# Learning Word-to-Concept Mappings for Automatic Text Classification

**Georgiana Ifrim** IFRIM@MPI-INF.MPG.DE
**Martin Theobald** MARTIN.THEOBALD@MPI-INF.MPG.DE
**Gerhard Weikum** WEIKUM@MPI-INF.MPG.DE
Max-Planck Institute for Informatics, D-66041 Saarbruecken, Germany

## Abstract

For both classification and retrieval of natural language text documents, the standard document representation is a term vector where a term is simply a morphological normal form of the corresponding word. A potentially better approach would be to map every word onto a concept, the proper word sense and use this additional information in the learning process. In this paper we address the problem of automatically classifying natural language text documents. We investigate the effect of word to concept mappings and word sense disambiguation techniques on improving classification accuracy. We use the WordNet thesaurus as a background knowledge base and propose a generative language model approach to document classification. We show experimental results comparing the performance of our model with Naive Bayes and SVM classifiers.

## 1. Introduction

### 1.1. Motivation

Text classification, e.g., for categorizing Web documents into topics like sports, science, math, etc., is usually based on supervised learning techniques such as support vector machines (SVM) with feature vectors as representatives of both the training and test documents. The features are usually derived from the bag-of-words model, where individual words or word stems constitute features and various frequency measures are used to compute weights, e.g., using the $tf \cdot idf$ approach or statistical language models (Manning & Schütze, 2000; Croft & Lafferty, 2003). Classification accuracy is limited by three potential bottlenecks: 1) the quality of the training data, 2) the discriminative power of the classifier and 3) the richness of the features to represent documents. The first point is usually an application issue and beyond control of the classifier, and with the great advances in statistical learning, the second point is widely perceived as the least limiting factor. In this paper, we address the third point.

Despite the sophisticated statistical models for computing feature weights, using words or word stems as features is a semantically poor representation of the text content. Richer features could be derived from syntactic analysis of the text (using part-of-speech tagging, chunk parsing, etc. (Manning & Schütze, 2000)), and most importantly, using concepts rather than words, thus capturing the intended word sense instead of the literal expressions in the document. As an example, consider the word "Java", a classical polysem (i.e., a word with multiple word senses). For classifying a document into topic "travel" or "computer science", the word itself is not helpful. But if we could map it to its proper meaning, within the context of the document, then we would be able to boost the classifier: if "Java" is used as the concept "island (part of Indonesia)" it should raise the probability of category "travel", whereas the use as the concept "object-oriented programming language" would give higher evidence to the category "computer science".

For mapping words onto concepts, we build on the availability of rich knowledge sources like lexicons, thesauri, and ontologies. For the scope of this paper, we specifically use the WordNet thesaurus (Fellbaum, 1999), which contains around 150,000 concepts (word senses in WordNet's terminology), each with a short textual description, and semantic relationships between concepts - hypernym/hyponym (IS A), holonym/meronym (PART OF). We use a machine learning approach, based on latent variables and EM

Appearing in *W4: Learning in Web Search, at the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005. 

iteration for parameter estimation, to compute the actual word-to-concept mappings. It is important to note that WordNet, albeit probably the most prominent source of this kind, is just an example of the explicit concept collections that could be leveraged for better text representation and classification accuracy. Ontologies are being built up (Staab & Studer, 2004), and it is conceivable that concepts can be mined from encyclopedia like Wikipedia.

### 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 previously unseen test document with known words but unobservable concepts belongs to a certain topic. The concepts are used as latent variables here, but note that unlike earlier work on spectral decomposition (Deerwester & Dumais & Harshman, 1990; Hofmann, 2001) for text retrieval our concepts are named and can be explicitly identified in the underlying thesaurus or ontology.

The learning procedure for estimating the probabilities that involve latent variables is a maximum-likelihood estimator based on the observed word-topic pairs in the training data. We use an EM (expectation-maximization) procedure for iteratively solving the analytically intractable estimation problem. The number of concepts that we consider in this approach is naturally limited and determined by an initialization step that uses a text-context similarity comparison for an initial, heuristic mapping of words onto concepts. Note, however, that the final result of the word-to-concept mapping is usually much better than the outcome of the initial heuristics. Our overall approach can also be seen as a learning-based method for word sense disambiguation coupled with a classifier for topic labeling.

Different flavors of latent variable models for text data exist in the literature (Cai & Hofmann, 2003; Bhattacharya & Getoor & Bengio, 2004). Previous work employed Wordnet for feature engineering (Scott & Matwin, 1999; Bloehdorn & Hotho, 2004), but our model has the following major advantages, which we claim as our main contributions:

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

2. By the same token, we avoid the potential combinatorial explosion in the space of parameters to be estimated, and we can do away with the need for parameter smoothing (often a very tricky and treacherous issue).

3. The initial mapping provides us with a good initialization of the EM iteration, positively affecting its convergence and reducing the (empirical) risk that it gets stuck in a local maximum of the likelihood function.

In our experiments, with real-life datasets from the Reuters newswire corpus and editorial reviews of books from the Amazon web site, we compare our approach with a Naive Bayes classifier and an SVM classifier (Hastie & Tibshirani & Friedman, 2003; McCallum & Nigam, 1998; Joachims, 1998). The results show that our method can provide substantial gains in classification accuracy for rich text data where the expressiveness and potential ambiguity of natural language becomes a bottleneck for traditional bag-of-words classifiers.

The rest of the paper is organized as follows. Section 2 describes our probabilistic generative model. Section 3 presents our techniques for efficiently estimating the model parameters. Section 4 discusses experimental results.

## 2. Probabilistic Model

### 2.1. Generative Model

In this section we introduce our framework and the theoretical model proposed. The general setup is the following:

- A *document collection*, $D = \{d_1, \ldots, d_r\}$, with known *topic labels*, $T = \{t_1, \ldots, t_m\}$, which is split into training and test data. In this work we assume a one-to-one mapping between documents and topic labels.

- A set of *lexical features*, $F = \{f_1, \ldots, f_n\}$, that can be observed in documents (*individual or composite words*).

- An *ontology DAG 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.

The goal is solving a document classification problem: for a given document $d$ with observed features, we would like to predict $P[t|d]$ for every topic $t$ or find $argmax_t P[t|d]$. To get an intuition behind our model, consider analyzing documents labeled with a certain topic label, e.g. *physics*. Conceptually, this broad concept (the topic label) can be described at semantic level by a subset of more fine grained concepts that describe for example phenomena or structures related to physics, e.g *atom, molecule, particle, corpuscle, physical science*, etc. In turn, these concepts are expressed at the lexical level, by means of simple terms or compounds: *physical science*, *material*. Thus, we want to explain feature-topic associations by means of latent concepts. Figure 1 shows a graphical representation of our generative model. The model proposed by us is similar to the *aspect model* developed in (Hofmann, 2001). It is a latent variable model for co-occurrence data which associates an unobserved variable $c \in \{c_1 \dots c_k\}$ with each observation.

Our model differs from the aspect model in the following respects. In the aspect model, the number of concepts is fixed beforehand, but the concepts themselves are derived in an unsupervised way from the data collection, *without recourse to a lexicon or thesaurus*; an observation is the occurrence of a word in a particular document; parameters are randomly initialized. Our model uses the existing knowledge resources to identify and select the latent concepts at runtime; an observation is a pair $(f, t)$, where $f \in F$ is a feature observed in some document and $t \in T$ stands for a topic label; parameters are pre-initialized to help model robustness. 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 process of word sense disambiguation and they are treated as latent variables. The model is based on two independence assumptions: observation pairs $(f, t)$ are assumed to be generated independently and it is assumed that features $f$ are conditionally independent of the topics $t$, given the latent variable $c$: $P[(f, t)|c] = P[f|c] \cdot P[t|c]$. To describe the generative process of an observation $(f, t)$ we sum up over all the possible values that the latent variables might take

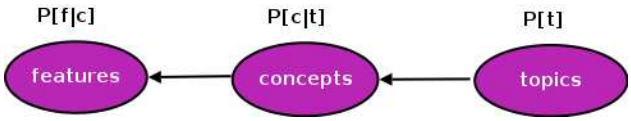


*Figure 1.* Graphical model representation of the generative model

$$P[f, t] = \sum_c P[c] \cdot P[(f, t)|c]. \quad (1)$$

The likelihood of the observed pairs $(f, t)$ can be expressed as:

$$L = \Pi_{f,t} P[f, t]^{n(f,t)} \quad (2)$$

where $n(f, t)$ is the number of occurrences of feature $f$ in the training set of topic $t$.
The learning problem can be expressed now as a maximization of the observed data log-likelihood:

$$\begin{aligned} l &= \sum_{(f,t)} n(f, t) \cdot log(P[f, t]) \quad (3) \\ &= \sum_{(f,t)} n(f, t) \cdot log(\sum_c P[c] \cdot P[(f, t)|c]) \end{aligned}$$

Due to the existence of the sum inside the logarithm direct maximization of the log-likelihood by partial derivatives is difficult. A solution in setups in which maximization of the likelihood is difficult, but made easier by enlarging the sample with latent data, is to apply an Expectation-Maximization (EM) algorithm. 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 (current estimates of the model parameters). For calculating the probabilities of the E-step, we use Bayes' formula:

$$P[c|(f, t)] = \frac{P[f|c] \cdot P[c|t]}{\sum_c P[f|c] \cdot P[c|t]} \quad (4)$$

- **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.

$$P[f|c] = \frac{\sum_t n(f, t) P[c|(f, t)]}{\sum_f \sum_t n(f, t) P[c|(f, t)]} \quad (5)$$

$$P[c|t] = \frac{\sum_f n(f, t) P[c|(f, t)]}{\sum_c \sum_f n(f, t) P[c|(f, t)]} \quad (6)$$

$$P[t] = \frac{\sum_{f,c} n(f,t)P[c|(f,t)]}{\sum_t \sum_{f,c} n(f,t)P[c|(f,t)]} \quad (7)$$

In our implementation the E-step and the M-step are iterated until convergence of the likelihood. Alternatively, one can also use the technique of *early stopping* - stop the algorithm when the performance on some held-out data starts decreasing, in order to avoid overfitting the model.

Now, we estimate the distribution of a document $d$, given a topic label $t$, by making use of the learned features' marginal distributions during the training process:

$$\begin{aligned} P[d|t] &= \Pi_{f\in d} P[f|t] = \Pi_{f\in d} \frac{P[f,t]}{P[t]} \quad (8) \\ &= \Pi_{f\in d} \sum_{c\in C} P[f|c] \cdot P[c|t] \end{aligned}$$

where $P[f|c]$ and $P[c|t]$ are estimated by the EM procedure so as to maximize $P[f,t]$ and implicitly $P[d|t]$.

### 2.2. Naive Bayes Classifier

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:

$$P[t|d] = \frac{P[d|t] \cdot P[t]}{P[d]} = \frac{P[d|t] \cdot P[t]}{\sum_t P[d|t] \cdot P[t]} \quad (9)$$

We can then substitute (8) into (9) and have a decision procedure for the classifier. The hope is that by the means of the latent variable model the distribution that generated the given document will be estimated in a more accurate way.

## 3. Model Parameter Estimation

EM can face two major problems:

- 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.
- The possibility of converging to a local maximum of the likelihood function (i.e. not finding the global maximum).

For the first problem, it is desirable to prune the parameter space to reflect only the meaningful latent variables. For the second problem, it is desirable to preinitialize the model parameters to values that are close to the global maximum of the likelihood function.

### 3.1. Pruning the Parameter Space

#### 3.1.1. Feature Selection

The feature selection process is done by retaining the features that have the highest average Mutual Information with the topic variable (McCallum & Nigam, 1998). For *multinomial models* the quantity is computed by calculating the mutual information between the topic of the document from which a word occurrence is drawn, and a random variable over all word occurrences.

$$f_k = \begin{cases} 1 & \text{if } w_k \text{ is present,} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

$$\begin{aligned} MI(T;W_k) &= H(T) - H(T|W_k) \quad (11) \\ &= \sum_{t\in T} \sum_{f_k\in\{0,1\}} P(t,f_k) \cdot log\left(\frac{P(t,f_k)}{P(t)\cdot P(f_k)}\right) \end{aligned}$$

As a preprocessing step before applying feature selection, we extract semantically significant compounds using a background dictionary (WordNet) e.g. *exchange market*, *linear algebra*, etc. This is a further step in capturing the semantics of interesting and common language constructions; it also reduces some of the computational overhead, while also achieving an increase in accuracy: many compound terms have only one meaning, e.g. *exchange market*, as a compound has fewer meanings than if analyzed separately *exchange* and *market*. After this stage, we can apply the MI selection criterion. Sorting the features in descending order of this measure gives us a ranking in terms of discriminative power of the features.

#### 3.1.2. Concept Set Selection

WordNet contains around 150,000 concepts linked by hierarchical relations. Using the full set of concepts provided by the ontology results in a high computational overhead combined with a high amount of noise. A better approach is to select from the ontology only a subset of concepts that reflects well the semantics of the training collection. In our work, we call this the *candidate set of concepts*. The set is selected in a preprocessing step, before running the EM algorithm. One way of capturing the candidate set well is to gather for each feature all the corresponding concepts (senses) from the ontology. The size order of this subset is only of a few thousands concepts, as opposed to some hundred-thousands available in the

ontology. Another way of even further improving the performance of this approach, is using PoS annotated data. We considered both approaches in our implementation.

### 3.2. Pre-initializing the Model Parameters

The standard way of using the EM algorithm is to randomly initialize the model parameters and iterate the algorithm until convergence. Since EM tends to stop in a local maximum of the likelihood function, the algorithm is restarted several times, and the values of the parameters that give the highest value of the likelihood are retained. However, this solution still does not guarantee that EM will stop at a global maximum. Our pre-initialization proposal combines the learning approach with a simpler approach of mapping features to concepts and concepts to topics, based on *similarity measures*.

For the initial mapping of words onto concepts in a thesaurus (ontology) we follow the approach in (Theobald & Schenkel & Weikum, 2003). The WordNet thesaurus can be seen as a DAG where the nodes are the different meanings and the edges are semantic relationships (Fellbaum, 1999). The vertices can be nouns, adverbs, verbs or adjectives.

Let $w$ be a word that we want to map to the ontological senses. First, we query WordNet for the possible meanings of word $w$; for improving precision we can use PoS annotations (i.e., noun vs. verb vs. adjective). Let $\{c_1, \ldots, c_m\}$ be the set of meanings associated with $w$. For example, if we query WordNet for the word *mouse* we get:

- *The* **noun** *mouse has 2 senses in WordNet.*
  1. *mouse – (any of numerous small rodents...)*
  2. *mouse, computer mouse – (a hand-operated electronic device...)*
- *The* **verb** *mouse has 2 senses in WordNet.*
  1. *sneak, mouse, creep, steal, pussyfoot – (to go stealthily or furtively)*
  2. *mouse – (manipulate the mouse of a computer)*

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.

The disambiguation technique proposed uses word statistics for a local context around both the word observed in a document and each of the possible meanings it may take. The context for the word is a window around its offset in the text document; the context for the concept is taken from the ontology: for each sense $c_i$ we take its synonyms, hypernyms, hyponyms, holonyms, and siblings 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 compare the context around the word $context(w)$ with $context(c_i)$ in terms of bag-of-words similarity measures. We use the cosine similarity measure between the $tf \cdot idf$ vectors of $context(w)$ and $context(c_i)$, $\quad i \in \{1, \ldots, m\}$. This process can either be seen as a proper word sense disambiguation step, if we take as corresponding word sense the one with the highest context similarity to the word's context, or as a step of establishing how words and concepts are related together and in what degree.

In a similar fashion, we relate concepts to topics based on similarity of bags-of-words. The context for a topic $t$ is defined to be the bag-of-features selected from the training collection by decreasing Mutual Information value. For our implementation, we used the top 50 (compound) terms with regards to MI rank. Once we have computed all the similarities for (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 initializing EM, as a preprocessing stage, in the model fitting process.

## 4. Preliminary Experiments

### 4.1. Setup

We present some preliminary experiments on two data collections. We analyze and compare four classification methods: *LatentM* - a first version of the latent generative model proposed by us that does not exploit PoS tags; *LatentMPoS* - our generative model, enhanced with methods for exploiting PoS information; *NBayes* - a terms-only Naive Bayes classifier; *SVM* - a multiclass SVM classifier. For the SVM classifier, we have used the SVM Light and SVM Struct tools, developed for multiclass classification (Tsochantaridis & Hoffman & Joachims & Altun, 2004). To measure classification quality, we use microaveraged $F_1$-measure (Manning & Schütze, 2000). Training and test were performed on disjoint document sets.

### 4.2. Reuters-21578

The Reuters-21578 dataset is a news collection compiled from the Reuters newswire in 1987. We used the

"ModApte" split, that led to a corpus of 9,603 training documents and 3,299 test documents. We parsed the document collection and retained only documents belonging to one topic. Out of these, we selected the top five categories in terms of number of training documents available: *earn, acq, crude, trade, money-fx*. This split the collection into approximately 5,000 files for training and 2,000 files for testing. The classification task is to assign articles to their corresponding topics. For many categories, there is a direct correspondence between words and category labels e.g., the appearance of the term *acquisition* is a very good predictor of the *acq* category. The vocabulary is fairly small and uniform, each topic is described with standard terms, e.g. *crude oil, opec, barrel* are very frequent terms in the topic *crude*, so by using frequency of terms only we can get a high classification accuracy. We tested the sensitivity to the training set size for all the four methods. We averaged the performance over 3 randomly selected training sets of sizes: 10 to 200 documents per topic. The number of features is set to 300 based on studies concerning the appropriate vocabulary size for Reuters (McCallum & Nigam, 1998), which indicate this number of features is enough for obtaining a high classification accuracy. Particularly for topic *trade*, a high amount of noise is introduced by enlarging the feature space. Table 1 shows statistics regarding the number of concepts in our model, for different training set sizes. For the method using part of speech annotations, we use nouns and verbs. Table 2 shows microaveraged F1 results for the 5 chosen topics. We can observe that on the Reuters collection, exploiting the semantics of natural language does not outperform the methods that use simple term frequencies. We explain this effect by the nature of the vocabulary used in this collection in which term frequencies capture the nature of the training data in each topic well enough. Further studies are necessary in order to fully understand the behavior of the techniques proposed on this data collection.

*Table 1.* Number of concepts extracted for various training set sizes on Reuters-21578.

| Training per topic | Concepts LatentM | Concepts LatentMPoS |
|---|---|---|
| 10 | 2669 | 1560 |
| 20 | 2426 | 1395 |
| 30 | 2412 | 1321 |
| 40 | 2364 | 1447 |
| 50 | 2411 | 1317 |
| 100 | 2475 | 1372 |
| 150 | 2477 | 1385 |
| 200 | 2480 | 1387 |

*Table 2.* Microaveraged F1 results on Reuters-21578.

| Training per topic | NBayes | LatentM | LatentM PoS | SVM |
|---|---|---|---|---|
| 10 | 88.9% | 88.7% | 87.8% | 90.0% |
| 20 | 89.6% | 92.2% | 90.7% | 92.1% |
| 30 | 92.7% | 94.0% | 92.2% | 93.6% |
| 40 | 92.1% | 93.0% | 91.2% | 94.5% |
| 50 | 93.8% | 95.0% | 93.8% | 93.8% |
| 100 | 95.3% | 95.0% | 93.8% | 95.5% |
| 150 | 96.0% | 95.0% | 94.4% | 95.4% |
| 200 | 95.9% | 95.8% | 94.5% | 95.9% |

### 4.3. Amazon

In order to further test our methods, we extracted a real-world collection of natural language text from `http://www.amazon.com`. This site promotes books, which are grouped according to a representative category. From the available taxonomy, we selected all the editorial reviews for books in: *Biological Sciences, Mathematics, Physics.* Total number of documents extracted was 6,000 (48MB). We split this set into training (largest 500 documents per topic) and test (remaining documents after training selection). After this process we obtained 1,500 training documents and 4,500 test documents. The dataset is available at `http://www.mpi-sb.mpg.de/~ifrim/data/Amazon.zip`. Table 3 shows the distribution of documents over topics. For the method using PoS annotations, we use nouns, adjectives and verbs. For each of the methods analyzed, we tested the sensitivity to vocabulary size. Table 4 presents statistics regarding the concepts involved in the latent models for different dimensions of the feature space. Figure 2 shows microaveraged F1 results. We can observe a significant improvement in terms of performance achieved at different dimensionalities of the feature space. The *PoS* label attached to each method's name stands for the

*Table 3.* Training/test documents on Amazon.

| Category Name | Train size | Test size |
|---|---|---|
| Mathematics | 500 | 2,237 |
| Biological Sciences | 500 | 1,476 |
| Physics | 500 | 787 |

*Table 4.* Number of concepts extracted for various feature set sizes on Amazon.

| Number of features | Concepts LatentM | Concepts LatentMPoS |
|---|---|---|
| 100 | 1099 | 509 |
| 200 | 1957 | 936 |
| 300 | 2886 | 1390 |
| 400 | 3677 | 1922 |
| 500 | 4623 | 2232 |
| 600 | 5354 | 2547 |
| 700 | 5973 | 2867 |
| 800 | 6551 | 3231 |
| 900 | 7230 | 3677 |
| 1,000 | 7877 | 3959 |

usage of $PoS$ annotated features for the respective method. The only difference between $LatentMPos$ and $NBayesPoS$ or $SVMPoS$ is the mapping of features onto the concept space.

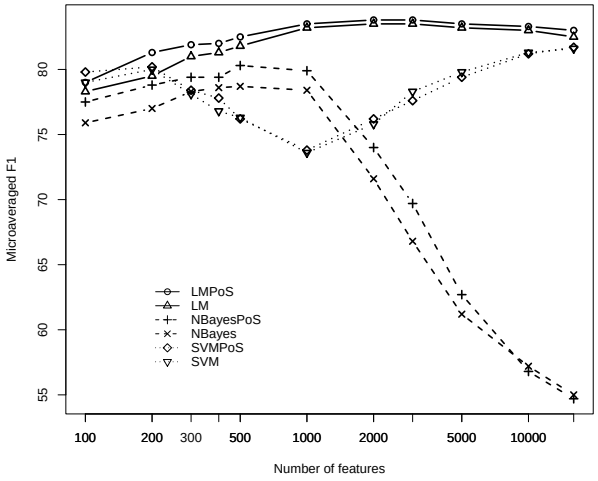


*Figure 2.* Microaveraged $F_1$ at different number of features.

Since this collection has a richer vocabulary, synonymy and polysemy effects can have more impact. We observe that exploiting semantics can have the potential of boosting classification performance. In Table 5 and 6, we show the exact values for microaveraged $F_1$ at higher dimensionalities of the feature space. We observe that $SVM$ performance using all the distinct terms in the collection (16,000) is inferior to our model with 1,000 features. Feature selection by MI does not eliminate correlations among features. This can have an effect on SVM performance for small dimensionalities of the feature space. We trained SVM using the default settings of SVM Struct: linear kernel and $C = 0.01$. In the future we plan a systematic study regarding SVM parameters tuning.

In Figure 3 and Table 7 we show the sensitivity of microaveraged $F_1$ to the training set size for all the methods under discussion. The number of features was set to 500 for Naive Bayes methods. For SVM we used all the available terms. Also, we compared our initialization heuristic to the random one. Table 8 shows the EM behavior, using the LatentMPOS model with 500 features on the entire training collection. As compared to the random initialization, our similarity based heuristic does not gain much in terms of accuracy. However, it converges faster. Table 9 shows experimental results targeted at assessing the strength of the heuristic itself, without any EM iteration. The column *Heuristic* shows classification results using only the similarity-based initialization heuristic, compared to the performance achieved after one EM iteration (column *Heuristic-EM1*). Column *Random-EM1* shows the performance after one EM iteration with *random* initialization of parameters.

*Table 5.* Microavg $F_1$ for different number of features.

| Number of features | Microavg $F_1$ NBayes | Microavg $F_1$ LatentM | Microavg $F_1$ SVM |
|---|---|---|---|
| 100 | 75.9% | **78.3%** | 79.0% |
| 200 | 77.0% | **79.5%** | 80.0% |
| 300 | 78.3% | **81.0%** | 78.1% |
| 400 | 78.6% | **81.3%** | 76.8% |
| 500 | 78.7% | **81.8%** | 76.3% |
| 1,000 | 78.4% | **83.2%** | 73.6% |
| 2,000 | 71.6% | **83.5%** | 75.8% |
| 3,000 | 66.8% | **83.5%** | 78.3% |
| 5,000 | 61.2% | **83.1%** | 79.8% |
| 10,000 | 57.2% | **82.7%** | 81.3% |
| 16,000 | 55.0% | **82.4%** | 81.6% |

*Table 6.* Microavg $F_1$ for different number of PoS features.

| Number of features | Microavg $F_1$ NBayesPoS | Microavg $F_1$ LatentMPoS | Microavg $F_1$ SVMPoS |
|---|---|---|---|
| 100 | 77.5% | **79.0%** | 79.8% |
| 200 | 78.8% | **81.3%** | 80.2% |
| 300 | 79.4% | **81.9%** | 78.4% |
| 400 | 79.9% | **82.0%** | 77.8 % |
| 500 | 80.3% | **82.5%** | 76.2% |
| 1,000 | 79.9% | **83.5%** | 73.8% |
| 2,000 | 74.0% | **83.8%** | 76.2% |
| 3,000 | 69.7% | **83.8%** | 77.6% |
| 5,000 | 62.7% | **83.4%** | 79.4% |
| 10,000 | 56.8% | **83.1%** | 81.2% |
| 16,000 | 54.7% | **82.5%** | 81.7% |

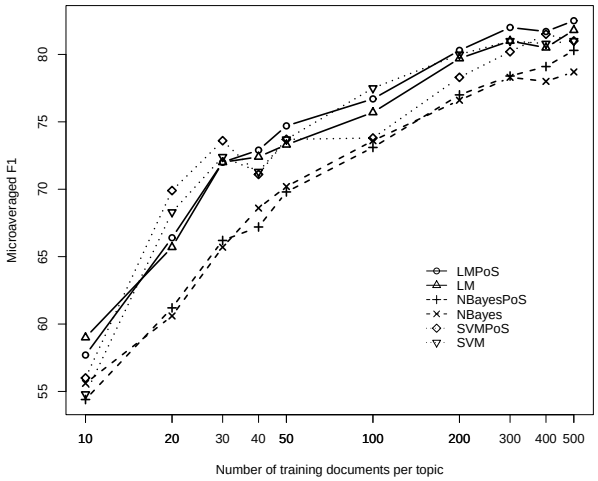


*Figure 3.* Microaveraged $F_1$ for different training set size.

*Table 7.* Microaveraged $F_1$ for different training set size.

| Training | Microavg $F_1$ NBayesPoS | Microavg $F_1$ LatentMPoS | Microavg $F_1$ SVMPoS |
|---|---|---|---|
| 10 | 54.4% | **57.7%** | 56.0% |
| 20 | 61.2% | **66.4%** | 69.9% |
| 30 | 66.2% | **71.9%** | 73.6% |
| 40 | 67.2% | **72.9%** | 71.1% |
| 50 | 69.8% | **74.7%** | 73.8% |
| 100 | 73.1% | **76.7%** | 78.3% |
| 200 | 77.0% | **80.3%** | 80.2% |
| 300 | 78.4% | **82.0%** | 81.5% |
| 400 | 79.1% | **81.7%** | 81.0% |
| 500 | 80.3% | **82.5%** | 81.7% |

*Table 8.* Sim-based vs random initialization.

| EM Iteration | Sim-based Init | Random Init |
|---|---|---|
| 1 | 80.5% | 59.0% |
| 2 | 81.5% | 70.6% |
| 3 | 81.9% | 76.5% |
| 4 | 82.2% | 79.8% |
| 5 | 82.3% | 80.9% |
| 10 | 82.5% | 82.3% |
| 15 | 82.5% | 82.4% |

*Table 9.* Heuristic, Heuristic & EM1, Random & EM1.

| Training | Heuristic | Heuristic-EM1 | Random-EM1 |
|---|---|---|---|
| 10 | 38.1% | 56.8% | 49.8% |
| 20 | 66.6% | 60.9% | 49.6% |
| 30 | 68.2% | 67.7% | 49.6% |
| 40 | 40.3% | 70.5% | 49.8% |
| 50 | 43.4% | 71.7% | 49.8% |
| 100 | 27.3% | 74.8% | 49.8 % |
| 200 | 29.9% | 79.3% | 49.8% |
| 300 | 27.6% | 80.8% | 51.0% |
| 400 | 30.4% | 80.3% | 51.0% |
| 500 | 32.3% | 80.5% | 52.0% |

### 4.4. Discussion

The results above clearly demonstrate the benefits of combining the initialization heuristic with EM; neither technique alone can achieve good performance. Further experiments are needed for a better understanding of the behavior of the proposed techniques.

## 5. Conclusions

In this paper, we proposed a generative language model approach to automatic document classification. Many similar models exist in the literature, but our approach is a step towards increasing the model robustness by introducing explicit information on the model and pruning the parameter space to only necessary data, encoded in the training collection. The approach proposed seems to be beneficial for collections with a rich natural language vocabulary, setups in which classical terms-only methods risk to be trapped in the semantic variations. Our future work includes more comprehensive experimental studies on various data collections and also studying the usage of different knowledge resources, such as customized ontologies extracted from large corpora.

## References


Baker, L. D., & McCallum, A. (1998). *Distributional Clustering of Words for Text Classification. Proceedings of the 21st ACM-SIGIR International Conference on Research and Development in Information Retrieval* (pp. 96–103).

Bhattacharya, I., & Getoor, L. & Bengio, Y. (2004). *Unsupervised Sense Disambiguation Using Bilingual Probabilistic Models. Meeting of the Association for Computational Linguistics.*

Bloehdorn, S., & Hotho, A. (2004). *Text Classification by Boosting Weak Learners based on Terms and Concepts. International Conference on Data Mining* (pp. 331–334).

Cai, L., & Hofmann, T. (2003). *Text Categorization by Boosting Automatically Extracted Concepts. 26th Annual International ACM-SIGIR Conference.*

Chakrabarti, S. (2003). *Mining the Web: Discovering Knowledge from Hypertext Data.* San Francisco: Morgan Kaufman.

Croft, W. B., & Lafferty, J. (2003). *Language Modeling for Information Retrieval.* Kluwer Academic Publishers.

Deerwester, S., & Dumais, S. T., & Harshman, R. (1990) *Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science* 41(6) (pp. 391–407).

Fellbaum, C. (1999). *WordNet: An Electronic Lexical Database.* Cambridge: MIT Press.

Hastie, T., & Tibshirani, R., & Friedman, J. H. (2003). *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* New York: Springer Verlag.

Hofmann, T. (2001). *Unsupervised Learning by Probabilistic Latent Semantic Analysis.* Kluwer Academic Publishers.

Joachims, T. (1998). *Text categorization with support vector machines: learning with many relevant features* (pp. 137–142). *Proceedings 10th European Conference on Machine Learning.*

Manning, C. D., & Schütze, H. (2000). *Foundations of Statistical Natural Language Processing.* Cambridge: MIT Press.

McCallum, A., & Nigam, K. (1998). *A Comparison of Event Models for Naive Bayes Text Classification. AAAI-98 Workshop on "Learning for Text Categorization".*

Scott, S., & Matwin, S. (1999). *Feature Engineering for Text Classification Proceedings of the Sixteenth International Conference on Machine Learning* (pp. 379–388)

Staab, S., & Studer, R. (2004). *Handbook on Ontologies* Berlin: Springer.

Theobald, M., & Schenkel, R., & Weikum, G. (2003). *Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data. Sixth International Workshop on the Web and Databases.*

Tsochantaridis, I., & Hoffman, T., & Joachims, T., & Altun Y. (2004). *Support Vector machine Learning for Interdependent and Structured Output Spaces. Proceedings of the 21st International Conference on Machine Learning.*