processing and so on. Indeed, the highest F_1, i.e. 85.82%, reached on the Reuters corpus using the extended representation is lower than the one achieved by means of the *bag-of-words* in other work, e.g. 87.8% using ADABOOST.MH (see the table in [20] about comparative TC results on Reuters corpus). Consequently, we cannot derive that LSA representations are better than the simple *bag-of-words* (when a sufficient amount of training data is used).

The use of *external* semantic knowledge for document retrieval has even been more problematic. In [22], a study on the impact of semantic ambiguity was carried out. A WN-based semantic similarity function between noun pairs was applied to improve indexing and document-query matching. However, the WSD algorithm had a performance ranging between 60-70%, and this made the overall semantic similarity not effective.

Other studies on semantic information for improving IR were carried out in [24] and [26, 27]. Word semantic information was used for text indexing and query expansion, respectively. In [27], it was shown that semantic information derived directly from WN with automatic WSD produces poor results. Nevertheless, recently, a revised approach to the use of word senses for document indexing was proposed in [23]. Only senses which are automatically determined with a high probability are utilized. This enabled the experimented retrieval system to improve the accuracy over the simple *bag-of-words*.

In TC word senses have a similar impact if not lower: when enough training data is available, the positive and negative examples of a category allow the learning algorithm to build implicit word clusters based on corpus statistics. These provide matching capabilities more accurate than the matching between concepts of different surface forms defined in external resources, e.g. WN term similarity. Moreover, different categories are better characterized by different words rather than different senses [17].

In [19], WN senses were used to replace words without any word sense disambiguation. The result was a small improvement on a poorly accurate *state-of-art* TC algorithm on a small corpus. When a more statistical reliable set of documents was used, the adopted representation produced a performance decrease. The scale and assessment provided in [17] (3 corpora using cross-validation techniques) showed that even an accurate disambiguation of WN senses (about 80% accuracy on nouns) did not improve TC.

In [14], an extensive experimentation with several algorithms has been carried out to compare the accuracy of classifiers based on words and senses. The target document collection was a subset of the Brown Corpus annotated with semantic concordance. The results indicate that the use of senses does not produce any significant categorization improvement.

In [5], AdaBoost classifiers are trained with document represented by concepts extracted from WN. Experiments with Reuters and Ohsumed show an improvement on the *bag-of-words* representation. Again this results cannot be generalized as the absolute value of the highest achieved F_1, i.e. 85.89%, on the Reuters corpus, is lower than the best literature results, i.e. 87.8%. Nevertheless, it is worth to note that other relevant improvements were obtained on the Ohsumed corpus for which the results of the best TC models are not available. Thus, on corpora different from Reuters we do not know if the conceptual representation improves the *bag-of-words*. According to the analysis carried out in [4], we may assume that the Reuters corpus is not representative in general and the *bag-of-words* approach is superior only on some corpora.

In [21] an approach similar to the one presented in this article was proposed. A term proximity function was used to design a kernel able to semantically smooth the similarity between two document terms. Such semantic kernel was designed as a combination of the Radial Basis Function (RBF) kernel with the term proximity matrix. Entries in this matrix are inversely proportional to the length of the WN hierarchy path linking the two terms. The performance, measured over the 20NewsGroups corpus, showed an improvement of 2% over the *bag-of-words*. The main differences with our approach are:

First, the term proximity is not fully sensitive to the information of the WN hierarchy. For example, if we consider pairs of equidistant terms, the nearer to the WN top level a pair is the lower similarity it should receive, e.g. *sky* and *location* (hyponyms of *entity*) should not accumulate similarity like *knife* and *gun* (hyponyms of *weapon*). Measures, like CD, that deal with this problem have been widely proposed in literature (e.g. [18]) and should be always applied.

Second, the kernel-based CD similarity is an elegant combination of lexicalized and semantic information. In [21] the combination of weighting schemes, the RBF kernel and the proximity matrix has a less clear interpretation.

Finally, the experiments were carried out by using only 200 features (selected via Mutual Information statistics). In this way the contribution of rare or non statistically significant terms is neglected. In our view, such features may give, instead, a relevant contribution once we move in the SK space generated by the WN similarities.

Other important work on semantic kernel for retrieval has been developed in [8, 13]. Two methods for inferring semantic similarity from a corpus were proposed:

In the first a system of equations were derived from the dual relation between word-similarity based on document-similarity and vice versa. The equilibrium point was used to derive the semantic similarity measure.

The second method models semantic relations by means of a diffusion process on a graph defined by lexicon and co-occurrence information. The major difference with our approach is the use of a different source of prior knowledge, i.e. WN. Similar techniques were also applied in [11] to derive a Fisher kernel based on a latent class decomposition of the term-document matrix.

In summary, a careful analysis of literature work shows that prior knowledge (derived directly from the corpus or extracted by external resources) is not able to improve the best TC model learned with an adequate number of training data. On the contrary, the experiments shown in this paper suggest the following reasonable hypothesis: when the statistical word distributions derivable from training data are not reliable, we can use external resources to provide an effective semantic smoothing. In other words, two documents containing different terms have zero match probability in the *bag-of-words* model. Using term similarity we associate them with a probability different from zero designing a more accurate model. Of course, this approximation is less accurate than the probability distributions derived from a statistically representative sample of documents.

7 Conclusions

The way to use semantic prior knowledge in IR has always been an interesting subject as confirmed by the examined literature work.

In this paper, we applied the conceptual density function on the WordNet (WN) hierarchy to define a document similarity metric. Accordingly, we defined a semantic kernel (SK) to train Support Vector Machine classifiers. Cross-validation experiments over 8 categories of 20NewsGroups and Reuters corpora over multiple samples have shown that:

- in poor training data conditions, the WN prior knowledge can be effectively used to improve the TC accuracy (up to 4.5 absolute percent points, i.e. 10%);
- the CD is an effective way to capture the topological WN properties; and
- the higher is the number of training documents, the lower is the improvement produced by SK. This suggests that the general prior knowledge embedded in WN is useful to increase accuracy until the category statistical information (e.g. its word probability distributions) is not *completely* reliable.

These promising results enable a number of future researches: (1) larger scale experiments with different measures and semantic similarity models (e.g. [18]); (2) improvement of the overall efficiency by exploring feature selection methods over the SK; and (3) the extension of the semantic similarity by a general (i.e. non binary) application of the conceptual density model as proposed in [2] for semantic tagging.

References

[1] E. Agirre and G. Rigau. Word sense disambiguation using conceptual density. In *Proceedings of COLING'96, pages 16–22, Copenhagen, Danmark.*, 1996.

[2] R. Basili and M. Cammisa. Unsupervised semantic disambiguation. In *In Proceedings of LREC Workshop on "Beyond Named Entity Recognition - Semantic Labelling for Natural Language Processing Tasks"*, Lisbon, Portugal, 2004.

[3] R. Basili, M. Cammisa, and F. M. Zanzotto. A similarity measure for unsupervised semantic disambiguation. In *In Proceedings of Language Resources and Evaluation Conference*, 2004.

[4] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. On feature distributional clustering for text categorization. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 146–153, New Orleans, Louisiana, United States, 2001. ACM Press.

[5] S. Bloehdorn and A. Hotho. Text classification by boosting weak learners based on terms and concepts. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 2004), 1-4 November 2004, Brighton, UK*, pages 331–334. IEEE Computer Society, NOV 2004.

[6] L. Cai and T. Hofmann. Text categorization by boosting automatically extracted concepts. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 182–189, New York, NY, USA, 2003. ACM Press.

[7] S. Clark and D. Weir. Class-based probability estimation using a semantic hierarchy. *Comput. Linguist.*, 28(2):187–206, 2002.

[8] N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *J. Intell. Inf. Syst.*, 18(2-3):127–152, 2002.

[9] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press., 1998.

[10] B. Haasdonk. Feature space interpretation of SVMs with indefinite kernels. *IEEE Trans Pattern Anal Mach Intell*, 27(4):482–92, Apr 2005.

[11] T. Hofmann. Learning probabilistic models of the web. In *Research and Development in Information Retrieval*, pages 369–371, 2000.

[12] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, 1999.

[13] J. Kandola, J. Shawe-Taylor, and N. Cristianini. Learning semantic similarity. In *in Neural Information Processing Systems (NIPS 15) - MIT Press.*, 2002.

[14] A. Kehagias, V. Petridis, V. G. Kaburlasos, and P. Fragkou. A comparison of word- and sense-based text categorization using several classification algorithms. *J. Intell. Inf. Syst.*, 21(3):227–247, 2003.

[15] A. Kontostathis and W. Pottenger. Improving retrieval performance with positive and negative equivalence classes of terms, 2002.

[16] H. Li and N. Abe. Generalizing case frames using a thesaurus and the mdl principle. *Computational Linguistics*, 23(3), 1998.

[17] A. Moschitti and R. Basili. Complex linguistic features for text classification: a comprehensive study. In S. McDonald and J. Tait, editors, *Proceedings of ECIR-04, 26th European Conference on Information Retrieval*, Sunderland, UK, 2004. Springer Verlag.

[18] P. Resnik. Selectional preference and sense disambiguation. In *Proceedings of ACL Siglex Workshop on Tagging Text with Lexical Semantics, Why, What and How?, Washington, April 4-5, 1997.*, 1997.

[19] S. Scott and S. Matwin. Feature engineering for text classification. In I. Bratko and S. Dzeroski, editors, *Proceedings of ICML-99, 16th International Conference on Machine Learning*, pages 379–388, Bled, SL, 1999. Morgan Kaufmann Publishers, San Francisco, US.

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

[21] G. Siolas and F. d'Alché Buc. Support vector machines based on a semantic kernel for text categorization. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)-Volume 5*, page 5205. IEEE Computer Society, 2000.

[22] A. F. Smeaton. Using NLP or NLP resources for information retrieval tasks. In T. Strzalkowski, editor, *Natural language information retrieval*, pages 99–111. Kluwer Academic Publishers, Dordrecht, NL, 1999.

[23] C. Stokoe, M. P. Oakes, and J. Tait. Word sense disambiguation in information retrieval revisited. In *Proceedings of SIGIR03, Canada*, 2003.

[24] M. Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In A. P. New York, editor, *The Second International Conference on Information and Knowledge Management (CKIM 93)*, pages 67–74, 1993.

[25] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[26] E. M. Voorhees. Using wordnet to disambiguate word senses for text retrieval. In R. Korfhage, E. M. Rasmussen, and P. Willett, editors, *Proceedings of the 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Pittsburgh, PA, USA, June 27 - July 1, 1993*, pages 171–180. ACM, 1993.

[27] E. M. Voorhees. Query expansion using lexical-semantic relations. In W. B. Croft and C. J. van Rijsbergen, editors, *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Dublin, Ireland, 3-6 July 1994 (Special Issue of the SIGIR Forum)*, pages 61–69. ACM/Springer, 1994.

[28] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval Journal*, 1999.