Feature Generation for Text Categorization Using World Knowledge

Evgeniy Gabrilovich and **Shaul Markovitch**
{gabr,shaulm}@cs.technion.ac.il
Computer Science Department, Technion, 32000 Haifa, Israel

Abstract

We enhance machine learning algorithms for text categorization with generated features based on domain-specific and common-sense knowledge. This knowledge is represented using publicly available ontologies that contain hundreds of thousands of concepts, such as the Open Directory; these ontologies are further enriched by several orders of magnitude through controlled Web crawling. Prior to text categorization, a feature generator analyzes the documents and maps them onto appropriate ontology concepts, which in turn induce a set of generated features that augment the standard bag of words. Feature generation is accomplished through contextual analysis of document text, implicitly performing word sense disambiguation. Coupled with the ability to generalize concepts using the ontology, this approach addresses the two main problems of natural language processing—synonymy and polysemy. Categorizing documents with the aid of knowledge-based features leverages information that cannot be deduced from the documents alone. Experimental results confirm improved performance, breaking through the plateau previously reached in the field.

1 Introduction

The state of the art systems for text categorization use induction algorithms in conjunction with word-based features ("bag of words"). After a decade of improvements, the performance of the best document categorization systems became more or less similar, and it appears as though a plateau has been reached, as neither system is considerably superior to others, and improvements are becoming evolutionary [Sebastiani, 2002].

The bag of words (BOW) approach is inherently limited, as it can only use pieces of information that are explicitly mentioned in the documents, and even that provided the same vocabulary is consistently used. Specifically, this approach has no access to the wealth of world knowledge possessed by humans, and is easily puzzled by facts and terms not mentioned in the training set.

To illustrate the limitations of the BOW approach, consider document #15264 in Reuters-21578, which is one of the most frequently used datasets in text categorization research. This document discusses a joint mining venture by a consortium of companies, and belongs to the category "copper". However, the document only briefly mentions that the aim of this venture is mining copper; instead, this fairly long document mainly talks about the mutual share holdings of the companies involved (Teck Corporation, Cominco, and Lornex Mining), as well as discusses other mining activities of the consortium. Consequently, three very different text classifiers we used (SVM, KNN and C4.5) failed to classify the document correctly. This comes as no surprise—"copper" is a fairly small category, and neither of these companies, nor the location of the venture (Highland Valley in British Columbia) are ever mentioned in the training set for this category. The failure of the bag of words approach is therefore unavoidable, as it cannot reason about the important components of the story.

We argue that this needs not be the case. When a Reuters editor originally handled this document, she most likely knew quite a lot about the business of these companies, and easily assigned the document to the category "copper". It is this kind of knowledge that we would like machine learning algorithms to have access to.

To date, quite a few attempts have been made to deviate from the orthodox bag of words paradigm, usually with limited success. In particular, representations based on phrases [Dumais *et al.*, 1998; Fuernkranz *et al.*, 2000], named entities [Kumaran and Allan, 2004], and term clustering [Lewis and Croft, 1990; Bekkerman, 2003] have been explored. In the above example, however, none of these techniques could overcome the underlying problem—lack of world knowledge.

In order to solve this problem and break through the existing performance barrier, a fundamentally new approach is apparently necessary. One possible solution is to completely depart from the paradigm of induction algorithms in an attempt to perform deep understanding of the document text. Yet, considering the current state of natural language processing systems, this does not seem to be a viable option (at least for the time being).

We propose an alternative solution that capitalizes on the power of existing induction techniques while enriching the language of representation, namely, exploring new feature spaces. Prior to text categorization, we employ a *feature generator* that uses common-sense and domain-specific knowledge to enrich the bag of words with new, more informative features. Feature generation is performed completely automatically, using machine-readable hierarchical repositories of knowledge such as the Open Directory Project (ODP), Yahoo! Web Directory, and the Wikipedia encyclopedia.

In this paper we use the ODP as a source of background knowledge. Thus, in the above example the feature generator "knows" that the companies mentioned are in the mining

business, and that Highland Valley happens to host a copper mine. This information is available in Web pages that discuss the companies and their operations, and are cataloged in corresponding ODP categories such as *Mining_and_Drilling* and *Metals*. Similarly, Web pages about Highland Valley are cataloged under *British_Columbia*. To amass this information, we crawl the URLs cataloged in the ODP, thus effectively multiplying the amount of knowledge available many times over. Armed with this knowledge, the feature generator constructs new features that denote these ODP categories, and adds them to the bag of words. The augmented feature space provides text classifiers with a cornucopia of additional information. Indeed, our implementation of the proposed methodology classifies this document correctly.

Feature generation (FG) techniques were found useful in a variety of machine learning tasks [Markovitch and Rosenstein, 2002; Fawcett, 1993; Matheus, 1991]. These techniques search for new features that describe the target concept better than the ones supplied with the training instances. A number of feature generation algorithms were proposed [Pagallo and Haussler, 1990; Matheus and Rendell, 1989; Hu and Kibler, 1996; Murphy and Pazzani, 1991], which led to significant improvements in performance over a range of classification tasks. However, even though feature generation is an established research area in machine learning, only a few works applied it to text processing [Kudenko and Hirsh, 1998; Mikheev, 1999; Cohen, 2000]. With the exception of a few studies using WordNet [Scott, 1998; Urena-Lopez *et al.*, 2001], none of them attempted to leverage repositories of world knowledge.

The contributions of this paper are threefold. First, we propose a framework and a collection of algorithms that perform feature generation based on very large-scale repositories of human knowledge. Second, we propose a novel kind of contextual analysis performed during feature generation, which views the document text as a sequence of local contexts, and performs implicit word sense disambiguation. Finally, we describe a way to further enhance existing knowledge bases by several orders of magnitude by crawling the World Wide Web. As we show in Section 3, our approach allows to break the performance barrier currently reached by the best text categorization systems.

2 Feature Generation Methodology

The proposed methodology allows principled and uniform integration of external knowledge to construct new features. In the preprocessing step we use knowledge repositories to build a feature generator. Applying the feature generator to the documents produces a set of generated features. These features undergo feature selection, and the most discriminative ones are added to the bag of words. Finally, we use traditional text categorization techniques to learn a text categorizer in the augmented feature space.

Suitable knowledge repositories satisfy the following requirements:

1. The repository contains a collection of *concepts* organized in a hierarchical tree structure, where edges represent the "is-a" relationship. Using a hierarchical ontology allows us to perform powerful generalizations.
2. There is a collection of texts associated with each concept. The feature generator uses these texts to learn the definition and scope of the concept, in order to be able to assign it to relevant documents.

We currently use the ODP as our knowledge base, however, our methodology is general enough to facilitate other sources of common-sense and domain-specific knowledge that satisfy the above assumptions. The Open Directory comprises a hierarchy of approximately 600,000 categories that catalog over 4,000,000 Web sites, each represented by a URL, a title, and a brief summary of its contents. The project constitutes an ongoing effort promoted by over 65,000 volunteer editors around the globe, and is arguably the largest publicly available Web directory. Being the result of *pro bono* work, the Open Directory has its share of drawbacks, such as non-uniform coverage, duplicate subtrees in different branches of the hierarchy, and sometimes biased coverage due to peculiar views of the editors in charge. Nonetheless, ODP embeds a colossal amount of human knowledge in a wide variety of areas, covering even most specific scientific and technical concepts. In what follows, we refer to the ODP category nodes as *concepts*, to avoid possible confusion with the term *categories*, which is usually reserved for the labels assigned to documents in text categorization.

2.1 Building a Feature Generator

We start with a preprocessing step, performed once for all future text categorization tasks. We induce a hierarchical text classifier that maps pieces of text onto relevant ODP concepts, which later serve as generated features. The resulting classifier is called a *feature generator* according to its true purpose in our scheme, as opposed to the text categorizer (or classifier) that we build ultimately. The feature generator represents ODP concepts as vectors of their most characteristic words, which we call *attributes* (reserving the term *features* to denote the properties of documents in text categorization). We now explain how the attribute vectors are built.

We use the textual descriptions of ODP nodes and their URLs as training examples for learning the feature generator. Although these descriptions alone constitute a sizeable amount of information, we devised a way to increase the volume of training data by several orders of magnitude. We do so by crawling the Web sites pointed at by all cataloged URLs, and obtain a small representative sample of each site. Pooling together the samples of all sites associated with an ODP node gives us a wealth of additional information about it.

Texts harvested from the WWW are often plagued with noise, and without adequate noise reduction crawled data may do more harm than good. To remedy the situation, we perform *attribute selection* for each ODP node; this can be done using any standard feature selection technique such as information gain. For example, consider the top 10 attributes selected for the ODP concepts *Science—science, research, scientific, biology, laboratory, analysis, university, theory, study, scientist*, and *Artificial_Intelligence—neural, artificial, algorithm, intelligence, AAAI, Bayesian, probability, IEEE, cognitive, inference*. Additional noise reduction is achieved by pruning nodes having too few URLs or situated too deep in

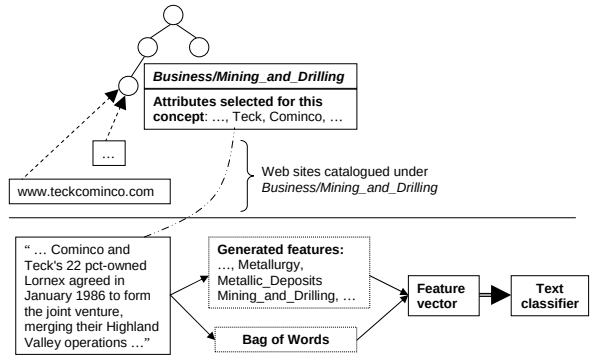

Figure 1: Feature generation example

the tree (and thus representing overly specific concepts), assigning their textual content to their parents.

In our current implementation, the feature generator works as a nearest neighbor classifier—it compares its input text to (the attribute vectors of) all ODP nodes, and returns the desired number of best-matching ones. The generator also performs *generalization* of concepts, and constructs features based on the classified concepts *per se* as well as their ancestors in the hierarchy.

Let us revisit the example from Section 1. While building the feature generator, our system crawls the Web sites cataloged under mining-related ODP concepts such as *Business/Mining_and_Drilling*, *Science/Technology/Mining* and *Business/Industrial_Goods_and_Services/Materials/Metals*. These include `www.teckcominco.com` and `www.miningsurplus.com` that belong to the (now merged) Teck Cominco company. Due to the company's prominence, it is mentioned frequently in the Web sites we have crawled, and consequently the words "Teck" and "Cominco" are included in the set of attributes selected to represent the above concepts. Figure 1 illustrates the process of feature generation for this example.

2.2 Contextual Feature Generation

Traditionally, feature generation uses the basic features supplied with the training instances to construct more sophisticated features. In the case of text processing, however, applying this approach to the bag of words leads to losing the important information about word ordering. Therefore, we argue that feature generation becomes much more powerful when it operates on the raw document text. But should the generator always analyze the whole document as a single unit, similarly to regular text classifiers?

We believe that considering the entire document may often be misleading, as its text can be too diverse to be readily mapped to the right set of concepts, while notions mentioned only briefly may be overlooked. Instead, we propose to partition the document into a series of non-overlapping segments (called *contexts*), and then generate features at this finer level. Each context is classified into a number of concepts in the knowledge base, and pooling these concepts together results in *multi-faceted* classification for the document. This way, the resulting set of concepts represents the various aspects or sub-topics covered by the document.

Potential candidates for such contexts are simple sequences of words, or more linguistically motivated chunks such as sentences or paragraphs. The optimal resolution for document segmentation can be determined automatically using a validation set. We propose a more principled *multi-resolution* approach that simultaneously partitions the document at several levels of linguistic abstraction (windows of words, sentences, paragraphs, up to taking the entire document as one big chunk), and performs feature generation at each of these levels. We rely on the subsequent *feature selection* step to eliminate extraneous features, preserving only those that genuinely characterize the document.

In fact, the proposed approach tackles the two most important problems in natural language processing, namely, *synonymy* and *polysemy*. Classifying individual contexts implicitly performs *word sense disambiguation*, and thus resolves word polysemy to some degree. A context that contains one or more polysemous words is mapped to the concepts that correspond to the sense *shared* by the context words. Thus, the correct sense of each word is determined with the help of its neighbors. At the same time, enriching document representation with high-level concepts and their generalizations addresses the problem of synonymy, as the enhanced representation can easily recognize that two (or more) documents actually talk about related issues, even though they do so using different vocabularies.

Let us again revisit our running example. During feature generation, document #15264 is segmented into a sequence of contexts, which are then mapped to mining-related ODP concepts (e.g., *Business/Mining_and_Drilling*). These concepts, as well as their ancestors in the hierarchy, give rise to a set of generated features that augment the bag of words (see Figure 1). Observe that the training documents for the category "copper" underwent a similar processing when a text classifier was induced. Consequently, features based on these concepts *were selected* during feature selection thanks to their high predictive capacity. It is due to these features that the document is now categorized correctly, while without feature generation it consistently caused BOW classifiers to err.

2.3 Feature Selection

Using support vector machines in conjunction with bag of words, Joachims [1998] found that SVMs are very robust even in the presence of numerous features, and further observed that the multitude of features are indeed useful for text categorization. These findings were corroborated in more recent studies [Brank *et al.*, 2002; Bekkerman, 2003] that observed either no improvement or even small degradation of SVM performance after feature selection.[1] Consequently, many later works using SVMs did not apply feature selection at all [Leopold and Kindermann, 2002; Lewis *et al.*, 2004].

This situation changes drastically as we augment the bag of words with generated features. First, nearly any technique for automatic feature generation can easily generate huge numbers of features, which will likely aggravate the "curse of dimensionality". Furthermore, it is feature selection that allows the feature generator not to be a perfect classifier. When at

[1] Gabrilovich and Markovitch [2004] described a class of problems where feature selection from the bag of words actually improves SVM performance.

least some of the concepts assigned to the document are correct, feature selection can identify them and seamlessly eliminate the spurious ones.

3 Empirical Evaluation

We implemented the proposed methodology using an ODP snapshot as of April 2004.

3.1 Implementation Details

After pruning the *Top/World* branch that contains non-English material, we obtained a hierarchy of over 400,000 concepts and 2,800,000 URLs. Applying our methodology to a knowledge base of this scale required an enormous engineering effort. Textual descriptions of the concepts and URLs amounted to 436 Mb of text. In order to increase the amount of information for training the feature generator, we further populated the ODP hierarchy by crawling all of its URLs, and taking the first 10 pages (in the BFS order) encountered at each site. This operation yielded 425 Gb worth of HTML files. After eliminating all the markup and truncating overly long files at 50 Kb, we ended up with 70 Gb of additional textual data. Compared to the original 436 Mb of text supplied with the hierarchy, we obtained over a 150-fold increase in the amount of data. After removing stop words and rare words (occurring in less than 5 documents) and stemming the remaining ones, we obtained 20,700,000 distinct terms that were used to represent ODP nodes as attribute vectors. Up to 1000 most informative attributes were selected for each ODP node using the Document Frequency criterion (other commonly used feature selection techniques, such as Information Gain, χ^2 and Odds Ratio, yielded slightly inferior results). We used the *multi-resolution* approach for feature generation, classifying document contexts at the level of words, sentences, paragraphs, and finally the entire document.Features were generated from the 10 best-matching ODP concepts for each context.

3.2 Experimental Methodology

The following test collections were used:

1. Reuters-21578 [Reuters, 1997]. Following common practice, we used the ModApte split (9603 training, 3299 testing documents) and two category sets, 10 largest categories and 90 categories with at least one training and testing example.

2. Reuters Corpus Volume I (RCV1) [Lewis *et al.*, 2004] has over 800,000 documents, and presents a new challenge for text categorization. To speedup experimentation, we used a subset of the corpus with 17808 training documents (dated 20–27/08/96) and 5341 testing ones (28–31/08/96). Following Brank *et al.* [2002], we used 16 Topic and 16 Industry categories that constitute a representative sample of the full groups of 103 and 354 categories, respectively. We also randomly sampled the Topic and Industry categories into several sets of 10 categories each (Table 1 shows 3 category sets in each group with the highest improvement in categorization performance).[2]

3. 20 Newsgroups (20NG) [Lang, 1995] is a well-balanced dataset of 20 categories containing 1000 documents each.

4. Movie Reviews (Movies) [Pang *et al.*, 2002] defines a sentiment classification task, where reviews express either positive or negative opinion about the movies. The dataset has 1400 documents in two categories (positive/negative).

We used support vector machines[3] as our learning algorithm to build text categorizers, since prior studies found SVMs to have the best performance for text categorization [Dumais *et al.*, 1998; Yang and Liu, 1999]. Following established practice, we use the precision-recall Break-Even Point (BEP) to measure text categorization performance. For the two Reuters datasets we report both micro- and macro-averaged BEP, since their categories differ in size significantly. Micro-averaged BEP operates at the document level and is primarily affected by categorization performance on larger categories. On the other hand, macro-averaged BEP averages results for individual categories, and thus small categories with few training examples have large impact on the overall performance. For both Reuters datasets we used a fixed data split, and consequently used macro sign test (S-test) [Yang and Liu, 1999] to assess the statistical significance of differences in classifier performance. For 20NG and Movies we performed 4-fold cross-validation, and used paired t-test to assess the significance.

3.3 The Effect of Feature Generation

We first demonstrate that the performance of basic text categorization in our implementation (column "Baseline" in Table 1) is consistent with other published studies (all using SVM). On Reuters-21578, Dumais *et al.* [1998] achieved micro-BEP of 0.920 for 10 categories and 0.870 for all categories. On 20NG, Bekkerman [2003] obtained BEP of 0.856. Pang *et al.* [2002] obtained accuracy of 0.829 on Movies. The minor variations in performance are due to differences in data preprocessing used in different systems; for example, for the Movies dataset we worked with raw HTML files rather than with the official tokenized version, in order to recover sentence and paragraph structure for contextual analysis. For RCV1, direct comparison with published results is more difficult, as we limited the category sets and the date span of documents to speedup experimentation.

Table 1 shows the results of using feature generation with significant improvements ($p < 0.05$) shown in bold. For both Reuters datasets, we consistently observed larger improvements in macro-averaged BEP, which is dominated by categorization effectiveness on small categories. This goes in line with our expectations that the contribution of external knowledge should be especially prominent for categories with few training examples. As can be readily seen, categorization performance was improved for all datasets, with notable improvements of up to 25.4% for Reuters RCV1 and 3.6% for Movies. Given the performance plateau currently reached by the best text categorizers, these results clearly demonstrate the advantage of knowledge-based feature generation.

3.4 Actual Examples Under a Magnifying Glass

Thanks to feature generation our system correctly classifies the running example document #15264. Let us consider additional testing examples from Reuters-21578 that are incor-

[2] Full definition of the category sets we used is available at `http://www.cs.technion.ac.il/~gabr/ijcai2005-appendix.html`.

[3] SVM^{light} implementation [Joachims, 1998].

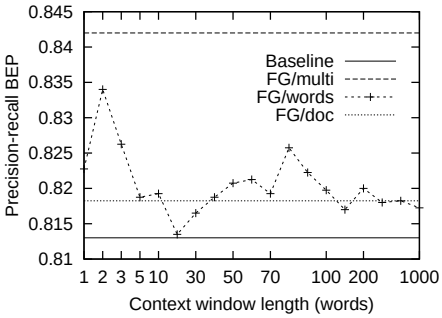

Figure 2: Varying context length (Movies)

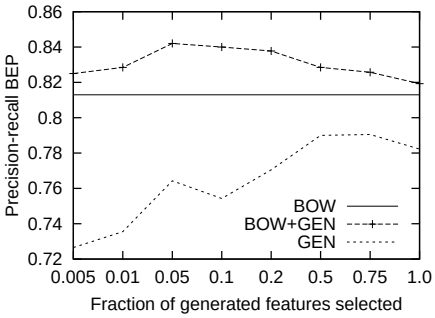

Figure 3: Feature selection (Movies)

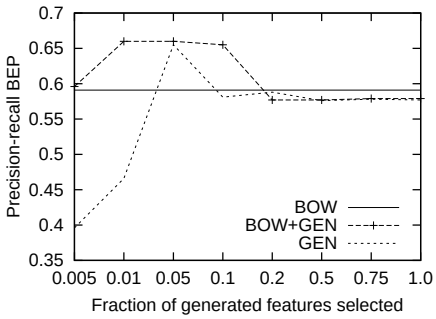

Figure 4: Feature selection (RCV1/Topic-16)

Dataset	Baseline		Feature generation		Improvement vs. baseline	
	micro BEP	macro BEP	micro BEP	macro BEP	micro BEP	macro BEP
Reuters-21578						
10 categories	0.925	0.874	0.930	0.884	+0.5%	+1.1%
90 categories	0.877	0.602	0.880	0.614	+0.3%	+2.0%
RCV1						
Industry-16	0.642	0.595	0.648	**0.613**	+0.9%	**+3.0%**
Industry-10A	0.421	0.335	**0.457**	**0.420**	**+8.6%**	**+25.4%**
Industry-10B	0.489	0.528	**0.530**	**0.560**	**+8.4%**	**+6.1%**
Industry-10C	0.443	0.414	**0.468**	**0.463**	**+5.6%**	**+11.8%**
Topic-16	0.836	0.591	0.840	**0.660**	+0.5%	**+11.7%**
Topic-10A	0.796	0.587	0.806	**0.695**	+1.3%	**+18.4%**
Topic-10B	0.716	0.618	0.731	**0.651**	+2.1%	**+5.3%**
Topic-10C	0.719	0.616	0.727	**0.646**	+1.1%	**+4.9%**
20NG	0.854		**0.858**		**+0.5%**	
Movies	0.813		**0.842**		**+3.6%**	

Table 1: Text categorization with and without feature generation

rectly categorized by the BOW classifier. Document #16143 belongs to the category "money-fx" (money/foreign exchange) and discusses the devaluation of the Kenyan shilling. Even though "money-fx" is one of the 10 largest categories, the word "shilling" does not occur in its training documents even once. However, the feature generator easily recognizes it as a kind of currency, and produces features such as *Recreation/Collecting/Paper_Money* and *Recreation/Collecting/Coins/World_Coins*. These high-level features were also constructed for many training examples, and consequently the document is now classified correctly.

Similarly, document #18748 discusses Italy's balance of payments and belongs to the category "trade", while the word "trade" itself does not occur in this short document. However, when the feature generator considers document contexts discussing Italian deficit as reported by the Bank of Italy, it correctly maps them to concepts such as *Society/Government/Finance*, *Society/Issues/Economic/International/Trade*, *Business/International_Business_and_Trade*. Due to these features, which were also generated for training documents in this category, the document is now categorized correctly.

3.5 The Effect of Contextual Analysis

We now explore the various possibilities to define document contexts for feature generation. Figure 2 shows how text categorization performance on the Movies dataset changes for various contexts. The x-axis measures context length in words, and the *FG/words* curve corresponds to applying the feature generator to the context of that size. With these word-level contexts, maximum performance is achieved when using pairs of words (x=2). The *FG/doc* line shows the result of using the entire document as a single context. In this case, the results are somewhat better than without feature generation (*Baseline*), but are still inferior to the more fine-grained word-level contexts (*FG/words*). However, the best performance by far is achieved when using the multi-resolution approach (*FG/multi*), in which we use a series of linguistically motivated chunks of text, starting with individual words, and then generating features from sentences, paragraphs, and finally the entire document.

3.6 The Utility of Feature Selection

Under the experimental settings defined in Section 3.1, feature generation constructs approximately 4–5 times as many features as are in the bag of words. We conducted two experiments to understand the effect of feature selection in conjunction with feature generation.

Since earlier studies found that most BOW features are indeed useful for SVM text categorization (Section 2.3), in our first experiment we only apply feature selection to the generated features, and use the selected ones to augment the (entire) bag of words. In Figures 3 and 4, the *BOW* line depicts the baseline performance without generated features, while the *BOW+GEN* curve shows the performance of the bag of words augmented with progressively larger fractions of generated features (sorted by information gain). For both datasets, the performance peaks when only a small fraction of the generated features are used, while retaining more generated features has a noticeable detrimental effect. Similar phenomena have been observed for other datasets; we omit the results owing to lack of space.

Our second experiment was set up to examine the performance of the generated features alone, without the bag of words (*GEN* curve in Figures 3 and 4). For Movies, discarding the BOW features hurts the performance somewhat, but the decrease is far less significant than what could be expected—using only the generated features we lose less than 3% in BEP compared with the BOW baseline. For 20NG, a similar experiment sacrifices about 10% off the BOW performance, as this dataset is known to have a very diversified vocabulary, for which many studies found feature selection to be particularly harmful. However, the situation is reversed for both Reuters datasets. For Reuters-21578, the generated features alone yield a 0.3% improvement in micro- and

macro-BEP for 10 categories, while for 90 categories they only lose 0.3% in micro-BEP and 3.5% in macro-BEP compared with the bag of words. For RCV1/Industry-16, disposing of the bag of words hurts the BEP by 1–3%. Surprisingly, for RCV1/Topic-16 (Figure 4) the generated features *per se* command a 10.8% improvement in macro-BEP, rivalling the performance of *BOW+GEN* that only gains another 1% improvement (Table 1). We interpret these findings as a further reinforcement of the quality of representation due to the generated features.

4 Conclusions and Future Work

We proposed a feature generation methodology for text categorization. In order to render machine learning algorithms with common-sense and domain-specific knowledge possessed by humans, we use large hierarchical knowledge bases to build a *feature generator*. The latter analyzes documents prior to text categorization, and augments the conventional bag of words representation with relevant concepts from the knowledge base. The enriched representation contains information that cannot be deduced from the document text alone.

We further described the multi-resolution analysis that examines the document text at several levels of linguistic abstraction, and performs feature generation at each level. Considering polysemous words in their native context implicitly performs word sense disambiguation, and allows the feature generator to cope with word synonymy and polysemy.

Empirical evaluation definitively confirmed that knowledge-based feature generation brings text categorization to a new level of performance. Interestingly, the sheer breadth and depth of the ODP, further boosted by crawling the URLs cataloged in the directory, brought about improvements both in regular text categorization as well as in the (non-topical) sentiment classification task.

We believe that this research only scratches the surface of what can be achieved with knowledge-rich features. In our future work, we plan to investigate new algorithms for mapping document contexts onto hierarchy concepts, as well as new techniques for selecting attributes that are most characteristic of every concept. We intend to apply focused crawling to only collect relevant web pages when crawling cataloged URLs. In addition to the ODP, we also plan to make use of domain-specific hierarchical knowledge bases, such as the Medical Subject Headings (MeSH). Finally, we conjecture that knowledge-based feature generation will also be useful for other information retrieval tasks beyond text categorization, and we intend to investigate this in our future work.

Acknowledgments

We thank Lev Finkelstein and Alex Gontmakher for many a helpful discussion. This research was partially supported by Technion's Counter-Terrorism Competition and by the MUSCLE Network of Excellence.

References

[Bekkerman, 2003] R. Bekkerman. Distributional clustering of words for text categorization. Master's thesis, Technion, 2003.

[Brank *et al.*, 2002] J. Brank, M. Grobelnik, N. Milic-Frayling, and D. Mladenic. Interaction of feature selection methods and linear classification models. In *Text Learning Workshop, ICML*, 2002.

[Cohen, 2000] W. Cohen. Automatically extracting features for concept learning from the web. In *ICML'00*, 2000.

[Dumais *et al.*, 1998] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM'98*, 1998.

[Fawcett, 1993] T. Fawcett. *Feature Discovery for Problem Solving Systems*. PhD thesis, UMass, May 1993.

[Fuernkranz *et al.*, 2000] J. Fuernkranz, T. Mitchell, and E. Riloff. A case study in using linguistic phrases for text categorization on the WWW. *Learning for Text Categorization*. AAAI Press, 2000.

[Gabrilovich and Markovitch, 2004] E. Gabrilovich and S. Markovitch. Text categorization with many redundant features: Using aggressive feature selection to make SVMs competitive with C4.5. In *ICML'04*, pages 321–328, 2004.

[Hu and Kibler, 1996] Y. Hu and D. Kibler. A wrapper approach for constructive induction. In *AAAI-96*, 1996.

[Joachims, 1998] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *ECML'98*, pages 137–142, 1998.

[Kudenko and Hirsh, 1998] D. Kudenko and H. Hirsh. Feature generation for sequence categorization. In *AAAI'98*, 1998.

[Kumaran and Allan, 2004] G. Kumaran and J. Allan. Text classification and named entities for new event detection. *SIGIR*, 2004.

[Lang, 1995] K. Lang. Newsweeder: Learning to filter netnews. In *ICML'95*, pages 331–339, 1995.

[Leopold and Kindermann, 2002] E. Leopold and J. Kindermann. Text categorization with support vector machines: How to represent texts in input space. *Machine Learning*, 46, 2002.

[Lewis and Croft, 1990] D. Lewis and W. Croft. Term clustering of syntactic phrases. In *SIGIR'90*, 1990.

[Lewis *et al.*, 2004] D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *JMLR*, 5:361–397, 2004.

[Markovitch and Rosenstein, 2002] S. Markovitch and D. Rosenstein. Feature generation using general constructor functions. *Machine Learning*, 49:59–98, 2002.

[Matheus and Rendell, 1989] C. Matheus and L. Rendell. Constructive induction on decision trees. *11th Int'l Conf. on AI*, 1989.

[Matheus, 1991] C. Matheus. The need for constructive induction. In *8th Int'l Workshop on Machine Learning*, 1991.

[Mikheev, 1999] A. Mikheev. Feature lattices and maximum entropy models. *Information Retrieval*, 1999.

[Murphy and Pazzani, 1991] P. Murphy and M. Pazzani. ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees. In *ICML'91*, 1991.

[Pagallo and Haussler, 1990] G. Pagallo and D. Haussler. Boolean feature discovery in empirical learning. *ICML'90*, 1990.

[Pang *et al.*, 2002] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *EMNLP'02*, 2002.

[Reuters, 1997] *Reuters-21578 text categorization test collection*. `daviddlewis.com/resources/testcollections/reuters21578`.

[Scott, 1998] S. Scott. Feature engineering for a symbolic approach to text classification. Master's thesis, U. Ottawa, 1998.

[Sebastiani, 2002] F. Sebastiani. Machine learning in automated text categorization. *ACM Comp. Surveys*, 34(1), 2002.

[Urena-Lopez *et al.*, 2001] L. Urena-Lopez, M. Buenaga, and J. Gomez. Integrating linguistic resources in TC through WSD. *Computers and the Humanities*, 35, 2001.

[Yang and Liu, 1999] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR'99*, pages 42–49, 1999.