Improving Text Categorization Bootstrapping via Unsupervised Learning

ALFIO GLIOZZO
STLab-ISTC-CNR, Rome
CARLO STRAPPARAVA
FBK-IRST, Povo
and
IDO DAGAN
Bar Ilan University

We propose a text-categorization bootstrapping algorithm in which categories are described by relevant seed words. Our method introduces two unsupervised techniques to improve the initial categorization step of the bootstrapping scheme: (i) using latent semantic spaces to estimate the similarity among documents and words, and (ii) the Gaussian mixture algorithm, which differentiates relevant and nonrelevant category information using statistics from unlabeled examples. In particular, this second step maps the similarity scores to class posterior probabilities, and therefore reduces sensitivity to keyword-dependent variations in scores. The algorithm was evaluated on two text categorization tasks, and obtained good performance using only the category names as initial seeds. In particular, the performance of the proposed method proved to be equivalent to a pure supervised approach trained on 70–160 labeled documents per category.

Categories and Subject Descriptors: I.2.7 [**Natural Language Processing**]: Text Categorization

General Terms: Algorithms, Experimentation, Languages, Theory

Additional Key Words and Phrases: Text categorization, bootstrapping, unsupervised machine learning

ACM Reference Format:

Gliozzo, A., Strapparava, C., and Dagan, I. 2009. Improving text categorization bootstrapping via unsupervised learning. ACM Trans. Speech Lang. Process. 6, 1, Article 1 (October 2009), 24 pages. DOI = 10.1145/1596515.1596516 http://doi.acm.org/10.1145/1596515.1596516

This work was conducted in the framework of the FIRB project RBIN045PXH and the collaboration project between FBK-IRST, University of Haifa and Bar Ilan University.
Authors' addresses: A. Gliozzo; email: gliozzo@gmail.com; C. Strapparava; email: strappa@fbk.eu; I. Dagan; email: dagan@cs.biu.ac.il.

DOI 10.1145/1596515.1596516 http://doi.acm.org/10.1145/1596515.1596516

1. INTRODUCTION

Classification is the task of assigning category labels, taken from a predefined set of categories (classes), to instances in a dataset. Within the classical supervised learning paradigm, the task is approached by providing a learning algorithm with a training set of manually labeled examples. In practice, it is not always easy to apply this schema to natural language processing (NLP) tasks. For example, supervised systems for text categorization (TC) require large amounts of hand-labeled texts, whereas in many applied cases it is quite difficult to collect hand-labeled data. On the other hand, unlabeled text collections are in general easily available, even with the requirement that they keep the same distribution from which the test documents are sampled.

Bootstrapping was proposed as a technique for avoiding, or at least considerably reducing, the need of corpora annotation. Some research focused on starting the bootstrapping process considering a small amount of labeled examples. In this article we follow an alternative approach, that is, improving the bootstrapping schemata that provide the necessary supervision by means of sets of "seeds" of intuitively relevant features. Adopting terminology from formal logic [Barendregt 1984], we think it is evocative referring to the standard example-based supervision mode as *extensional learning* (EL), as classes are being specified by means of examples of their elements (their *extension*). Feature-based supervision[1] is referred to as *intensional learning* (IL), as features may often be perceived as describing the *intension* of a category, such as providing the name or prominent key terms for a category in TC.

The IL approach goes back to classical rule-based classification methods, where the user is expected to specify exact classification rules that operate in the feature space. Within the machine-learning paradigm, IL has been incorporated as a technique for bootstrapping an extensional learning algorithm, as in Yarowsky [1995]; Collins and Singer [1999]; Abney [2002]; Liu et al. [2004]; Abney [2004]. In this approach the user does not need to specify exact classification rules (and feature weights), but rather performs a somewhat simpler task of specifying a few seed features typical for the category. Given the list of seed features, the typical bootstrapping scheme consists of (i) preliminary categorization of the unlabeled dataset based on a similarity score between the seed features and the classified document; and (ii) training an (extensional) supervised classifier using the automatic label classification from step (i) as the training data (the second step is possibly iterated, e.g., by an expectation-maximization schema). The core part of IL bootstrapping is step (i), that is, the initial unsupervised classification of the unlabeled dataset based on the seed features. This step has often been approached by relatively simple methods, in the hope that the second step of supervised training would be robust enough to deal with noise from the initial training set. Even so, the effectiveness of the first step is crucial for satisfactory performance of the subsequent supervised training.

The goal of this article is to investigate and improve two specific weaknesses that inherently affect the IL schema. First, to be attractive with respect to

[1]Providing hand-specified features is in effect supervision, since a human is specifying the features.

EL, where typically a substantial number of labeled documents is assumed, an important aim is that the user usually prefers to provide very little information about the intension of a category. Therefore, one suitable method is to augment the information provided by the user by applying some similarity measure in the feature space. Second, step (i) of IL inherently suffers from a score-scaling problem. Indeed, supervised methods typically scale their classification decision scores to optimize a separator between positive and negative examples. As far as supervised learning maximizes accuracy, they typically learn more relaxed constraints for large categories, while they tend to be very conservative in classifying the smaller ones. Supervised classifiers are then able to learn the bias of the category distribution directly from the training data. However, in the IL settings no labelled examples are provided. What the algorithm "knows" is only a set of discriminative features for each category, so directly using the output of the similarity function for classification can be problematic. In fact, if two different categories get assigned a different number of keywords, any cosine-like similarity function will tend to score higher for the category having more keywords, irrespective of the document. This is a general problem, as it is strongly related to the dualism between intensional and extensional definitions, well-known in description logics, for example. If one concept is a subclass of another, then the set of properties (i.e., the intensional definition) of the latter will be a subset of the former. On the other hand, the cardinality of the set of instances belonging to the latter will be obviously higher. Hence it is desirable to find a way to scale the similarity score into a value range that is unified across all categories.

Following our preliminary work [Gliozzo et al. 2005], in this article we propose two unsupervised mechanisms within the initial classification step to be applied to TC. In particular, (a) utilizing a latent semantic space to obtain better similarity assessments between category seeds and classified examples; and (b) applying a Gaussian mixture (GM) algorithm, which provides a principled unsupervised estimation of classification probability. As shown in our experiments, incorporating these steps consistently improved the accuracy of the initial categorization step, which in turn yielded a better final classifier after the bootstrapping step, thanks to the more accurate training set. Most importantly, we obtained comparable or better performance than previous IL methods using *only* the category names as seed features, while the other IL methods in the literature required collecting a larger number of seed terms, which in general is not a trivial task.

Interesting results were revealed when comparing our IL method to the learning curve of a state-of-the-art (extensional) support vector machine (SVM) classifier, trained on manually labeled documents. The EL classifier required 70 (Reuters dataset) or 160 (Newsgroup dataset) documents per category to achieve the same performance that IL obtained using only the category names. These results suggest that IL may provide an appealing cost-effective alternative when suboptimal accuracy suffices, or when it is too costly or impractical to obtain sufficient labeled training. Optimal combination of extensional and intensional supervision is raised as a challenging topic for future research.

2. BACKGROUND: BOOTSTRAPPING FOR TEXT CATEGORIZATION

It is worth mentioning that many problems in computational linguistics are suitable for bootstrapping (see for example, Abney [2002] for a general mathematical introduction; Yarowsky [1995] for an application to word sense disambiguation; Collins and Singer [1999] for named entity classification, just to mention a few). The cotraining approach [Blum and Mitchell 1998] is another related work that deals with using large unlabeled samples to boost the performance of a learning algorithm when only a small set of labeled examples is available.

In this article we will focus on text categorization tasks describing a new approach in which just category names are used as initial sets of discriminative words. The approach is quite general, and we believe it could also be exported to other NLP tasks. We leave for future work the verification that the approach can be generally applied to any categorization problem in which categories can be described by an initial set of discriminative features.

The TC task consists in assigning category labels to documents. In particular, we refer to topical categorization in which the features are the words of the documents. In the IL setting, a category C is described by providing a set of relevant features, termed an *intensional description* (ID), $id_c \subseteq V$, where V is the vocabulary. In addition, a training corpus $\mathcal{T} = \{t_1, t_2, \dots t_n\}$ of *unlabeled* texts is provided. Evaluation is performed on a separate test corpus of labeled documents, to which standard evaluation metrics can be applied.

The approach of categorizing texts based on lists of keywords has rarely been attempted in the literature [McCallum and Nigam 1999; Ko and Seo 2000; Liu et al. 2004; Ko and Seo 2004]. Some related references that are worthwhile mentioning Adami et al. [2003], in which they exploit the hierarchical organization of data (web directories) and a semi-automatic process (interleaved with human suggestions) to create the precondition for successfully training a supervised classifier; Godbole et al. [2004] propose an interactive workbench for text classification that combines the cognitive capability of humans with the power of automated learners. In particular, they propose a system based on active learning, starting with a small pool of labeled documents and a large pool of unlabeled documents. Finally, see Bekkerman [2003] for a good study on the effectiveness of sophisticated techniques for document representation using distributional clustering of words.

Several terms have been proposed for this approach—such as *TC by bootstrapping with keywords*, *unsupervised TC*, *TC by labelling words*—where the proposed methods fall (mostly) within the IL settings described here. Below, we survey the major contributions to the field that we found in the literature. It should be noted that it has been more difficult to define a common evaluation framework for comparing IL algorithms for TC, due to the subjective selection of seed IDs and to the lack of common IL test sets (see Section 4).

Expectation maximization. In the work of McCallum and Nigam [1999], the TC problem was approached by applying a bootstrap procedure, starting from category descriptions composed by a list of words. The learning process was performed in three steps: (i) a rule-based classifier based on keyword matching is defined to perform a preliminary categorization of the unlabeled examples; (ii) the produced labeled data is then used to train a naive Bayes classifier by means of an iterative re-estimation algorithm; (iii) the previous step is iterated until the likelihood reaches the optimal value. Experiments have shown that the algorithm improves at each step on a TC benchmark in the computer science domain, achieving F1 0.65, which is close to human agreement on the same task.

Vector space model. In a more recent work, Liu et al. [2004] approached the same problem by proposing a bootstrap schema relying on information retrieval and machine-learning techniques. First of all, the authors addressed the problem of collecting discriminative features for each category, and finding that it is not a trivial task. To provide a preliminary list of informative words for the task, they applied term clustering techniques to find a list of candidate discriminative words. Then, a lexicographer selected from that list a set of words for each category. The preliminary categorization step was performed by defining similarity metrics in the vector space model (VSM) [Salton and McGill 1983], where texts are represented by vectors, and the category descriptions are regarded as query vectors. For each text, the category with the highest similarity score[2] was then selected as a final classification result, and the collected labeled data was used to train a supervised naive Bayes classifier. The obtained classifier was then used to classify the whole collection of unlabeled documents. Optionally, this step is iterated following an iterative re-estimation schema. Evaluation was performed on four “easier” subsets of the 20newsgroups text collection (see Section 4.5), composed of four or five categories each. Authors compared the keyword-based approach to a supervised naive Bayes classifier in the same task, reporting slightly lower results.

Feature projection. The most recent attempt to appoach the keyword-based TC problem was reported by Ko and Seo [2004]. They proposed an algorithm to retrieve context clusters for each category (composed of text fragments similar to those containing the words in the IDs) to train a proposed text categorization using a feature projections (TCFP) classifier, which has the property of being more robust to noisy data than other learning algorithms such as k-NN or SVMs (see Ko and Seo [2002] for more details on the TCFP classifier). The authors reported very high results for three standard TC benchmarks, achieving performances comparable to those of supervised classifiers. On the other hand, they exploited labeled data to perform a chi-square-based feature selection. This clearly falls outside the IL setting, making their results incomparable to other IL methods.

In summary, it is possible to recognize a common structure behind these three works, based on a typical bootstrap schema [Yarowsky 1995; Collins and Singer 1999; Abney 2002]:

Step 1. *Initial similarity-based categorization.* This step was approached by applying some similarity criterion between the initial category seed and

[2]They also experimented with different thresholds of similarity values.

each unlabeled document. Similarity may be determined as a binary criterion, considering each seed keyword as a classification rule [McCallum and Nigam 1999], or by applying an IR style vector similarity measure. The result of this step is an initial categorization of (possibly a subset of) the unlabeled documents. In Ko and Seo [2004], term similarity techniques were exploited to expand the set of seed keywords, in order to improve the quality of the initial categorization.

Step 2. *Train a supervised classifier on the initially categorized set.* The output of step 1 is exploited to train an (extensional) supervised classifier. Different learning algorithms have been tested, including support vector machines (SVMs), naive bayes, nearest neighbors, and Rocchio (see Sebastiani [2002] for a survey of these algorithms applied to text categorization). Some works [McCallum and Nigam 1999; Liu et al. 2004] performed an additional iterative re-estimation algorithm over the training data, but reported rather small incremental improvements that do not seem to justify the additional effort.

3. INCORPORATING UNSUPERVISED LEARNING INTO THE BOOTSTRAP SCHEMA

As pointed out in Section 1, in this work we investigate some techniques to improve the specific weaknesses of the IL schema. In this section we show how the core step 1 of the IL schema—the initial categorization—can be improved by two unsupervised techniques. These techniques fit the IL setting, addressing its major constraints. The first technique exploits a similarity metric between category seeds (IDs) and documents, which defined in a latent semantic space, also known as a latent semantic index (LSI). Such unsupervised similarity among words (i.e., among features in the feature space) acts as a feature expansion technique, that is, it enables expanding the amount of information that is exploited from each seed feature, aiming at reducing the number of seeds needed. The second technique applies the unsupervised Gaussian mixture algorithm, which maps similarity scores to a principled classification probability value.[3] In particular, this step maps the similarity scores to class posterior probabilities, and therefore reduces sensitivity to keyword-dependent variations in scores, which is typically obtained through calibration over labeled examples in extensional learning.

3.1 Similarity in Latent Semantic Space

As explained above, step 1 of the IL scheme assesses a degree of "match" between the seed terms and the document to be classified. It is possible to first follow the intuitively appealing approach of Liu et al. [2004], in which IDs (category seeds) and documents are represented by vectors in the usual IR-style vector space model (VSM), and similarity is measured by the cosine function:

$$\mathrm{sim}_{vsm}(id_{\mathrm{c}}, \mathrm{t}) = \cos(\vec{id}_{\mathrm{c}}, \vec{\mathrm{t}}) \tag{1}$$

[3]Fitting a mixture of two distributions to the population of document scores was also used for the problem of online document filtering; see Zhang and Callan [2001] for details.

where $\vec{id}_c \in \mathrm{R}^{|\mathrm{V}|}$ and $\vec{t} \in \mathrm{R}^{|\mathrm{V}|}$ are the vectorial representations in the space $\mathrm{R}^{|\mathrm{V}|}$, respectively, of the category ID id_c and the text to be classified t (the instance), and V is the set of all the features (the vocabulary).

However, representing seeds and instances in a standard feature (i.e., word) space is severely affected in the IL setting by feature sparseness. In general, IDs are composed of short lists of words, possibly just a single word. Due to data sparseness, most documents do not contain any words in common with any category ID, which makes the seeds irrelevant for most documents. Furthermore, applying direct matching only for a few seed terms is often too crude, as it ignores the occurrences of other semantically related terms in the document (e.g., synonyms, hyponyms).

The problems above may be reduced by considering some form of similarity in the feature space, as it enables comparing additional terms with the original seeds. As mentioned in Section 2, Ko and Seo [2004] explicitly expanded the original category IDs with more terms, using a concrete query expansion scheme. We preferred using a similarity measure based on representing words and documents with LSI [Deerwester et al. 1990], as this is one of the most prominent techniques in IR to consider unsupervised similarity of terms. The dimensions of the latent semantic space are the most explicative principal components of the word-by-document matrix that describes the unlabeled dataset. This dimensionality reduction is accomplished by means of a singular value decomposition (SVD), which results in finding out the most informative dimensions, which will be used as a basis for defining a new space (i.e., the latent semantic space). SVD decomposes the term-by-document matrix $\mathrm{T}_{\mathrm{m}\times\mathrm{n}}$ into three matrixes $\mathrm{T}_{\mathrm{m}\times\mathrm{n}} = \mathrm{V}_{\mathrm{m}\times\mathrm{k}}\Sigma_{\mathrm{k}}\mathrm{U}^{\mathrm{T}}_{\mathrm{k}\times\mathrm{n}}$ where Σ_{k} is the diagonal $k \times k$ matrix, $k = min(m, n)$, containing the singular values of T. In applications, it is quite unusual to consider the full SVD. In fact, it is often sufficient to use a k'-*truncated* version, where $\Sigma_{\mathrm{k}'}$ is the diagonal $k' \times k'$ matrix containing the highest $k' \ll k$ singular values of T, and all the remaining elements set to 0. The parameter k' is the dimensionality of the latent semantic space (far smaller than $|V|$), and can be fixed in advance. Of course the truncated SVD is no longer an exact decomposition of the original matrix T, but the resulting approximate matrix is in a very useful sense its closest approximation that can be achieved by a matrix of rank k'. Finally, if only term vectors are considered, as in our present case, it is convenient to use an even more compact version $\mathrm{Term}_{\mathrm{m}\times\mathrm{k}'} = \mathrm{V}\Sigma_{\mathrm{k}'}$. This last form is the SVD decomposition that we exploit in the present article. In practice, as we will see in the next section, some scaling through idf can be applied.

Indeed, the dimensions coming out of the SVD process can be regarded as latent semantic domains.[4] In the latent semantic space, both coherent features (i.e., features that often co-occur in the same instances) and coherent instances

[4]Semantic domains have been used with a dual role in linguistic description. One role is characterizing word senses, typically as the semantic field of a word sense in a dictionary or lexicon (e.g., "crane" has senses in the domains of zoology and construction). A second role is to characterize texts, typically as a generic level of text categorization (e.g., for classifying news and articles). See Magnini et al. [2002] and Gliozzo et al. [2004] for more details.

(i.e., instances that share coherent features) are represented by similar vectors in the reduced dimensionality space. As a result, a document would be considered similar to a category ID if the seed terms and the document terms tend to co-occur overall in the given corpus.

The latent semantic vectors for IDs and documents were calculated by an empirically effective variation Gliozzo and Strapparava [2005] of the *pseudo-document* methodology to fold-in documents, originally suggested in Berry [1992]. Roughly speaking, each document or set of words is represented in the LSI space by averaging the LSI vectors of all the words contained in it, also using a tf.idf weighting scheme.

More formally:

$$\vec{t}_{lsi} = \vec{t}(\mathrm{I}^{\mathrm{IDF}}\mathrm{T}_{\mathrm{lsi}}) \tag{2}$$

where $\mathrm{I}^{\mathrm{IDF}}$ is a $k \times k$ diagonal matrix such that $I^{IDF}_{i,i} = IDF(w_i)$, $\vec{t}$ is represented as a row vector, and $IDF(w_i)$ is the *inverse document frequency* of w_i.

$$\mathrm{T}_{\mathrm{lsi}} = \mathrm{I}^{\mathrm{N}}\mathrm{V}\sqrt{\Sigma_{\mathrm{k'}}} \tag{3}$$

where I^{N} is a diagonal matrix such that $\mathrm{i}^{\mathrm{N}}_{\mathrm{i,i}} = \frac{1}{\sqrt{\langle \vec{\mathrm{w}}'_{\mathrm{i}}, \vec{\mathrm{w}}'_{\mathrm{i}} \rangle}}$, $\vec{w}'_i$ is the i^{th} row of the matrix $\mathrm{V}\sqrt{\Sigma_{\mathrm{k'}}}$. [5]

The similarity function sim_{lsi} is computed by the cosine metric, following Formula 1, where $\vec{id}_c$ and $\vec{t}$ are replaced by their latent semantic vectors. As shown in Section 4.2, using this nonsparse representation allows us to drastically reduce the number of seeds while significantly improving the performance of the initial categorization step.

3.2 The Gaussian Mixture (GM) Algorithm and the Initial Classification Step

Once there is a similarity function between category IDs and instances, a simple strategy is to base the classification decision (of step 1) directly on the similarity values obtained (as in Liu et al. [2004], for example). Typically, in step 1, IL approaches adopt a single-label classification approach, and classify each instance (document) to only one category. The chosen category is the one whose ID is most similar to the classified instance among all the categories. The subsequent training in step 2 yields a standard EL classifier, which can then be used to assign multiple categories to a document.

Using the output of the similarity function for classification directly is problematic, because the scales of similarity values obtained vary substantially across different categories. The variability in the range of similarity values is caused by variations in the number of seed terms per category and the levels of their generality and ambiguity. As a consequence, choosing the class with the highest absolute similarity value with respect to the classified text often leads to selecting a category whose similarity values tend to be generally higher,

[5]This technique is similar to the definition of the latent semantic space found in Deerwester et al. [1990]. The only difference in our formulation is that the vectors representing the terms are normalized by the matrix I^{N}, and then rescaled, according to their IDF value by matrix $\mathrm{I}^{\mathrm{IDF}}$. Note the analogy with the *tf idf* term-weighting schema [Salton and McGill 1983], widely adopted in information retrieval.

whereas another category could have been more similar to the classified instance if normalized similarity values had been used.

As a solution, we propose mapping the similarity values into class posterior probabilities using unsupervised estimation of Gaussian mixtures, which differentiates relevant and nonrelevant category information using statistics from unlabeled instances. Mixture models have been widely used in pattern recognition and statistics to approximate probability distributions. In particular, a well-known nonparametric method for density estimation is the so-called kernel method [Silverman 1986], which approximates an unknown density with a mixture of kernel functions, such as Gaussian functions. Under mild regularity conditions of the unknown density function, it can be shown that mixtures of Gaussians converge, in a statistical sense, to *any* distribution.

More formally, let $t \in \mathcal{T}$ be a document described by a vector $\vec{t} \in \mathrm{R}^{|V|}$ and let $id_c \subset V$ be the ID of category C; let $\mathrm{sim}(id_c, \mathrm{t}) \in \mathrm{R}$ be a similarity function among instances and IDs, with the only expectation that it monotonically increases according to the "closeness" of id_c and t (as in Section 3.1). For a category C, GM induces a mapping from the similarity scores between its ID and any instance $t_j \in \mathcal{T}$, $\mathrm{sim}(id_c, \mathrm{t_j})$, into the probability of C given the text t_j, $P(C|t_j)$. To achieve this goal, GM performs the following operations:

(i) it computes the set $\mathcal{S}_c = \{\mathrm{sim}(id_c, \mathrm{t_j}) | \mathrm{t_j} \in \mathcal{T}\}$ of the similarity scores between the ID id_c of the category C and all the instances t_j in the unlabeled training set $\mathcal{T}$;

(ii) it induces from the empirical distribution of values in $\mathcal{S}_c$ a Gaussian mixture distribution composed of two "hypothetic" distributions $\mathcal{C}$ and $\overline{\mathcal{C}}$, which are assumed to describe, respectively, the distributions of similarity scores for positive and negative examples (selecting $\mathcal{C}$ the one with the higher mean). Thus the output of this step includes the category and the complementary category distributions;

(iii) it estimates the conditional probability $P(C|\mathrm{sim}(id_c, \mathrm{t_j}))$ by applying the Bayes theorem on the distributions $\mathcal{C}$ and $\overline{\mathcal{C}}$.

These steps are explained in more detail below.

Step (i) is straightforward, and is implemented according to Section 3.1. The core idea of the algorithm is in step (ii). Since we do not have labeled training examples, we can only obtain the set $\mathcal{S}_c$, which includes the similarity scores for all examples mixed together, both positive and negative, without knowing which examples are positive and which are negative. We assume, however, that similarity scores that correspond to positive examples are drawn from one distribution, $P(\mathrm{sim}(id_c, \mathrm{t_j})|\mathrm{C})$, while the similarity scores that correspond to negative examples are drawn from another distribution, $P(\mathrm{sim}(id_c, \mathrm{t_j})|\overline{\mathrm{C}})$. The observed distribution of similarity values in $\mathcal{S}_c$ is thus assumed to be a mixture of the above two distributions, which are recovered by the GM estimation.[6]

[6]We recall that a Gaussian is fully specified by the mean μ and variance σ parameters. The mean and variance parameters (μ, σ and $\overline{\mu}$, $\overline{\sigma}$) of the two distributions $\mathcal{C}$ and $\overline{\mathcal{C}}$ are estimated by the rather simple application of the EM algorithm for Gaussian mixtures, as summarized in Appendix 6 (formulas 11 to 13). In addition to the GM parameters, the marginal probabilities $P(C)$, $P(\overline{C})$ (area

The probabilistic mapping estimated in step (iii) for a category C given an instance t is computed by applying Bayes rule:

$$\begin{aligned} P(C|t) &= P(C|\mathrm{sim}(id_c, t)) \\ &= \frac{P(\mathrm{sim}(id_c, t)|C)P(C)}{P(\mathrm{sim}(id_c, t)|C)P(C) + P(\mathrm{sim}(id_c, t)|\overline{C})P(\overline{C})} \end{aligned} \tag{4}$$

where $P(\mathrm{sim}(id_\mathrm{c}, \mathrm{t})|\mathrm{C})$ is the value of the Probability Density Function (*PDF* of $\mathcal{C}$ at the point $\mathrm{sim}(id_\mathrm{c}, \mathrm{t})$, $P(\mathrm{sim}(id_\mathrm{c}, \mathrm{t})|\overline{\mathrm{C}})$ is the value of the *PDF* of $\overline{\mathcal{C}}$ at the same point (see Appendix 6, formula 7), and $P(C)$ and $P(\overline{C})$ are class probabilities, $P(C) + P(\overline{C}) = 1$.

Finally, following the single-labeled categorization setting of step 1 in the IL scheme, the most likely category is assigned to each instance, that is, $argmax_{C_i} P(C_i|t)$. In this unsupervised classification step, we regard the classification task as a single-label problem, even if the dataset itself has multiclass labels. Indeed, the algorithm is designed in a way that multiclass problems are decomposed into a set of binary classification tasks in the intermediate steps, and then recombined at the end. As far as the GM algorithm is concerned, we use only seeds from the positive classes to train the system. Therefore, the GM approach is in some sense similar to a one-class learning algorithm, where features are used instead of examples. Then, at the next classification step, for each class we create a binary classifier, adopting a one vs. all schema, where all the positive examples for all classes, except the one under consideration, are regarded as negative examples.

Figure 1 illustrates the mapping induced by GM from the empirical mixture distribution: dotted lines describe the PDFs estimated by GM for $\mathcal{C}$, $\overline{\mathcal{C}}$. Their mixture approximates the empirical distribution ($\mathcal{S}_c$) (in step (ii)). The continuous line is the mapping induced in step (iii) of the algorithm from similarity scores between instances and IDs (x axis) to the probability of the instance belonging to the category (y axis).

3.3 Extensional Supervised Categorization

As an extensional supervised learning device, we used the SVM implementation described in Joachims [1999], exploiting the same feature set adopted in the first step (i.e., using the representations in the classical VSM) and the standard parameter settings for the linear kernel. The system is trained by exploiting the categories assignments produced as an output of the first categorization step, and then applied to the test data to perform the final categorization step. Thus a document is taken as positive example for the single category to which it was assigned, and as a negative example for all other categories [Rifkin and Klautau 2004]. We did not perform any iterative re-estimation step, because this did not produce any significant improvement during our preliminary experiments. As an option, the final categorization has been approached by either defining a set of binary filtering problems (allowing the TC system to assign more than one category label to each document) or by implementing a one-versus-all

under each Gaussian) are computed by formula 11 in Appendix 6.

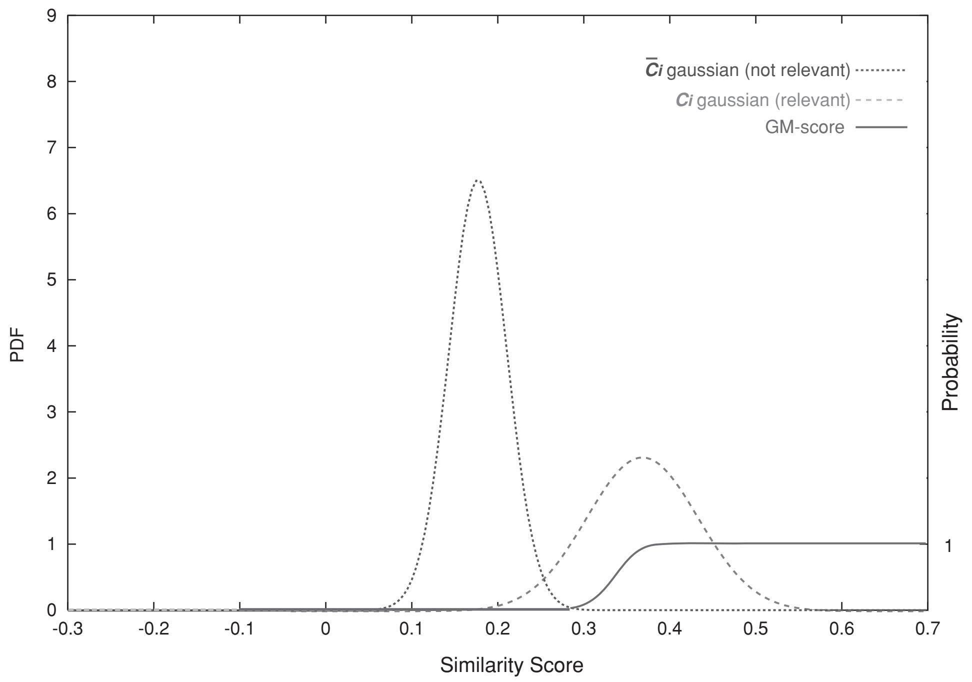

Fig. 1. Mapping induced by GM for the category *sci.med* in the 20newsgroups dataset.

classification schema (i.e., a single category is assigned to each news article, picking the category with the highest classification score). To validate the quality of our supervised classifier, we evaluate its performance on two standard TC benchmarks, achieving state-of-the-art performances in both tasks (i.e., F1 0.90 for the Reuters task and F1 0.88 for the 20-newsgroups task).

3.4 Summary of the Generalized Bootstrapping Algorithm

Step 1.a: *Latent Semantic Space.* Instances and IDs of categories (the seeds) are represented by vectors in a latent semantic space. As an option, the algorithm can work with the classical VSM using the original feature space. Similarity scores between IDs and instances are computed by the cosine measure.

Step 1.b: *GM.* The mapping functions $P(C_i|t_j)$ for each category C_i, conditioned on instances $t_j \in \mathcal{T}$, are induced by the GM algorithm. To that end, an EM algorithm estimates the parameters of the two component distributions of the observed mixture, which correspond to the distributions of similarity values for positive and negative examples. This step is performed by considering all the unlabeled training data, and it is repeated for each category C_i. As a baseline, the GM mapping can be avoided, considering only the previous step.

Step 1.c: *Categorization.* Each instance in the unlabeled training set is classified to the most probable category—$argmax_{C_i} P(C_i|t_j)$.

Table I. Initial Seeds for the 20newsgroups and Reuters-10 Datasets

20newsgroups		Reuters-10	
Categories	Seeds	Categories	Seeds
alt.atheism	atheism#n	acq	acquisition#n
comp.graphics	graphics#n	corn	corn#n
comp.os.ms-windows.misc	microsoft#n windows#n	crude	crude#n
comp.sys.ibm.pc.hardware	ibm#n pc#n	earn	earn#v
comp.sys.mac.hardware	mac#n	grain	grain#n
comp.windows.x	x-windows#n	interest	interest#n
misc.forsale	sale#n	money-fx	money#n
rec.autos	car#n	ship	ship#n
rec.motorcycles	motorcycle#n	trade	trade#n
rec.sport.baseball	baseball#n	wheat	wheat#n
rec.sport.hockey	hockey#n		
sci.crypt	cryptography#n		
sci.electronics	electronics#n		
sci.med	medicine#n		
sci.space	space#n		
soc.religion.christian	christian#n christian#a		
talk.politics.guns	gun#n		
talk.politics.mideast	mideast#n		
talk.politics.misc	politics#n		
talk.religion.misc	religion#n		

Step 2: *Bootstrapping an Extensional Classifier.* An EL classifier based on SVM is trained on the set of automatically labeled instances resulting from step 1.c.

4. EVALUATION

4.1 Intensional Text Categorization Datasets

Even though some typical datasets have been used in the TC literature [Sebastiani 2002], the datasets used for IL learning were not standard. Often there is not sufficient clarity regarding details such as the exact version of the corpus used and the training/test splitting. Furthermore, the choice of categories was often not standard: Ko and Seo [2004] omitted four categories from the 20newsgroups dataset, while Liu et al. [2004] evaluated their method on four separate subsets of the 20newsgroups, each containing only four to five categories. Such issues make it rather difficult to thoroughly compare different techniques, yet we have conducted several comparisons in Section 4.5 below. In the remainder of this section we clearly state the corpora used in our experiments and the preprocessing steps performed on them. For up-to-date categorization performance on these corpora, Gabrilovich and Markovitch [2007] report very good numbers on both the 20newsgroups and the Reuters data.

Preprocessing. In both datasets we used (i.e., 20newsgroups and Reuters-10), we tagged the texts for part-of-speech (PoS) and represented the documents by the frequency of each PoS-tagged lemma,[7] considering only nouns, verbs, adjectives, and adverbs. Here we use the notation `word#pos` to indicate the term with its part-of-speech (e.g., the noun "house" is represented as `house#n`).

This allows us to reduce ambiguity and sparseness at the same time. In fact, different morphological variations are collapsed and polysemy introduced by different syntactic roles is avoided.

20newsgroups. The 20newsgroups dataset is a collection of newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups [Lang 1995]. The collection is available at `www.ai.mit.edu/people/jrennie/20Newsgroups`. As suggested by this Web site, we used the "bydate" version: the corpus (18,846 documents) is sorted by date and divided in advance into a training (60%, 11,308 documents) set and a chronologically following test set (40%, 7,538 documents), so there is no randomness in train/test set selection. It does not include cross-posts (duplicates), and, more importantly, does not include nontextual newsgroup-identifying headers, which often help classification (Xref, Newsgroups, Path, Followup-To, Date), keeping from the header only the subject line that was treated as words in the body.

We first report results using *initial seeds* for the category IDs, which were selected using only the words in the category names, with some trivial transformations (i.e., `cryptography#n` for the category `sci.crypt`, `x-windows#n` for the category `comp.windows.x`). We also tried to avoid "overlapping" seeds, that is, for the categories `rec.sport.baseball` and `rec.sport.hockey` the seeds are only {`baseball#n`} and {`hockey#n`} respectively, and not {`sport#n, baseball#n`} and {`sport#n, hockey#n`}.[8]

Reuters-10. We used the top ten categories (Reuters-10) in the *Reuters-21578* collection Aptè split.[9] The complete Reuters collection includes 12,902 documents for 90 categories, with a fixed splitting between training and test data (70/30%). Both the Aptè and Aptè-10 splits are often used in TC tasks, as surveyed in Sebastiani [2002]. The Reuters-10 Aptè split is a restriction of Reuters-21578 ModAptè to the ten categories with the highest generality: `Earn, Acquisition, Money-fx, Grain, Crude, Trade, Interest, Ship, Wheat`, and `Corn`. The final dataset includes 9,296 documents (6,507 training/2,789 test). The initial seeds are only the words appearing in the category names.

[7]We used TreeTagger, a tool for annotating text with part-of-speech and lemma information, freely available at `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger`. It deals with many languages, for example, English, Italian, German, French, Dutch, Spanish, Bulgarian, and Russian.

[8]One could propose as a guideline for seed selection those seeds that maximize their distances in the latent semantic space. From this perspective, the latent semantic vectors built from {`sport#n, baseball#n`} and {`sport#n, hockey#n`} are closer than the vectors that represent {`baseball#n`} and {`hockey#n`}. Note that this is the reason for the slight initial performance decrease in the learning curve in Figure 2 below.

[9]Available at `http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html`. In particular, `http://kdd.ics.uci.edu/databases/reuters21578/README.txt` describes the distribution of this text collection.

Table II. Impact of LSI and GM on Step 1 (*initial*) and step 2 (+ *bootstrap*)

			Reuters	20Newsgroups
	LSI	GM	F1-micro	F1-micro
initial	no	no	0.38	0.25
+ *bootstrap*			0.42	0.28
initial	no	yes	0.41	0.30
+ *bootstrap*			0.46	0.34
initial	yes	no	0.46	0.50
+ *bootstrap*			0.47	0.53
initial	yes	yes	0.58	0.60
+ *bootstrap*			**0.74**	**0.65**
Extensional Learning (Sec. 4.4)			0.90	0.79

Latent Semantic Space and GM. We induced the LSI (and the corresponding idf weighting) only from the training part and considered the first 400 dimensions. As far as GM is concerned, we calculated it only on the training part of the datasets. From a machine-learning point of view, it would be legitimate to run the SVD on the full corpus (i.e., training and test), the LSI being a completely unsupervised technique (i.e., it does not take into account the category labels on the documents). However, from an applicative point of view (i.e., at a given time we have a set of documents at disposal, and then we classify documents coming later) it is more sensible to have the LSI space built on the training part only. If we run the SVD on the full corpus, the performance figures increase by almost four points.

4.2 The Impact of LSI Similarity and GM on IL Performance

In this section we evaluate the incremental impact of LSI similarity and the GM algorithm on IL performance. When avoiding both techniques, the algorithm uses the simple cosine-based method over the original feature space, which can be considered as a baseline (similar to the method of Liu et al. [2004]). We first report results using only the names of the categories as initial seeds.

In all the experiments in this article we use F1 measure,[10] in particular microaveraging: *precision* and *recall* are obtained by summing over all individual categorization decisions [Sebastiani 2002]. Instead, macroaveraging means that precision and recall are first evaluated locally for each category and then averaged over the results of the different categories. Microaverage gives equal weight to every document, whereas macroaverage gives equal weight to every category, regardless of its frequency.

Table II displays the microaverage F1 measure for the 20newsgroups and Reuters datasets, with and without LSI and with and without GM. Just for information, in Table III we report the macroaverage F1 measure of the last three lines of Table II. The performance figures show the incremental benefit of both LSI and GM. In particular, when using just initial seeds and not exploiting LSI similarity the performance is heavily penalized, as can be expected given

[10] $F1 = \frac{2 \times precision \times recall}{precision + recall}$.

Table III. F1-Macroaverage Performance

	Reuters	20 Newsgroups
initial	0.57	0.58
+ *bootstrap*	0.61	0.61
Extensional Learning	0.83	0.78

the small amount of information provided by the initial seeds. Bold figures in Table II display the performance after bootstrapping using both LSI and GM.

Finally, the bootstrapping step of the algorithm (Step 2) exploits the initially classified instances to train a supervised text categorization classifier based on SVMs. It is worthwhile noting that the increment in performance after bootstrapping is generally higher when GM and LSI are incorporated, thanks to the higher quality of the initial categorization which was used for training.

4.3 Learning Curves for the Number of Seeds

This section evaluates change in accuracy as a function of the number of seeds. The experiment was performed for the 20 newsgroups corpus using both the LSI and the classical VSM (using GM in both cases, but without applying bootstrapping). Additional seeds, beyond the category names, were identified by two lexicographers. For each category, the lexicographers were provided with a list of 100 candidate seed terms produced by the LSI similarity function and applied to the category name (one list of 100 candidate terms for each category). From these lists the lexicographers selected the words that were judged as significantly related to the respective category, keeping the order from the LSI similarity and picking on average 40 seeds per category.

As shown in Figure 2, the learning curve using LSI dramatically outperformed the one using classical vector space. As can be expected, when using the classical vector space (no LSI generalization), the curve improves quickly with a few more terms. More surprisingly, with LSI similarity the best performance is obtained using the minimal initial seeds of the category names, while adding more seeds degrades performance. This might suggest that category names tend to be highly indicative for the intensional meaning of the category, and therefore adding more terms introduces additional noise. Further research is needed to find out whether other methods for selecting additional seed terms might yield incremental improvements. The current results, however, emphasize the benefit of utilizing LSI and GM. These techniques obtain good performance for IL (see comparisons in Section 4.5) using only the category names as seeds, allowing us to skip the difficult phase of manually collecting a larger number of seeds.

4.4 Extensional vs. Intensional Learning

A major point of comparison between IL and EL is the amount of supervision required to obtain a certain level of performance. To this aim, we used the SVM-based classifier [Vapnik 1995] implemented for the bootstrapping phase in a strictly supervised setting, using the labelled data; we draw its learning curves as a function of the percentage of the training set size (Figure 3). In the case of 20newsgroups, to achieve the 0.65 F1 performance of IL, the supervised

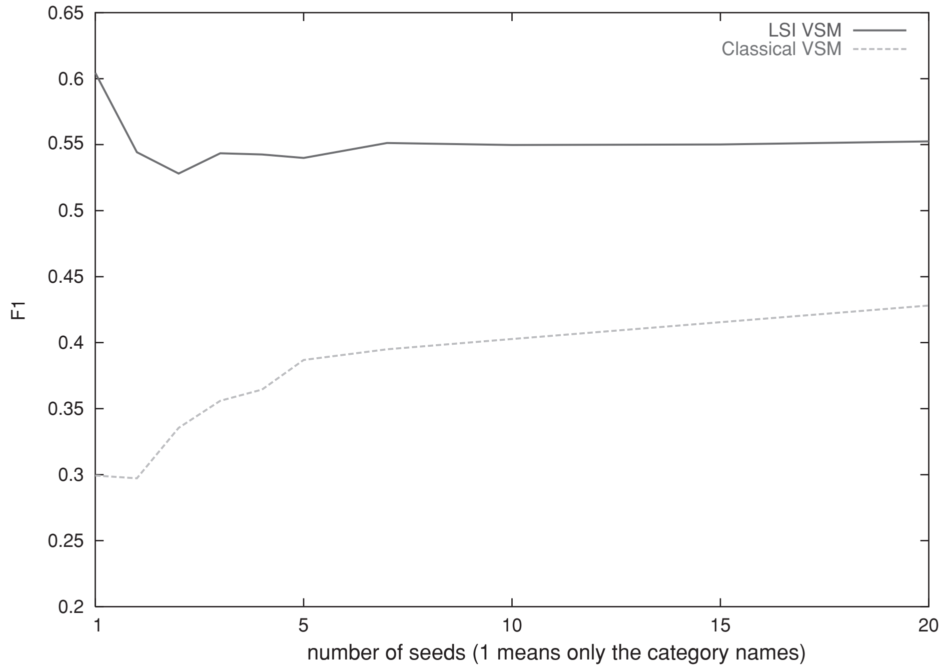

Fig. 2. Learning curves on initial seeds for 20newsgroups, LSI, and classical VSM (no LSI). Point 1 on the x-axis corresponds to the initial seeds, which include only the category names (but possibly more than strictly one term, e.g., when the category name consists of two words).

settings requires about 3,200 documents (about 160 texts per category), while our IL method requires only the category name. Reuters-10 is an easier corpus, therefore EL rather rapidly achieves a high performance. But even here using just the category name is equal on average to labeling 70 documents per category (700 in total), yielding an F1 value of 0.74. These results suggest that IL may provide an appealing cost-effective alternative in practical settings when suboptimal accuracy suffices, or when it is too costly or impractical to obtain sufficient amounts of labeled training sets.

It should also be stressed that when using the complete standard labeled training corpus of each dataset, state-of-the-art EL outperforms our best IL performance. This result deviates notably from the flavor of previous IL literature, which reported almost comparable performance relative to EL. As mentioned earlier, the method of Ko and Seo [2004] (as we understand it) utilizes labeled examples for feature selection, and therefore cannot be compared with our strict IL setting. As for the results in Liu et al. [2004], we conjecture that their comparable performance for IL and EL may not be sufficiently general, for several reasons: the easier classification task (four subsets of 20newsgroups of four to five categories each); the use of the usually weaker naive Bayes as the EL device; and the use of clustering as an aid for selecting the seed terms from the 20newsgroups subsets, which might not scale up well when applied to a large number of categories of varying size. Thus, our current conclusion is that IL is still weaker than exhaustively trained EL.

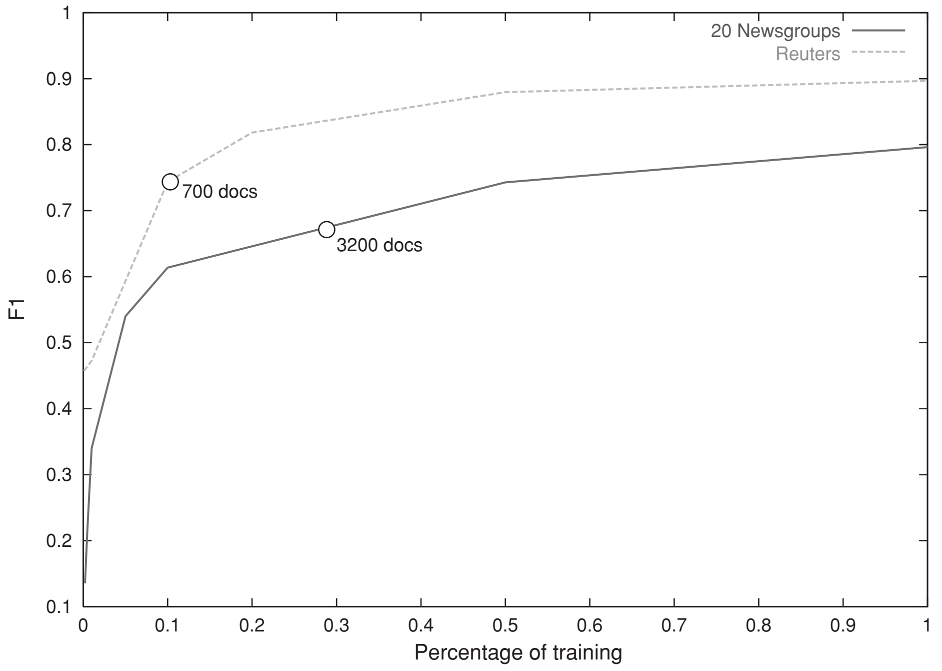

Fig. 3. *Extensional* learning curves as a function of percentage of the standard training set. The two circles specify the F1 performance of our IL method and indicate the number of documents needed to achieve this performance level by EL.

4.5 Comparisons with Other IL Algorithms

As mentioned earlier, it is not easy to conduct a thorough comparison with other algorithms in the literature. Most datasets used for IL are either not available publicly [McCallum and Nigam 1999] or are composed of somewhat arbitrary subsets of standard datasets. Another crucial aspect is the particular choice of the seed terms selected to compose a category seed, which significantly affects the overall performance of the algorithm.

As a baseline system, we implemented a rule-based approach in the spirit of McCallum and Nigam [1999]. It is based on two steps. First, all the documents in the unlabeled training corpus containing at least one word in common with one and only one category ID are assigned to the respective class. Second, a supervised classifier based on SVM is trained on the labeled examples. Finally, the supervised classifier is used to perform the final categorization step on the test corpus. Table IV reports the F1 measure of our replication of this method, using the category name as seed, which is substantially lower than the performance of the method we presented in this article.

We also tried to approximate two of the nonstandard datasets used in Liu et al. [2004]. There are some differences in that we used sequential splitting (70/30) rather than random splitting, and did not apply any feature selection. This setting might be somewhat more difficult than the original one. Table V displays the performance of our approach in comparison to the results reported

Table IV. Rule-based Baseline Performance (F1-micro)

	Reuters	20Newsgroups
	0.34	0.30
+ *bootstrap*	0.42	0.47

Table V. Accuracy of the Liu et al. [2004] Dataset (i.e., basically the 20newsgroups' top hierarchy): four "REC", four "TALK", four "SCI" and five "COMP" newsgroup categories.

	Our	IDs per cat.	Liu et al.	IDs per cat.
REC	0.94	1	0.95	5
TALK	0.80	1	0.80	20
SCI	0.92	1	0.93	20
COMP	0.81	1	0.71	15

in Liu et al. [2004]. Following the evaluation metric adopted in that paper, here we report accuracy instead of F1. For each dataset, Liu et al. [2004] reported several results varying the number of seed words (from 5 to 30), as well as varying some heuristic thresholds, so in the table we report their best results. Notably, our method obtained comparable accuracy by using just the category name as ID for each class instead of multiple seed terms. This result suggests that our method enables us to avoid the somewhat cumbersome process of manually collecting a substantial number of additional seed words, exploiting a combination of clustering and manual selection.

5. EXPERIMENTS WITH WIKIPEDIA AND WORDNET DOMAINS

After experimenting on some standard datasets in the text categorization community, we wanted to test the IL technique in a somewhat different setting. The idea was to perform an experiment in an adverse case of a *small* dataset with a relatively *large* number of categories. We have seen that the IL technique uses only the name of the categories as initial seeds, where a key point is to have meaningful names for representing the categories. Given our past work on the semantic domains [Gliozzo et al. 2004; Magnini et al. 2001, 2002] particularly in exploiting WordNet Domains,[11] from this resource we selected a set of about 40 disjoint labels. As we saw from our past work on word sense disambiguation [Magnini et al. 2002; Gliozzo et al. 2004], this set allows a good level of abstraction without losing relevant information and, in addition, it overcomes the problem of applying learning techniques to domains not well enough represented in texts.

We then extracted a text categorization benchmark from Wikipedia, which reflects the high-level domains provided by the WordNet Domains taxonomy. To this aim, we mapped the WordNet Domains labels into the corresponding most similar categories in Wikipedia, and collected the extended abstracts of the

[11]WordNet Domains is freely available at `http://wndomains.itc.it`. It is a lexical resource that attempts a systematization of relevant aspects in domain organization and representation. WordNet Domains is an extension of WordNet [Fellbaum 1998], in which each synset is annotated with one or more domain labels [Magnini and Cavaglià 2000].

Table VI. Wikipedia Categories and WordNet Domains Labels

WN Domain	Wikipedia Category	#	WN Domain	Wikipedia Category	#
administration	administration	16	law	law	210
agriculture	agriculture	353	linguistics	linguistics	463
alimentation	food_and_drink	22	literature	literature	125
anthropology	anthropology	254	mathematics	mathematics	60
archaeology	archaeology	66	medicine	medicine	5
architecture	architecture	210	military	military	11
art	visual_arts	9	pedagogy	pedagogy	136
artisanship	crafts	53	philosophy	philosophy	30
astrology	astrology	23	physics	physics	208
astronomy	astronomy	31	play	play	25
biology	biology	222	politics	politics	122
body_care	hygiene	48	psychology	psychology	473
chemistry	chemistry	38	publishing	publishing	74
commerce	commerce	92	religion	religion	49
computer_science	computer_science	102	sexuality	sexuality	34
earth	earth	42	sociology	sociology	631
economy	economics	25	sport	olympic_sports	62
engineering	engineering	93	telecommunication	telecommunications	224
fashion	fashion	30	tourism	tourism	83
history	history	32	transport	transportation	156
industry	industry	98	veterinary	veterinary_medicine	76
				TOTAL	5116

corresponding Wikipedia articles. The general criterion adopted for the mapping was matching the category name itself. When mappings were not obvious, we manually inspected the Wikipedia categories to find a category reflecting approximately the same intended meaning. Table VI reports the number of examples collected and the mapping between Wikipedia and WordNet Domains categories.

DB-pedia[12] distributes the extended abstracts of Wiki pages, labeled with the corresponding Wiki categories. These are documents of approximately the same length as the documents in the 20newsgroups and Reuters datasets. The quality of the obtained dataset is not as high as the others for two main reasons. First, Wikipedia categories have been assigned manually by Wikipedia users without adopting any strict policy or any predefined category set. Second, the distribution of texts into different categories does not reflect our intuition. For example, the category anthropology is 50 times bigger than medicine. This makes this dataset quite difficult for any learning algorithm, so we used it as a stress test to measure the robustness of our method. Overall, we obtained a dataset of 5116 documents with 42 categories (domains) randomly split into 80% training and 20% test. In principle, a document may belong to more than one Wiki category, even if multiple labels occur quite rarely in the dataset (e.g., about 100 documents have two labels). Tokenization and POS tagging were applied to the documents. As initial seeds, we used just the names of the categories (Table VI), as for the other experiments. Table VII shows the technique's performance (F1 measure).

[12]http://wiki.dbpedia.org/Downloads32

Table VII. Performance (F1-micro) on Wikipedia DataSet

Wikipedia			
	LSI	GM	F1-micro
baseline	no	no	0.32
	no	yes	0.36
	yes	no	0.47
LSI and GM	yes	yes	0.50
+ *bootstrap*			0.56
Extensional Learning			0.70

For comparison, the extensional classification (SVM) on the same dataset scores 0.70 F1. To obtain a complete picture, analyzing the learning curves, the equivalence point IL vs. EL is about 740 documents, which corresponds to about 20% of the training set. These results are consistent with those we obtained for the 20newsgroups and Reuters datasets.

6. CONCLUSIONS

We presented a general bootstrapping algorithm for feature-based supervision in text categorization, which we call *intensional learning*. Our algorithm utilizes a generalized similarity measure based on latent semantic spaces and a Gaussian mixture algorithm as a principled method to scale similarity scores into probabilities. Both techniques address inherent limitations of the IL setting, and leverage unsupervised information from an unlabeled corpus.

We applied and evaluated our algorithm on some text categorization tasks and demonstrated the contributions of the two techniques. In particular, we obtained competitive performance using only the category names as initial seeds. This minimal information per category, when exploited by the IL algorithm, is shown to be equivalent to labeling about 70–160 training documents per-category for state-of-the-art extensional learning.

Future work is needed: on one hand to investigate optimal procedures for collecting seed features and to find out whether additional seeds might still contribute to better performance, while on the other hand re-evaluating the technique for the case in which one does not have a prespecified dataset to be classified. Furthermore, it would be very interesting to explore optimal combinations of intensional and extensional supervision, provided by the user as a combination of seed features *and* labeled examples. Finally, we would like to experiment with our algorithm beyond text categorization tasks and to try its effectiveness on some categorization problems in which categories are described by initial sets of discriminative features and an unlabeled training dataset is provided. We plan to apply the IL technology to a variety of application scenarios, where the lack of labelled examples is an intrinsic limitation, such as content-based user modeling, ontology population, and sentiment analysis.

APPENDIX: THE EXPECTATION MAXIMIZATION ALGORITHM FOR THE GAUSSIAN MIXTURE MODEL

The framework provided by the Gaussian mixture algorithm described in Section 3.2 requires the definition of an algorithm to estimate the parameters characterizing the required PDFs. In fact, the observed distribution of similarity values in $\mathcal{S}_i$ is assumed to be decomposed into the weighted sum (a mixture) of two components, (i.e., two Gaussian functions), describing, respectively, relevant and nonrelevant texts. We recall that under mild regularity conditions of the unknown density function, it can be shown that mixtures of Gaussians converge, in a statistical sense, to *any* distribution.

The expectation maximization (EM) algorithm for Gaussian mixture (GM) models [Redner and Walker 1984] allows us to efficiently perform this operation. Details of this algorithm are reported in this appendix. While in the present work the considered mixture is formed by only two components, in this Appendix we give details for a general mixture composed by $m \geq 2$ Gaussians.

A *Gaussian mixture* allows us to represent every smooth PDF as a linear combination of normal distributions of the type in Formula 5[13]:

$$p(x|\theta) = \sum_{j=1}^{m} a_j G(x, \mu_j, \sigma_j) \tag{5}$$

with

$$a_j \geq 0 \text{ and } \sum_{j=1}^{m} a_j = 1 \tag{6}$$

and

$$G(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{7}$$

where $\theta = \langle a_1, \mu_1, \sigma_1, \ldots, a_m, \mu_m, \sigma_m \rangle$ is a parameter list describing the Gaussian mixture.

Each component j is univocally determined by its weight a_j, its mean μ_j, and its variance σ_j. Weights also represent the areas of each component, that is, its total probability.

The GM algorithm here defined to perform the preliminary categorization step exploits a Gaussian mixture to approximate the empirical distribution of the similarity scores, (e.g., $\mathcal{S} = \{\mathrm{sim}(id_c, t_j) | t_j \in \mathcal{T}\}$ being id_c the intensional description of the category C). The goal of the Gaussian mixture algorithm is to find the GM that maximizes the likelihood on the empirical data, where the likelihood function is evaluated by Formula 8.

$$L(\mathcal{T}, C, \theta) = \prod_{t \in \mathcal{T}} p(sim(id_c, t)|\theta) \tag{8}$$

[13]In the following formulas, we use a compact symbol notation: with respect to Section 3.2, please consider that $a_j = P(C_j)$, that is, the weights of the mixture correspond to the marginal probabilities of C and $\overline{C}$.

More formally, the EM algorithm for GM models explores the space of parameters in order to find the set of parameters θ such that the maximum likelihood criterion is satisfied:

$$\theta_D = \underset{\theta'}{\operatorname{argmax}}\, L(\mathcal{T}, C, \theta') \tag{9}$$

This condition ensures that the model obtained fits the original data as much as possible. Estimation of parameters is the only information required in order to evaluate domain relevance for texts using the Gaussian mixture algorithm. The EM algorithm for Gaussian mixture models [Redner and Walker 1984] allows us to perform this operation efficiently.

The strategy followed by the EM algorithm is to start from a random set of parameters θ_0, that has a certain initial likelihood value L_0, and then iteratively change them in order to augment likelihood at each step. To this aim, the EM algorithm exploits a growth transformation of the likelihood function $\Phi(\theta) = \theta'$ such that $L(\mathcal{T}, C, \theta) \leq L(\mathcal{T}, C, \theta')$. Applying this transformation iteratively, starting from θ_0, a sequence of parameters is produced, until the likelihood function achieves a stable value (i.e., $L_{i+1} - L_i \leq \epsilon$). In our settings the transformation function Φ is defined by the following set of equations, in which all the parameters must be solved together.

$$\Phi(\theta) = \Phi(\langle a_1, \mu_1, \sigma_1, a_2, \mu_2, \sigma_2\rangle) = \langle a'_1, \mu'_1, \sigma'_1, a'_2, \mu'_2, \sigma'_2\rangle \tag{10}$$

$$a'_j = \frac{1}{|\mathcal{T}|}\sum_{k=1}^{|\mathcal{T}|} \frac{a_j G(sim(id_c, t_k), \mu_j, \sigma_j)}{p(sim(id_c, t_k), \theta)} \tag{11}$$

$$\mu'_j = \frac{\sum_{k=1}^{|\mathcal{T}|} sim(id_c, t_k) \cdot \frac{a_j G(sim(id_c, t_k), \mu_j, \sigma_j)}{p(sim(id_c, t_k), \theta)}}{\sum_{k=1}^{|\mathcal{T}|} \frac{a_j G(sim(id_c, t_k), \mu_j, \sigma_j)}{p(sim(id_c, t_k), \theta)}} \tag{12}$$

$$\sigma'_j = \frac{\sum_{k=1}^{|\mathcal{T}|} (sim(id_c, t_k) - \mu'_j)^2 \cdot \frac{a_i G(sim(id_c, t_k), \mu_i, \sigma_i)}{p(sim(id_c, t_k), \theta)}}{\sum_{k=1}^{|\mathcal{T}|} \frac{a_j G(sim(id_c, t_k), \mu_j, \sigma_j)}{p(sim(id_c, t_k), \theta)}} \tag{13}$$

The EM algorithm was used to estimate the parameters to describe distributions for relevant and nonrelevant texts. This learning method is totally unsupervised. Estimated parameters has been used to estimate relevance values by Formula 4 in all the experiments reported in this article.

REFERENCES

Abney, S. 2002. Bootstrapping. In *Proceeding of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*.

Abney, S. 2004. Understanding the Yarowsky algorithm. *Comput. Linguist. 30*, 3.

Adami, G., Avesani, P., and Sona, D. 2003. Bootstrapping for hierarchical document classication. In *Proceedings of the 12th International Conference on Information and Knowledge Management (CIKM'03)*.

Barendregt, H. 1984. *The Lambda Calculus: Its Syntax and Semantics*. North Holland, Amsterdam.

BEKKERMAN, R. 2003. Distributional clustering of words for text categorization. M.S. thesis, Technion-Israel Institute of Technology.

BERRY, M. 1992. Large-scale sparse singular value computations. *Int. J. Supercomput. Appl. 6*, 1, 13–49.

BLUM, A. AND MITCHELL, T. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*. 92–100.

COLLINS, M. AND SINGER, Y. 1999. Unsupervised models for named entity classification. In *Proceedings of the EMNLP'99 Conference*.

DEERWESTER, S., DUMAIS, S., FURNAS, G., LANDAUER, T., AND HARSHMAN, R. 1990. Indexing by latent semantic analysis. *J. Amer. Soc. Inform. Sci*.

FELLBAUM, C. 1998. *WordNet. An Electronic Lexical Database*. MIT Press, Cambridge, MA.

GABRILOVICH, E. AND MARKOVITCH, S. 2007. Harnessing the expertise of 70,000 human editors: Knowledge-based feature generation for text categorization. *J. Machine Learn. Resear. 8*, 2297–2345.

GLIOZZO, A. AND STRAPPARAVA, C. 2005. Domains kernels for text categorization. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL'05)*.

GLIOZZO, A., STRAPPARAVA, C., AND DAGAN, I. 2004. Unsupervised and supervised exploitation of semantic domains in lexical disambiguation. *Comput. Speech Lang. 18*, 275–299.

GLIOZZO, A., STRAPPARAVA, C., AND DAGAN, I. 2005. Investigating unsupervised learning for text categorization bootstrapping. In *Proceedings of the Joint Conference on Human Language Technology / Empirical Methods in Natural Language Processing (HLT / EMNLP)*.

GODBOLE, S., HARPALE, A., SARAWAGI, S., AND CHAKRABARTI, S. 2004. Document classication through interactive supervision of document and term labels. In *Proceedings of the 15th European Conference on Machine Learning (ECML) and the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*.

JOACHIMS, T. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods: Support Vector Learning*, B. Scholkopf et al., Eds. MIT Press, Cambridge, MA, 169–184.

KO, Y. AND SEO, J. 2000. Automatic text categorization by unsupervised learning. In *Proceedings of the 18th International Conference on Computational Linguistics*.

KO, Y. AND SEO, J. 2002. Text categorization using feature projections. In *Proceedings of the International Conference on Computational Linguistics*.

KO, Y. AND SEO, J. 2004. Learning with unlabeled data for text categorization using bootstrapping and feature projection techniques. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'04)*.

LANG, K. 1995. NewsWeeder: Learning to filter netnews. In *Proceedings of the12th International Conference on Machine Learning (ICML'95)*. 331–339.

LIU, B., LI, X., LEE, W. S., AND YU, P. S. 2004. Text classification by labeling words. In *Proceedings of the Conference on Natural Language Processing and Information Extraction*.

MAGNINI, B. AND CAVAGLIA, G. 2000. Integrating subject field codes into WordNet. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC'00)*. 1413–1418.

MAGNINI, B., STRAPPARAVA, C., PEZZULO, G., AND GLIOZZO, A. 2001. Using domain information for word sense disambiguation. In *Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL2)*. 111–114.

MAGNINI, B., STRAPPARAVA, C., PEZZULO, G., AND GLIOZZO, A. 2002. The role of domain information in word sense disambiguation. *Natural Lang. Engin. 8*, 4, 359–373.

MCCALLUM, A. AND NIGAM, K. 1999. Text classification by bootstrapping with keywords, EM and shrinkage. In *Proceedings of the Workshop for Unsupervised Learning in Natural Language Processing (ACL'99)*.

REDNER, R. AND WALKER, H. 1984. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review 26*, 2, 195–239.

RIFKIN, R. AND KLAUTAU, A. 2004. In defense of one-vs-all classification. *J. Machine Learn. Resear. 5*, 101–141.

SALTON, G. AND MCGILL, M. 1983. In *Introduction to Modern Information Retrieval*. McGraw-Hill, New York.

SEBASTIANI, F. 2002. Machine learning in automated text categorization. ACM *Comput. Surv. 34*, 1, 1–47.

SILVERMAN, B. W. 1986. In *Density Estimation for Statistics and Data Analysis*. Chapman and Hall.

VAPNIK, V. 1995. *The Nature of Statistical Learning Theory*. Springer, Berlin.

YAROWSKY, D. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. 189–196.

ZHANG, Y. AND CALLAN, J. 2001. Maximum likelihood estimation for filtering thresholds. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, ACM, New York.

Received July 2008; revised June 2009; accepted July 2009