Transductive Learning for Text Classification Using Explicit Knowledge Models

Georgiana Ifrim and Gerhard Weikum

Max-Planck Institute for Informatics,
Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
{ifrim, weikum}@mpi-inf.mpg.de

Abstract. We present a generative model based approach for transductive learning for text classification. Our approach combines three methodological ingredients: learning from background corpora, latent variable models for decomposing the topic-word space into topic-concept and concept-word spaces, and explicit knowledge models (light-weight ontologies, thesauri, e.g. WordNet) with named concepts for populating latent variables. The combination has synergies that can boost the combined performance. This paper presents the theoretical model and extensive experimental results on three data collections. Our experiments show improved classification results over state-of-the-art classification techniques such as the Spectral Graph Transducer and Transductive Support Vector Machines, particularly for the case of sparse training.

Keywords: transductive learning, latent models, expectation-maximization.

1 Introduction

1.1 Motivation

Many applications require classification models able to learn from few labeled data and rich background corpora. Examples could be Amazon organizing book descriptions into pre-defined categories, Google or Yahoo! classifying crawled Web pages into topic directories for more convenient access, or Wikipedia categorizing encyclopedia articles for better search and browsing. A learning paradigm in which the data collection to be automatically labeled is available beforehand is referred to as *transductive inference.*

Transductive learning is particularly attractive for text classification with very few explicitly labeled training documents, which happens whenever human assessment is the (time or cost) bottleneck on rapidly growing and highly diverse corpora. In such a setting, being able to harness the feature distributions and relations among the unlabeled documents is an important asset to improve the classifier. Another potentially beneficial asset is explicit knowledge about concepts, words and phrases that express concepts, and the semantic relations among concepts (e.g., hyponymy). Such knowledge sources may be given in the form of an ontology or thesaurus.

Prior work has mostly pursued "latent semantic" models such as spectral analysis, and the few approaches that have attempted to leverage explicit knowledge sources focused on concept-aware feature spaces and did not integrate concept-word relationships into the learning procedure itself. Moreover, many of these prior methods faced difficult model selection problems regarding feature engineering and parameter tuning. In the current paper, we develop a novel approach, based on a generative model with explicit concepts, that combines background ontologies with transductive learning on large corpora. Our approach aims to make classifiers more robust and improve classification accuracy with very few training data and as little model tuning as possible.

1.2 Contribution

Our approach is based on a generative model for text documents where words are generated by concepts which in turn are generated by topics. We postulate conditional independence between words and topics given the concepts. Once the corresponding probabilities for word-concept and concept-topic pairs are estimated, we can use Bayesian inference to compute the probability that a test document with known words but unobservable concepts belongs to a certain topic. The concepts are used as latent variables here, but unlike earlier work on spectral decomposition and latent semantic models [2], [6], [8] our concepts are named and can be explicitly identified in the underlying ontology or thesaurus. We employ an iterative EM (expectation-maximization) procedure for maximum-likelihood parameter estimation. The effectiveness of EM greatly benefits from a judicious initialization step that estimates word-concept probabilities on the *unlabeled* part of the document collection, thus leading us to a transductive learning method.

For illustration, consider a training document that contains the ambiguous word *Java* and belongs to the topic *geography.* Our method uses the background ontology and the word-occurrence contexts of the entire corpus, comprising both training and unlabeled documents, to map the word *Java* to its corresponding concept *Java, island belonging to Indonesia* and, at the same time, estimate the probability distribution of words that can express this concept (with, e.g., "Sumatra" or "Jakarta" having higher probabilities than "method" or "inheritance"). The statistics gained from the full corpus are fed into the EM procedure to further improve the concept-word estimates and to learn the concept-topic distribution. This technique populates only latent concept variables that have significant probability of being associated with the words occurring in documents, thus making the EM iterations much more efficient and effective. We consider the following as our main contributions:

1. By using explicit concepts from an ontology or thesaurus and by using a heuristic technique for bootstrapping the word-to-concept mapping, we avoid the model selection problem inevitably faced by all techniques based on latent dimensions (i.e., choosing an appropriate number of dimensions).

2. By the same token, we avoid the combinatorial explosion in the space of parameters to be estimated (i.e., concept-word pairs), and we largely eliminate the need for parameter smoothing (which is often a very tricky issue).
3. The initial word-to-concept mapping is very beneficial for fast convergence of EM, and reduces the danger of getting stuck in local maxima of the likelihood function. The latter is an issue especially with very few training data.
4. Our method provides an intuitive and effective way of exploiting the available resources, unlabeled documents from the full corpus and ontological relationships among concepts, resulting in improved classification accuracy with few training data.
5. Our approach is more robust in the sense that it requires considerably less tuning than other transductive methods.

In our experiments, with real-life datasets from the Reuters newswire corpus, Amazon book reviews, and Wikipedia articles, we compare our method with the Spectral Graph Transducer [11] and Transductive SVM classifiers [12] and demonstrate the viability and superiority of our approach.

1.3 Related Work

There are many approaches to transductive learning. Transductive SVMs were introduced by [20] and applied to text classification by [1], [12]. They exploit the structure in both training and test data for better positioning the maximum margin hyperplane. However, as shown empirically and theoretically in [21], learning a maximum margin from the unlabeled data in order to assign the labels is unreliable. Generative model based approaches can exploit the information in the unlabeled test collection for better estimating the generating distribution. An example is the usage of unlabeled data for re-weighting labeled examples [16]. Other methods represent the dataset as a graph and exploit the structure in the entire dataset in search for mincuts [4] or for min average cuts [11].

Various latent aspect models have been proposed in the literature [2], [6], [8]. In all this prior work, latent dimensions are mathematical constructions without any association to explicitly denoted concepts. This can make the interpretation of classification results by such methods difficult or unintuitive. Moreover, all prior methods require careful tuning of the number of latent dimensions. Finally, latent aspect models are either completely unsupervised or used as preprocessing step for feature engineering.

Explicit knowledge sources like ontologies, thesauri, or dictionaries have been used in prior work on text classification only for feature engineering, e.g., constructing composite words or phrases as features based on a thesaurus. Most notably, the WordNet thesaurus [7] has been leveraged in various approaches of this kind [3], [19]. In contrast, we propose integrating explicit knowledge about word-concept relationships into the learning procedure itself.

2 Transductive Latent Model

In this section we introduce our framework and the theoretical model proposed.

2.1 Generative Model

We are given a large *document collection*, $D = \{d_1, \ldots, d_r\}$, split into a small *training* set with known *topic labels*, $T = \{t_1, \ldots, t_m\}$, and a large test set with unknown topic labels. We have access to an *ontology graph of concepts*, $C = \{c_1, \ldots, c_k\}$, where each concept has a set of synonyms and a short textual description, and is related to other concepts by semantic edges (e.g., hypernymy/hyponymy relations). From the given collection we can select a set of *features*, $F = \{f_1, \ldots, f_n\}$ (*words or phrases*). A document is considered to be a multiset of features. Our generative model for feature-topic co-occurrence can be described as:

1. Select a topic t with probability $P[t]$;
2. Pick a latent variable c with probability $P[c|t]$, the probability that concept c describes topic t;
3. Generate a feature f with probability $P[f|c]$, the probability that feature f means concept c.

The pairs (f, t) can be directly observed, while the existence of concepts implies some form of word sense disambiguation; concepts are treated as latent variables.

Mathematically, our model is similar to the *aspect model* developed in [8]. However, while in the aspect model the number of concepts has to be known a priori and the concepts themselves are implicit (an abstract mathematical notion), our model uses existing knowledge resources (ontologies, thesauri) to identify and select explicit concepts at runtime. A similar model was used in our earlier work [9] for the inductive learning setting.

2.2 Learning Model Parameters

For tractability reasons, the generative model is based on two independence assumptions: The observation pairs (f, t) are generated independently and the features f are conditionally independent of the topics t, given the latent variable c: $P[(f, t)|c] = P[f|c] \cdot P[t|c]$. Using explicit concepts from a fine-grained ontology like WordNet helps alleviating the impact of these assumptions on the model robustness. To describe the generative process of an observation (f, t), we sum up over all the possible values that the latent variables might take

$$P[f, t] = \sum_{c \in C} P[c] \cdot P[(f, t)|c]. \tag{1}$$

We assume that the (f,t) pairs are generated from a multinomial distribution. Thus, the likelihood of the observed (f, t) samples can be expressed as:

$$\begin{aligned} L &= \Pi_{f \in F, t \in T} P[f, t]^{n(f,t)} \\ &= \Pi_{f \in F, t \in T} (\sum_{c \in C} P[c] \cdot P[(f, t)|c])^{n(f,t)} \\ &= \Pi_{f \in F, t \in T} (\sum_{c \in C} P[f|c] \cdot P[c|t] \cdot P[t])^{n(f,t)} \end{aligned} \tag{2}$$

where $n(f,t)$ is the number of occurrences of feature f in the training set of topic t (multiple occurrences in one document count multiple). The learning problem consists in estimating the parameters ($P[f|c]$, $P[c|t]$, $P[t]$) of the generating distribution, given the observed (f,t) samples. This can be formally expressed as a maximization of the observed data log-likelihood:

$$l = \sum_{f\in F, t\in T} n(f,t)\cdot log(\sum_{c\in C} P[f|c]\cdot P[c|t]\cdot P[t]) \tag{3}$$

Due to the sum inside the logarithm direct maximization of the log-likelihood by partial derivatives is difficult. To overcome this problem, we employ an Expectation-Maximization (EM) algorithm (see [10] for more details). The EM algorithm works by 2 iterative steps:

– **E-Step:** Expectation step, in which posterior probabilities are estimated for the latent variables, taking as evidence the observed data. For calculating the estimates of the E-step, we use Bayes' formula:

$$\begin{aligned} P[c|(f,t)] &= \frac{P[(f,t)|c]\cdot P[c]}{\sum_{c'\in C} P[(f,t)|c']\cdot P[c']} = \frac{P[f|c]\cdot P[c|t]\cdot P[t]}{\sum_{c'\in C} P[f|c']\cdot P[c'|t]\cdot P[t]} \\ &= \frac{P[f|c]\cdot P[c|t]}{\sum_{c'\in C} P[f|c']\cdot P[c'|t]} \end{aligned} \tag{4}$$

– **M-Step:** Maximization step, in which the current parameters are updated based on the expected complete data log-likelihood, which depends on the posterior probabilities estimated in the E-Step. Detailed justification for the equations (5), (6) and (7) is given in [10].

$$P[f|c] = \frac{\sum_{t\in T} n(f,t)P[c|(f,t)]}{\sum_{f'\in F}\sum_{t\in T} n(f',t)P[c|(f',t)]} \tag{5}$$

$$P[c|t] = \frac{\sum_{f\in F} n(f,t)P[c|(f,t)]}{\sum_{c'\in C}\sum_{f\in F} n(f,t)P[c'|(f,t)]} \tag{6}$$

$$P[t] = \frac{\sum_{f\in F, c\in C} n(f,t)P[c|(f,t)]}{\sum_{t'\in T}\sum_{f\in F, c\in C} n(f,t')P[c|(f,t')]} \tag{7}$$

In order to compute the conditional probability of a document given a topic $P[d|t]$, postulating feature independence, the estimates $P[f|c]$ and $P[c|t]$ are used:

$$P[d|t] = \Pi_{f\in d} P[f|t] = \Pi_{f\in d}\sum_{c\in C} P[f|c]\cdot P[c|t] \tag{8}$$

Once we have estimates for the marginal distribution describing the generative model, we can use Bayes' rule to reverse the model and predict which topic generated a certain document:

$$P[t|d] = \frac{P[d|t]\cdot P[t]}{P[d]} = \frac{P[d|t]\cdot P[t]}{\sum_{t'\in T} P[d|t']\cdot P[t']} \tag{9}$$

We then substitute (8) into (9) and have a decision procedure for the classifier.

2.3 Problems and Solutions

EM faces two major problems:

1. The combinatorial explosion of the variable space in the model, since the number of parameters is directly proportional to the cross-product of the number of features, concepts and topics. These parameters are sparsely represented in the observed training data.
2. The possibility of slow convergence or convergence to a local maximum of the likelihood function and missing the global optimum.

For the first problem, it is desirable to **prune the parameter space** to reflect only the meaningful latent variables. We propose two techniques to this end.

Feature Selection. From the given collection we extract a set of features using $tf \cdot idf$ ranking as a quality measure. For each term we compute its frequency in the entire collection (tf) and its inverse document frequency (idf). We normalize these quantities [5] and rank the terms according to their $tf \cdot idf$ value. As a preprocessing step, we extract compound words, e.g. "exchange market", using a sliding window parser and a background dictionary (e.g WordNet), and treat them as individual features. This step can play a role in capturing the semantics of interesting and common language constructions; it also reduces some of the computational overhead, while helping model robustness: many compounds have only one meaning, e.g. "exchange market", versus "exchange" and "market".

Concept Set Selection. We use the WordNet thesaurus [7] as the basis for our ontology graph. WordNet contains around 150,000 concepts (word senses) linked by hierarchical relations. Using the entire set of concepts would result in a high computational overhead and a high amount of noise. A better approach is to select from the ontology only a subset of concepts that reflects the semantics of the data collection. We call this the *candidate set of concepts*. This set is selected in a preprocessing step, before running the EM algorithm; details are given below. The size of this subset is only a few thousands concepts, as opposed to some hundred-thousands available in the ontology.

For the second problem, slow or suboptimal convergence, it is desirable to **pre-initialize the model parameters** to values that are close to the global maximum of the likelihood function. In order to get a good initialization for our parameters $P[f|c]$ and $P[c|t]$ we use a similarity-based mapping approach. The technique consists of mapping features to concepts and concepts to topics, based on *similarity measures* between contexts. The WordNet thesaurus can be seen as a directed acyclic graph (DAG) where the nodes are the different concepts and the edges are semantic relationships [7]. Let w be a word that we want to map to the ontological senses. First, we query WordNet for the possible meanings of word w. Let $\{c_1, \ldots, c_m\}$ be the set of meanings associated with w. By taking also the synonyms of these word senses, we can form *synsets* for each of the word meanings. Next, we apply a word sense disambiguation step by computing the overlap between the local contexts of both the word observed in a document and each of its possible meanings. This type of approach is commonly used in the word sense disambiguation literature. For each occurrence of word w in the text collection, its local context is a text window around its offset; the context for the concept is taken from the ontology: for each sense c_i we take its hypernyms, hyponyms, holonyms and their short textual descriptions. The context of a concept in the ontology graph can be taken until a certain depth, depending on the amount of noise one is willing to introduce in the disambiguation process. In this work we use depth 2. For each of the candidate senses c_i, we compute the cosine similarity between the tf vectors of $context(w)$ and $context(c_i),\ \ i \in \{1, \ldots, m\}$. Only the most dominant meaning (over all occurrences of w) is kept in the concept space. For words having multiple meanings, our hypothesis is that "secondary" meanings are introduced by their corresponding features, e.g., for the word *Java* having meanings *island* and *coffee*, if *island* is selected as the main meaning of *Java*, the second meaning *coffee* can still be introduced in the concept space by occurrences of words like *espresso, latte, coffee beans*, etc.

In a similar fashion, we relate concepts to topics based on similarity of contexts. The context for a topic t is defined to be the bag-of-features selected from the training collection by decreasing Mutual Information (MI) [14] value. For our implementation, we used the top 50 terms with regard to MI rank. Once we have computed all the similarities for the (feature, concept) and (concept, topic) pairs, we normalize them, and interpret them as estimates of the probabilities $P[f|c]$ and $P[c|t]$. In the $sim(f, c)$ and $sim(c, t)$ computations, we only consider the (f, c) and (c, t) pairs in the pruned parameter space. The computed values are then used for enhancing the EM algorithm in the model fitting process.

Transductive Learning. Our *feature-concept* mapping does not require labeled documents, thus it can be computed on the entire unlabeled collection. This has the effect of *drastically reducing the need for training data.* The *concept-topic* mapping might be poorly estimated from sparse training as the topics might be poorly described in the training set. This problem can be alleviated by extending the topic description using background corpora. For example, we can improve the training description of the topic *Biology* by using a better description from an encyclopedia, e.g. the Biology page from Wikipedia. We note that this methodology of using background knowledge makes our model robust to variations in vocabulary or data distribution, problems that can occur when directly adding background data to the training set.

2.4 Enhanced EM Algorithm

The mapping step presented in Section 2.3 can be seen as defining a prior probability distribution on the model parameters $P[f|c],\ P[c|t]$. This can be exploited in a Maximum A Posteriori (MAP) estimation of parameters, rather than simple maximum likelihood estimation. We denote by $\theta = (\theta_{f|c}, \theta_{c|t}, \theta_t)$, $\theta_{f|c} = P[f|c]$, $\theta_{c|t} = P[c|t]$, $\theta_t = P[t]$ our model parameters. Let $\theta_{f|c} \sim Dirichlet(\alpha_f^c)$ and $\theta_{c|t} \sim Dirichlet(\beta_c^t)$, where $\alpha_f^c = sim(f, c)$ for each $f \in F$ and $c \in C$ and

$\beta_c^t = sim(c,t)$ for each $c \in C$ and $t \in T$. Let θ_t be uniformly distributed, with density $g(\theta_t) = \frac{1}{|T|}$. The corresponding densities for $\theta_{f|c}$ and $\theta_{c|t}$ are:

$$g(\theta_{f|c}) \sim \Pi_{f\in F}\ {\theta_{f|c}}^{\alpha_f^c},\ \theta_{f|c} \geq 0,\ \alpha_f^c \geq 0,\ \sum_{f\in F}\theta_{f|c} = 1,\ \forall c \in C \tag{10}$$

$$g(\theta_{c|t}) \sim \Pi_{c\in C}\ {\theta_{c|t}}^{\beta_c^t},\ \theta_{c|t} \geq 0,\ \beta_c^t \geq 0,\ \sum_{c\in C}\theta_{c|t} = 1,\ \forall t \in T \tag{11}$$

Because $\theta_{f|c}$, $\theta_{c|t}$ and θ_t are independent random variables, their joint density can be written as: $g(\theta) = g(\theta_{f|c}, \theta_{c|t}, \theta_t) = g(\theta_{f|c}) \cdot g(\theta_{c|t}) \cdot g(\theta_t)$. Let $x = (f,t)$ be an observation from a multinomial distribution. Let $F(\theta|x) = g(\theta) \cdot L(x|\theta)$, where $g(\theta)$ defines a prior distribution on the parameters and $L(x|\theta)$ is the likelihood of the (f,t) samples (as in Section 2.2). We want to compute the MAP estimate

$$\theta = argmax_\theta\ F(\theta|x) \tag{12}$$

As $g(\theta_t)$ is a constant function, it does not influence the maximization, and we leave it out in the following estimations. Let

$$F(\theta|x) = (\Pi_{c,f}\ {\theta_{f|c}}^{\alpha_f^c}) \cdot (\Pi_{t,c}\ {\theta_{c|t}}^{\beta_c^t}) \cdot [\Pi_{f,t}\ (\sum_c \theta_{f|c} \cdot \theta_{c|t} \cdot \theta_t)^{n(f,t)}] \tag{13}$$

For maximizing the above function we employ an EM algorithm (similar to the estimation in Section 2.2). The **E-Step** remains the same. The parameter estimates $\theta_{f|c}$ and $\theta_{c|t}$ for the **M-Step** become:

$$\theta_{f|c} = P[f|c] = \frac{\alpha_f^c + \sum_{t\in T} n(f,t)P[c|(f,t)]}{\sum_{f'\in F}(\alpha_{f'}^c + \sum_{t\in T} n(f',t)P[c|(f',t)])} \tag{14}$$

$$\theta_{c|t} = P[c|t] = \frac{\beta_c^t + \sum_{f\in F} n(f,t)P[c|(f,t)]}{\sum_{c'\in C}(\beta_{c'}^t + \sum_{f\in F} n(f,t)P[c'|(f,t)])} \tag{15}$$

Combining the similarity-based estimates with the estimates based on training counts strengthens the model robustness. We show in the next section how the additional flexibility of our model helps learning even when, due to little training, other methods fail.

3 Experiments

3.1 Methodology

The experimental setup is the following: given a large unlabeled collection of documents, the goal is to categorize it into predefined categories. For this purpose, we want to find out how much labeled data is needed for obtaining a reasonable classification accuracy, and what classifier methods perform best. As we most likely have a small amount of labeled (training) data and a large amount of unlabeled (test) data, the most interesting methods are those that learn over the entire dataset, i.e. transductive techniques. For comparison, we show experiments with both inductive (i.e. methods that learn only from training) and transductive methods. Our results are averaged over 10 repetitions with random splits into training and test data, with the size of the splits ranging from 0.25% training data and 99.75% test data, up to 10% – 90% training-test splits.

3.2 Test Collections

We evaluate our techniques on three test collections. For all three collections, both stemming and stop-word removal are used.

The first one is the **Reuters-21578 dataset** collected from the Reuters newswire in 1987. Of the 135 potential categories only the most frequent 10 are used (so that enough training/test documents are present) and we keep only documents labeled with a unique topic. This results in a total of 8,024 documents.

The **Amazon** dataset[1] was extracted from `http://www.amazon.com`. It contains editorial reviews of books, organized into categories. From the available categorization, we selected all the editorial reviews for books in *Biological Sciences, Mathematics, and Physics.* This amounts to 5,634 documents.

The **Wikipedia** collection[2] was obtained by a topic-focused crawl of `http://www.wikipedia.org`. The selected topics were *Politics, Computer Science, Physics, Chemistry, Biology, Mathematics, and Geography.* The crawl was started from the main pages of each of these classes and topic-specific words in the anchor text were used as indicators for whether an outgoing link should be followed. The dataset gathered this way contains 5,384 documents. Due to the crawling procedure, this dataset contains a considerable amount of noise. The last two collections differ in nature from the Reuters dataset: their vocabulary is much richer and expressive, the language ambiguity is higher.

3.3 Results

The following experiments show the effect of using explicit knowledge models for transductive inference. As a baseline for comparison the results of a multinomial Naive Bayes (NB) [14], the inductive version of our model coined ILM (Inductive Latent Model) using the MAP estimator of Section 2.4, inductive SVM (ISVM) [13], transductive SVM (TSVM) [12] and the Spectral Graph Transducer (SGT) [11] are shown. Our transductive model is coined TLM (Transductive Latent Model). Based on previous studies [18] about robust evaluation measures for text classification we chose microaveraged F1 as our main performance metric.

In settings with few training data, parameter tuning is infeasible: it is difficult to perform cross-validation or to provide held-out data. For this reason, for all the methods presented, the parameters are kept fixed across experiments. For

[1] Available at `http://www.mpi-sb.mpg.de/~ifrim/data/Amazon.zip`

[2] Available at `http://www.mpi-sb.mpg.de/~ifrim/data/Wikipedia.zip`

Table 1. Reuters. Microaveraged F_1 for different training set sizes.

Training	NB	ILM	ISVM	TSVM	SGT	**TLM**
split0.4	64.5	67.1	71.8	57.6	76.5	**79.7**
split0.5	67.2	70.0	73.5	50.2	79.7	**80.7**
split1	77.2	76.5	81.0	60.8	88.6	**84.1**
split2	83.4	81.7	82.2	72.7	89.9	**85.1**
split5	91.1	90.0	84.9	87.1	92.1	**89.4**
split10	92.9	92.0	88.4	89.4	93.0	**89.6**

Table 2. Wikipedia. Microaveraged F_1 for different training set sizes.

Training	NB	ILM	ISVM	TSVM	SGT	**TLM**
split0.25	72.7	70.4	43.0	32.5	70.5	**79.2**
split0.5	76.0	74.8	61.3	52.0	77.1	**80.5**
split1	77.9	76.9	73.2	63.1	79.3	**80.9**
split2	80.8	80.7	78.9	69.3	81.4	**80.8**
split5	83.6	82.1	83.2	77.8	82.9	**82.8**
split10	85.8	84.5	85.0	81.6	84.3	**84.8**

Table 3. Amazon. Microaveraged F_1 for different training set sizes.

Training	NB	ILM	ISVM	TSVM	SGT	**TLM**
split0.25	77.2	72.7	48.7	64.1	77.8	**83.7**
split0.5	79.1	74.8	63.0	68.0	82.4	**84.9**
split1	81.1	77.9	72.2	75.6	85.0	**85.1**
split2	83.3	81.4	78.2	82.1	86.0	**85.2**
split5	84.2	83.2	83.8	85.8	87.5	**85.5**
split10	85.2	84.9	84.5	86.7	88.1	**85.9**

NB, ILM and TLM, the vocabulary size is fixed to 10,000 terms selected by Mutual Information for NB and ILM and $tf \cdot idf$ rank for TLM. Due to our bootstrapping mapping, our *enhanced* EM algorithm converges very fast, in one or two iterations; for TLM we set the number of EM iterations to 1. As suggested by [12] we do not use any feature selection for SVMs. We use SVMLight [13] with linear kernels and default parameter settings. The parameters for SGT are fixed to the values found in [11] to give the best results on the Reuters dataset: $c = 3200$, $d = 80$ and $k = 800$; no feature selection is done.

As mentioned in Section 2.3, the concept-topic mapping might be poorly estimated from sparse training. We propose solving this problem by automatically querying Wikipedia as background knowledge for improving the bag-of-words context of a given topic, prior to computing the initial topic-concept mapping. For a fair comparison, we give all the other methods the same information as used by TLM. Directly adding the background knowledge to training (e.g. we add the *Coffee* page from Wikipedia as training for the *Coffee* topic from Reuters) sometimes decreases the accuracy of the other methods (due to variation in vocabulary and data distribution), and sometimes increases the accuracy, particularly for sparse training. We report the best results for all the other methods, so as to give them the best possible advantage against our TLM method.

Table 1 shows results for the Reuters dataset. Using only 33 labeled documents (0.4% of the total dataset), TLM improves the microaveraged F1 from 71.8% for ISVM and 76.5% for SGT, to 79.7%. In our experiments with this dataset, transductive SVM performed considerably worse than inductive SVM for small training sets. This result is different from the one in [12]. We suspect TSVM is particularly sensitive to parameter tuning, thus using default parameters could affect the results. We also note that even if SGT uses its best parameter setting on Reuters, TLM outperforms it for small training data.

Table 2 gives results for the Wikipedia dataset. For the interesting case of little training, TLM outperforms all other methods by a significant margin. The results are particularly impressive for very small training sets, when our model proves to be robust in spite of the little supervised information available. In most of our experiments NB is better than SVM for small training sets; this result is in compliance with existing text classification literature [15], [17].

Table 3 shows results for the Amazon dataset. The results show the same trend: for little training TLM outperforms all other methods. For small training sets (split0.25: 14 documents) TLM outperforms SGT by 6%; for training data large enough (split2: 113 documents) SGT becomes slightly better.

In order to understand the mutual benefits of the bootstrapping mapping and the EM algorithm, we studied the effect of the similarity-based mapping on the classification results, as compared to the results after applying one EM iteration (using maximum likelihood estimates, not MAP). On Reuters, the mapping results were considerably improved by using EM iterations. This can be explained by the nature of this dataset: topics overlap heavily at concept level, thus the mapping alone cannot discriminate well among topics. For both Amazon and Wikipedia the situation differed. For sparse training, employing EM iterations degraded a bit the results obtained with the bootstrapping mapping. This problem of bootstrapping mapping versus EM iterations' quality is largely solved by combining the two estimates in a MAP estimate as shown in Section 2.4.

Running time. The most time consuming part for our method is the mapping of features to concepts which takes about 20 minutes for 10,000 features. Due to our bootstrapping mapping, EM converges very fast, in one or two iterations. Classifying the entire collection is done in a few seconds.

4 Conclusion

This paper has introduced a generative model and a parameter learning algorithm for transductive text classification with explicit background knowledge. In contrast to previously proposed latent semantic models, our approach has explicit concepts associated with its latent variables, which enables it to bootstrap the EM-based parameter estimation procedure in a highly efficient and effective manner and provides the flexibility for learning from the unlabeled part of the corpus or even additional background corpora. Our extensive experiments have shown significant improvements of classification accuracy, in comparison to state-of-the-art techniques like Spectral Graph Transducer and Transductive as well as Inductive Support Vector Machines. The gains are most pronounced for small training sets, a typical and inevitable situation in many real-life text mining applications.

Acknowledgments. We would like to thank Thomas Hofmann, Deepak Ajwani and the anonymous reviewers for their helpful suggestions.

References

1. Bennet, K.: Combining support vector and mathematical programming methods for classification. Advances in Kernel Methods. MIT-Press (1999)
2. Blei, D., Ng, A., Jordan, M.: Latent dirichlet allocation. NIPS (2002)
3. Bloehdorn, S., Hotho, A.: Text classification by boosting weak learners based on terms and concepts. ICDM (2004)
4. Blum, A., Chawla, S.: Learning from labeled and unlabeled data using graph mincuts. ICML (2001)
5. Chakrabarti, S.: Mining the Web: Discovering Knowledge from Hypertext Data. Morgan Kaufman Publishers (2003).
6. Deerwester, S., Dumais, S.T., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society of Information Science 41(6) (1990)
7. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press (1999)
8. Hofmann, T.: Unsupervised learning by probabilistic latent semantic analysis. Machine Learning 42(1) (2001)
9. Ifrim, G., Theobald, M., Weikum, G.: Learning word-to-concept mappings for automatic text classification. Learning in Web Search Workshop, ICML (2005)
10. Ifrim, G.: A Bayesian Learning Approach to Concept-Based Document Classification. Master Thesis. http://www.mpi−inf.de/∼ifrim/publications/ (2005)
11. Joachims, T.: Transductive learning via spectral graph partitioning. ICML (2003)
12. Joachims, T.: Transductive inference for text classification using Support Vector Machines. ICML (1999)
13. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. ECML (1998)
14. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. AAAI-98 Workshop on "Learning for Text Categorization (1998)
15. Ng, A., Jordan, M.: On discriminative versus generative classifiers: A comparison of logistic regression and naive bayes. NIPS (2001)
16. Nigam, K., McCallum, A., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. Machine Learning (39) (2000)
17. Rennie, J.: Tackling the poor assumptions of naive bayes. ICML (2003)
18. Sebastiani, F.: Machine learning in automated text categorization. ACM (2002)
19. Scott, S., Matwin, S.: Feature engineering for text classification. ICML (1999)
20. Vapnik, V.: Statistical learning theory. Wiley (1998)
21. Zhang, T., Oles, F.J.: A probability analysis on the value of unlabeled data for classification problems. ICML (2000)