RC23135 (W0403-028) March 2, 2004
Computer Science

IBM Research Report

Semantic Lexicon Construction: Learning from Unlabeled Data via Spectral Analysis

Rie Kubota Ando
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

LIMITED DISTRIBUTION NOTICE: This report has been submitted for publication outside of IBM and will probably be copyrighted if accepted for publication. It has been issued as a Research Report for early dissemination of its contents. In view of the transfer of copyright to the outside publisher, its distribution outside of IBM prior to publication should be limited to peer communications and specific requests. After outside publication, requests should be filled only by reprints or legally obtained copies of the article (e.g, payment of royalties). Copies may be requested from IBM T. J. Watson Research Center , P. O. Box 218, Yorktown Heights, NY 10598 USA (email: reports@us.ibm.com). Some reports are available on the internet at http://domino.watson.ibm.com/library/CyberDig.nsf/home

Semantic Lexicon Construction: Learning from Unlabeled Data via Spectral Analysis

Rie Kubota Ando
IBM T.J. Watson Research Center
19 Skyline Dr., Hawthorne, NY 10532
rie1@us.ibm.com

Abstract

This paper considers the task of automatically collecting words with their entity class labels, starting from a small number of labeled examples ('seed' words). We show that spectral analysis is useful for compensating for the paucity of labeled examples by learning from unlabeled data. The proposed method significantly outperforms a number of methods that employ techniques such as EM and co-training. Furthermore, when trained with 300 labeled examples and unlabeled data, it rivals Naive Bayes classifiers trained with 7500 labeled examples.

1 Introduction

Entity detection plays an important role in information extraction systems. Whether entity recognizers employ machine learning techniques or rule-based approaches, it is useful to have a gazetteer of words[1] that reliably suggest target entity class membership. This paper considers the task of generating such gazetteers from a large unannotated corpus with minimal manual effort. Starting from a small number of labeled examples (*seeds*), e.g., { "car", "plane", "ship" } labeled as vehicles, we seek to automatically collect more of these.

This task is sometimes called the semi-automatic construction of *semantic lexicons*, e.g. (Riloff and Shepherd, 1997; Roark and Charniak, 1998; Thelen and Riloff, 2002; Phillips and Riloff, 2002). A common trend in prior studies is bootstrapping, which is an iterative process to collect new words and regard the words newly collected with high confidence as additional labeled examples for the next iteration. The aim of bootstrapping is to compensate for the paucity of labeled examples. However, its potential danger is label 'contamination' — namely, wrongly (automatically) labeled examples may misdirect the succeeding iterations. Also, low frequency words are known to be problematic. They do not provide sufficient corpus statistics (e.g., how frequently the word occurs as the subject of "said"), for adequate label prediction.

[1]Our argument in this paper holds for relatively small linguistic objects including words, phrases, collocations, and so forth. For simplicity, we refer to words.

By contrast, we focus on improving feature vector representation for use in standard linear classifiers. To counteract data sparseness, we employ *subspace projection* where subspaces are derived by *singular value decomposition (SVD)*. In this paper, we generally call such SVD-based subspace construction *spectral analysis*.

Latent Semantic Indexing (LSI) (Deerwester et al., 1990) is a well-known application of spectral analysis to word-by-document matrices. Formal analyses of LSI were published relatively recently, e.g., (Papadimitriou et al., 2000; Azar et al., 2001). Ando and Lee (2001) show the factors that may affect LSI's performance by analyzing the conditions under which the LSI subspace approximates an *optimum subspace*. Our theoretical basis is partly derived from this analysis. In particular, we replace the abstract notion of 'optimum subspace' with a precise definition of a subspace useful for our task.

The essence of spectral analysis is to capture the most prominently observed vector directions (or sub-vectors) into a subspace. Hence, we should apply spectral analysis *only* to 'good' feature vectors so that useful portions are captured into the subspace, and then factor out 'harmful' portions of *all* the vectors via subspace projection. We first formalize the notion of harmful portions of the commonly used feature vector representation. Experimental results show that this new strategy significantly improves label prediction performance. For instance, when trained with *300 labeled examples* and unlabeled data, the proposed method rivals Naive Bayes classifiers trained with *7500 labeled examples*.

In general, generation of labeled training data involves expensive manual effort, while *unlabeled data* can be easily obtained in large amounts. This fact has motivated supervised learning with unlabeled data, such as *co-training* (e.g., Blum and Mitchell (1998)). The method we propose (called *Spectral*) can also be regarded as exploiting unlabeled data for supervised learning. The main difference from co-training or popular EM-based approaches is that the process of learning from unlabeled data (via spectral analysis) does not use any class information. It encodes learned information into feature vectors – which essentially serves as *prediction of unseen feature occurrences* – for use in supervised classification. The absence of class information during the learning process may seem to be disadvantageous. On the contrary, our experiments show that Spectral *consistently outperforms all the tested methods* that employ techniques such as EM and co-training.

We formalize the problem in Section 2, and propose the method in Section 3. We discuss related work in Section 4. Experiments are reported in Section 5, and we conclude in Section 6.

2 Word Classification Problem

The problem is to classify *words* (as *lexical items*) into the entity classes that are most likely referred to by their *occurrences*, where the notion of 'most likely' is with respect to the domain of the text[2].

More formally, consider all the possible instances of word occurrences (including their context) in the world, which we call set X, and assume that each word occurrence in X refers to one of the entity classes in set C (e.g., $C = \{$ 'Person', 'Location', 'Others' $\}$). Further assume that observed word occurrences (i.e., corpora) are independently drawn from X according to some probability distribution Ω. An example of Ω might be the distribution observed in all the newspaper articles in 1980's, or the distribution observed in biomedical articles. That is, Ω represents the assumed domain of text.

We define $\mathcal{C}_\Omega(w)$ to be the entity class most likely referred to by word w's occurrences in the assumed domain of text, i.e., $\mathcal{C}_\Omega(w) = \arg\max_{c \in C} \mathrm{P}(x \text{ refers to } c \mid x \text{ is an occurrence of } w)$, given that x is arbitrarily drawn from X according to Ω. Then, our word classification problem is to predict $\mathcal{C}_\Omega$-labels of all the words (as lexical items) in a given word set W, when the following resources are available:

- An unannotated corpus of the domain of interest – which we regard as unlabeled word occurrences arbitrarily drawn from X according to Ω. We assume that all the words in W appear in this corpus.
- Feature extractors. We assume that some *feature extractors* $\mathcal{F}$ are available, which we can apply to word occurrences in the above unannotated corpus. Feature $\mathcal{F}(x)$ might be, for instance, the set of head nouns that participate in list construction with the focus word of x.
- Seed words and their $\mathcal{C}_\Omega$ labels. We assume that the $\mathcal{C}_\Omega$-labels of several words in W are revealed as labeled examples.

Note that in this task configuration, *test data is known at the time of training* (as in the *transductive setting*). Although we do not pursue transductive learning techniques (e.g., Vapnik (1998)) in this work, we will set up the experimental framework accordingly.

3 Using Vector Similarity

3.1 Error Factors

Consider a straightforward feature vector representation using normalized joint counts of features and the word, which we call *count vector* $\vec{\#}_w$. More formally, the i-th element of $\vec{\#}_w$ is $\#(f_i, w)/\#(w)$ where $\#(\cdot)$ denotes the count of events observed in the unannotated corpus.

One way to classify words would be to compare count vectors for seeds and words and to choose the most similar seeds, using inner products as the similarity measure. Let us investigate the factors that may affect the performance of such inner product-based label prediction. Let $\vec{p}_w$ (for word w) and $\vec{p}_c$ (for class c) be the vectors of feature occurrence probabilities, so that their i-th elements are $\mathrm{P}(f_i|w)$ [3] and $\mathrm{P}(f_i|c)$, respectively. Now we set vectors $\vec{\varepsilon}_w$ and $\vec{\delta}_w$ so that they satisfy:

$$\begin{aligned}\vec{\#}_w &= \vec{\varepsilon}_w + \vec{p}_w \\ &= \vec{\varepsilon}_w + \vec{\delta}_w + \sum_{c \in C} \mathrm{P}(c|w)\,\vec{p}_c\ .\end{aligned}$$

That is, $\vec{\varepsilon}_w$ is a vector of the difference between true (but unknown) feature occurrence probabilities and their maximum likelihood estimations. We call $\vec{\varepsilon}_w$ *estimation error*.

If occurrences of word w and features are conditionally independent given labels, then $\vec{\delta}_w$ is zero[4]. Therefore, we call $\vec{\delta}_w$, *dependency*. It would be ideal (even if unrealistic) if the dependency were zero so that features convey class information rather than information specific to w.

Now consider the conditions under which a word pair with the same label has a larger inner product than the pair with different labels. It is easy to show that, with feature extractors fixed to reasonable ones, *smaller estimation errors and smaller dependency ensure better performance* of label prediction, in terms of lower-bound analysis. More precise descriptions are found in the Appendix.

[2]E.g., "plant" might be *most likely* to be a living thing if it occurred in *gardening books*, but it might be *most likely* to be a facility in *newspaper articles*.

[3]$\mathrm{P}(f_i|w)$ denotes the probability that feature f_i is in $\mathcal{F}(x)$ given that x is an occurrence of word w, where x is randomly drawn from X according to Ω.

[4]Because their conditional independence implies $\mathrm{P}(f_i|w) = \sum_{c \in C} \mathrm{P}(f_i|c)\mathrm{P}(c|w)$.

3.2 Spectral analysis for classifying words

We seek to remove the above harmful portions $\vec{\delta}_w$ and $\vec{\varepsilon}_w$ from count vectors — which correspond to estimation error and feature dependency — by employing spectral analysis and succeeding subspace projection.

Background A brief review of spectral analysis is found in the Appendix. Ando and Lee (2001) analyze the conditions under which the application of spectral analysis to a term-document matrix (as in LSI) approximates an *optimum subspace*. The notion of 'optimum' is with respect to the accuracy of topic-based document similarities. The proofs rely on the mathematical findings known as the *invariant subspace perturbation theorems* proved by Davis and Kahan (1970).

Approximation of the span of $\vec{p}_c$'s By adapting Ando and Lee's analysis to our problem, it can be shown that spectral analysis will approximate the span of $\vec{p}_c$'s, essentially,

- if the count vectors (chosen as input to spectral analysis) well-represent all the classes, and
- if these input vectors have sufficiently small estimation errors and dependency.

This is because, intuitively, $\vec{p}_c$'s are the most prominently observed sub-vectors among the input vectors in that case. (Recall that the essence of spectral analysis is to capture the most prominently observed vector directions into a subspace.) Then, the error portions can be mostly removed from *any* count vectors by orthogonally projecting the vectors onto the subspace, assuming error portions are mostly orthogonal to the span of $\vec{p}_c$'s.

Choice of count vectors As indicated by the above two conditions, the choice of input vectors is important when applying spectral analysis. The tightness of subspace approximation depends on the degree to which those conditions are met. In fact, it is easy to choose vectors with small estimation errors so that the second condition is likely to be met. Vectors for high frequency words are expected to have small estimation errors. Hence, we propose the following procedure.

1. From the unlabeled word set W, choose the k most frequent words. k is a sufficiently large constant. Frequency is counted in the given unannotated corpus.
2. Generate count vectors for all the k words by applying a feature extractor to word occurrences in the given unannotated corpus.
3. Compute the h-dimensional subspace by applying spectral analysis to the k count vectors generated in Step 2 [5].
4. Generate count vectors (as in Step 2) for all the words (including seeds) in W. Generate new feature vectors by orthogonally projecting them onto the subspace[6].

When we have multiple feature extractors, we perform the above procedure independently for each of the feature extractors, and concatenate the vectors in the end. Hereafter, we call this procedure and the vectors obtained in this manner *Spectral* and *spectral vectors*, respectively. Spectral vectors serve as feature vectors for a linear classifier for classifying words.

Note that we do *not* claim that the above conditions for subspace approximation are always satisfied. Rather, we consider them as insight into spectral analysis on this task, and design the method so that the conditions are likely to be met.

3.3 The number of input vectors and the subspace dimensionality

There are two parameters: k, the number of count vectors used as input to spectral analysis, and h, the dimensionality of the subspace.

k should be sufficiently large so that all the classes are represented by the chosen vectors. However, an excessively large k would result in including low frequency words, which might degrade the subspace approximation.

In principle, the dimensionality of the subspace h should be set to the number of classes $|C|$, since we seek to approximate the span of $\vec{p}_c$'s for all $c \in C$. However, for the typical practice of semantic lexicon construction, h should be greater than $|C|$ because at least one class tends to have very broad coverage – 'Others' as in { Person, Organization, Others }. It is reasonable to assume that features correlate to its (unknown) inherent subclasses rather than to such a broadly defined class itself. The dimensionality h should take account of the number of such subclasses.

In practice, k and h need be determined empirically. We will return to this issue in Section 5.2.

[5]We generate a matrix so that its columns are the k length-normalized count vectors. We compute left singular vectors of this matrix corresponding to the h largest singular values. The computed left singular vectors are the basis vectors of the desired subspace.

[6]We compute $\sum_{i=1}^{h} \mathbf{u}_i \mathbf{u}_i^T \vec{\#}_w$ where $\mathbf{u}_i$ is the left singular vector computed in the previous step. Alternatively, one can generate the vector whose i-th entry is $\mathbf{u}_i^T \vec{\#}_w$, as it produces the same inner products, due to the orthonormality of left singular vectors.

4 Related Work and Discussion

4.1 Spectral analysis for word similarity measurement

Spectral analysis has been used in traditional factor analysis techniques (such as *Principal Component Analysis*) to summarize high-dimensional data. LSI uses spectral analysis for measuring document or word similarities. From our perspective, the LSI word similarity measurement is similar to the special case where we have a single feature extractor that returns the document membership of word occurrence x.

Among numerous empirical studies of LSI, Landauer and Dumais (1997) report that using the LSI word similarity measure, 64.4% of the synonym section of TOEFL (multi-choice) were answered correctly, which rivals college students from non-English speaking countries. We conjecture that if more effective feature extractors were used, performance might be better.

Schüetze (1992)'s word sense disambiguation method uses spectral analysis for vector dimensionality reduction. He reports that use of spectral analysis does not affect the task performance, either positively or negatively.

4.2 Bootstrapping methods for constructing semantic lexicons

A common trend for the semantic lexicon construction task is that of bootstrapping, exploiting strong syntactic cues — such as a bootstrapping method that iteratively grows seeds by using cooccurrences in lists, conjunctions, and appositives (Roark and Charniak, 1998); *meta-bootstrapping* which repeatedly finds extraction patterns and extracts words from the found patterns (Riloff and Jones, 1999); a co-training combination of three bootstrapping processes each of which exploits appositives, compound nouns, and ISA-clauses (Phillips and Riloff, 2002). Thelen and Riloff (2002)'s bootstrapping method iteratively performs feature selection and word selection for each class. It outperformed the best-performing bootstrapping method for this task at the time. We also note that there are a number of bootstrapping methods successfully applied to text – e.g., word sense disambiguation (Yarowsky, 1995), named entity instance classification (Collins and Singer, 1999), and the extraction of 'parts' word given the 'whole' word (Berland and Charniak, 1999).

In Section 5, we report experiments using syntactic features shown to be useful by the above studies, and compare performance with Thelen and Riloff (2002)'s bootstrapping method.

4.3 Techniques for learning from unlabeled data

While most of the above bootstrapping methods are targeted to NLP tasks, techniques such as EM and co-training are generally applicable when equipped with appropriate models or classifiers. We will present high-level and empirical comparisons (Sections 4.4 and 5, respectively) of Spectral with representative techniques for learning from unlabeled data, described below.

Expectation Maximization (EM) is an iterative algorithm for model parameter estimation (Dempster et al., 1977). Starting from some initial model parameters, the *E-step* estimates the expectation of the hidden class variables. Then, the *M-step* recomputes the model parameters so that the likelihood is maximized, and the process repeats. EM is guaranteed to converge to some *local maximum*. It is very popular and useful, but also known to be sensitive to the initialization of parameters.

The *co-training* paradigm proposed by Blum and Mitchell (1998) involves two classifiers employing two distinct views of the feature space, e.g., 'textual content' and 'hyperlink' of web documents. The two classifiers are first trained with labeled data. Each of the classifiers adds to the labeled data pool the examples whose labels are predicted with the highest confidence. The classifiers are trained with the new augmented labeled data, and the process repeats. Its theoretical foundations are based on the assumptions that two views are redundantly sufficient and conditionally independent given classes. Abney (2002) presents an analysis to relax the (fairly strong) conditional independence assumption to *weak rule dependence*.

Nigam and Ghani (2000) study the effectiveness of co-training through experiments on the text categorization task. Pierce and Cardie (2001) investigate the scalability of co-training on the base noun phrase bracketing task, which typically requires a larger number of labeled examples than text categorization. They propose to manually correct labels to counteract the degradation of automatically assigned labels on large data sets. We use these two empirical studies as references for the implementation of co-training in our experiments.

Co-EM (Nigam and Ghani, 2000) combines the essence of co-training and EM in an elegant way. Classifier A is initially trained with the labeled data, and computes probabilistically-weighted labels for all the unlabeled data (as in E-step). Then classifier B is trained with the labeled data plus the probabilistic labels computed by classifier A. It computes probabilistic labels for A, and the process repeats. Co-EM differs from co-training in that *all* the unlabeled data points are re-assigned probabilistic labels in every iteration. In Nigam and Ghani (2000)'s experiments, co-EM outperformed EM, and rivaled co-training. Based on the results, they argued for the benefit of exploiting distinct views.

4.4 Discussion

We observe two major differences between spectral analysis and the above techniques for learning from unlabeled data.

Feature prediction (Spectral) vs. label prediction First, the learning processes of the above techniques are driven by the prediction of class labels on the unlabeled data. As their iterations proceed, for instance, the estimations of class-related probabilities such as $\mathrm{P}(c)$, $\mathrm{P}(f|c)$ may be improved. On the other hand, a spectral vector can be regarded as an approximation of $\vec{p}_w$ (a vector of $\mathrm{P}(f_i|w)$) when the dependency $\vec{\delta}_w$ is sufficiently small. In that sense, *spectral analysis predicts unseen feature occurrences* which might be observed with word w if w had more occurrences in the corpus.

Global optimization (Spectral) vs. local optimization Secondly, starting from the status initialized by labeled data, EM performs local maximization, and co-training and other bootstrapping methods proceed greedily. Consequently, they are sensitive to the given labeled data. In contrast, spectral analysis performs global optimization (eigenvector computation) independently from the labeled data. Whether or not the performed global optimization is meaningful for classification depends on the 'usefulness' of the given feature extractors. We say features are useful if dependency and feature mingling (defined in the Appendix) are small.

It is interesting to see how these differences affect the performance on the word classification task. We will report experimental results in the next section.

5 Experiments

We study Spectral's performance in comparison with the algorithms discussed in the previous sections.

5.1 Baseline algorithms

We use the following algorithms as baseline: EM, co-training, and co-EM, as established techniques for learning from unlabeled data in general; the bootstrapping method proposed by Thelen and Riloff (2002) (hereafter, TRB and TR) as a state-of-the-art bootstrapping method designed for semantic lexicon construction.

5.1.1 Implementation of EM, co-training, and co-EM

Naive Bayes classifier To instantiate EM, co-training, and co-EM, we use a standard Naive Bayes classifier, as it is often used for co-training experiments, e.g., (Nigam and Ghani, 2000; Pierce and Cardie, 2001). As in Nigam and Ghani (2000)'s experiments, we estimate $\mathrm{P}(f|c)$ with Laplace smoothing, and for label prediction, we compute for every $c \in C$:

$$\mathrm{P}(c|w) \propto \tilde{\mathrm{p}}(c) \prod_{\#(f',w)>0} \tilde{\mathrm{p}}(f'|c)^{\#(f',w)} .$$

The underlying naive Bayes assumption is that occurrences of features are conditionally independent of each other, given class labels. The generative interpretation in this case is analogous to that of text categorization, when we regard features (or contexts) of all the occurrences of word w as a pseudo document.

We initialize model parameters ($\tilde{\mathrm{p}}(c)$ and $\tilde{\mathrm{p}}(f|c)$) using labeled examples. The test data is labeled after m iterations. We explore $m = 1, \cdots, 10$ for EM and co-EM, and $m = 1, \cdots, 100$ for co-training[7]. Analogous to the choice of input vectors for spectral analysis, we hypothesize that using all the unlabeled data for EM and co-EM may rather degrade performance. We feed EM and co-EM with the k most frequent unlabeled words[8]. As for co-training, we let each of the classifiers predict labels of all the unlabeled data, and choose 50 words labeled with the highest confidence[9].

Co-training and co-EM require two redundantly sufficient and conditionally independent views of features. We split features randomly, as in one of the settings in Nigam and Ghani (2000). We also tested left context vs. right context (not reported in this paper), and found that random split performs slightly better.

To study the potential best performance of the baseline methods, we explore the parameters described above and report the *best results*.

5.2 Implementation of Spectral

In principle, spectral vectors can be used with any linear classifier. In our experiments, we use a standard centroid-based classifier using cosine as the similarity measure. For comparison, we also test count vectors (with and without tf-idf weighting) with the same centroid-based classifier.

Spectral has two parameters: the number of input vectors k, and the subspace dimensionality h. We set $k = 1000$ and $h = 30$ based on the observation on a corpus disjoint from the test corpora, and use these settings for all the experiments.

[7]The maximum numbers of iterations were chosen so that the best performance of the baseline algorithms can be observed.

[8]Indeed, it turned out that setting k to an appropriate value (2500 on the particular data described below) produces significantly better results than using all the unlabeled data.

[9]We also tested Pierce and Cardie (2001)'s modification to choose examples according to the label distribution, but it did not make any significant difference.

	Spectral	TRB	co-TR	co-EM	EM	NB	Tf-idf	Count
100 seeds	*60.2*	51.7	50.4	49.4	50.7	43.3	40.7	32.6
300 seeds	*62.9*	47.1	57.8	55.8	53.2	50.8	46.7	35.2
500 seeds	*63.8*	42.7	56.7	56.6	54.3	53.2	49.9	36.0
Exploiting unlabeled data?	Yes					No		
Classification model	Centroid	-	Naive Bayes				Centroid	

Figure 1: F-measure results (%) on high frequency seeds. AP-corpus.
Cf. Naive Bayes classifiers (NB) trained with *7500 seeds* produce *62.9%* on average over five runs with random training/test splits.

5.3 Target Classes and Data

Following previous semantic lexicon studies, we evaluate on the classification of lemma-form nouns. As noted by several authors, accurate evaluation on a large number of proper nouns (without context) is extremely hard since the judgment requires real-world knowledge. We choose to focus on non-proper head nouns. To generate the training/test data, we extracted all the non-proper nouns which appeared at least twice as the head word of a noun phrase in the AP newswire articles (25K documents), using a statistical syntactic chunker and a lemmatizer. This resulted in approx. 10K words. These 10K words were manually annotated with six classes: five target classes – persons, organizations, geo-political entities (GPE), locational entities, and facilities —, and 'others'. The assumed distribution (Ω) was that of general newspaper articles. The definitions of the classes follow the annotation guidelines for ACE (Automatic Content Extraction)[10]. Our motivation for choosing these classes is the availability of such independent guidelines. The breakdown of the 10K words is as follows.

Per.	1347	13.8%	Fac.	238	2.4%
Loc.	145	1.5%	GPE	17	0.2%
Org.	136	1.4%	Others	7871	80.7%

The majority (80.7%) are labeled as Others. The most populous target class is Person (13.8%). The reason for GPE's small population is that geo-political entities are typically referred to by their names or pronouns rather than common nominal. We measure precision ($\#(\text{target-class match})/\#(\text{proposed as target-class})$) and recall ($\#(\text{target-class match})/\#(\text{target-class members})$), and combine them into the F-measure with equal weight. The chance performance is extremely low since target classes are very sparse. Random choice would result in F-measure=6.3%. Always proposing Person would produce F=23.1%.

5.4 Features

Types of feature extractors used in our experiments are essentially the same as those used in TR's experiments, which exploit the syntactic constructions such as subject-verb, verb-object, NP-pp-NP (pp is preposition), and subject-verb-object. In addition, we exploit syntactic constructions shown to be useful by other studies — lists and conjunctions (Roark and Charniak, 1998), and adjacent words (Riloff and Shepherd, 1997).

We count feature occurrences ($\#(f_i, w)$) in the unannotated corpus. All the tested methods are given exactly the same data points.

5.5 High-frequency seed experiments

Prior semantic lexicon studies (e.g., TR) note that the choice of seeds is critical – i.e., seeds should be high-frequency words so that methods are provided with plenty of feature information to bootstrap with. In practice, this can be achieved by first extracting the most frequent words from the target corpus and manually labeling them for use as seeds.

To simulate this practical situation, we split the above 10K words into a labeled set and an unlabeled set[11], by choosing the s most frequent words as the labeled set, where $s = 100, 300$, and 500. Note that approximately 80% of the seeds are negative examples ('Others'). As we assume that test data is known at the time of training, we use the unlabeled set as both unlabeled data and test data.

5.5.1 AP-corpus high-frequency seed results

Overall F-measure results on the AP corpus are shown in Figure 1. The columns of the figure are roughly sorted in the descending order of performance. Spectral significantly outperforms the others. The algorithms that exploit unlabeled data outperform those which do not. Tf-idf and Count perform poorly on this task. Although TRB's performance was better on a smaller number of seeds in this particular setting, it showed different trends in other settings.

Spectral trained with 300 or 500 labeled examples (and 1000 unlabeled examples via spectral analysis) rivals Naive Bayes classifiers trained with 7500 labeled examples (which produce 62.9% on average over five runs

[10]http://www.nist.gov/speech/index.htm

[11]The labels of the 'unlabeled set' are hidden from the methods.

	Spectral	baseline ceiling	
100 seeds	*59.2*	52.3	(co-EM)
300 seeds	*62.3*	55.8	(co-TR)
500 seeds	*61.4*	56.6	(co-TR)

Figure 2: F-measure results (%) on high frequency seeds. WSJ-corpus. Results of Spectral and the best-performing baseline are shown.
Cf. Naive Bayes classifiers *trained with 7500 seeds* achieve *62.6%* on average over five runs with random training/test splits.

	Spectral	baseline ceiling	
100 seeds	*61.3* (+1.1)	38.9 (-11.5)	co-TR
300 seeds	*64.5* (+1.6)	47.9 (-9.9)	co-TR
500 seeds	*64.9* (+1.1)	53.4 (-3.2)	co-EM

Figure 3: Results on randomly chosen seeds. AP-corpus. Average over five runs with different seeds. Numbers in parentheses are 'random-seed performance' minus 'high-frequency-seed performance' (in Figure 1).

with random training/test splits).

Also note that the reported numbers for TRB, co-training, co-EM, and EM are the best performance among the explored parameter settings (described in Section 5.1.1), whereas Spectral's parameters were determined on a corpus disjoint from the test corpora once and used for all the experiments (Section 5.2).

5.5.2 WSJ-corpus high-frequency seed results

Figure 2 shows the results of Spectral and the best-performing baseline algorithms when features are extracted from a different corpus (Wall Street Journal 36K documents). We use the same 10K words as the labeled/unlabeled word set while discarding 501 words which do not occur in this corpus. Spectral outperforms the others. Furthermore, Spectral trained with 300 or 500 seeds rivals Naive Bayes classifiers trained with 7500 seeds on this corpus (which achieve 62.6% on average over five runs with random training/test splits).

5.6 Random-seed experiments

To study performance dependency on the choice of seeds, we made labeled/unlabeled splits randomly. Figure 3 shows results of Spectral and the best-performing baseline algorithms. The average results over five runs using different seeds are shown.

All the methods (except Spectral) exhibit the same tendency. That is, performance on random seeds is lower than that on high-frequency seeds, and the degradation is larger when the number of seeds is small. This is not surprising since a small number of randomly chosen seeds provide much less information (corpus statistics) than high frequency seeds. However, Spectral's performance does not degrade on randomly chosen seeds. We presume that this is because it learns from unlabeled data independently from seeds.

	High (Spectral)	Medium	Low	All
100 seeds	*61.3*	47.8	30.6	48.1
300 seeds	*64.5*	57.6	37.9	53.9
500 seeds	*64.9*	57.0	37.9	54.3

Figure 4: F-measure results (%) in relation to the choice of input vectors for spectral analysis. AP-corpus. Random seeds. Using count vectors from high-, medium, low-frequency words (1000 each), and all the 10K words.

5.7 Choice of input vectors for spectral analysis

Recall that our basic idea is to use vectors with small estimation errors to achieve better subspace approximation. This idea led to applying spectral analysis to the most frequent words. We confirm the effectiveness of this strategy in Figure 4. 'Medium' and 'Low' in the figure compute the subspaces from 1000 words with medium frequency (68 to 197) and with low frequency (2 on average), respectively. Clearly, standard Spectral ('High': computing subspace from the most frequent 1000 words; frequency $\geq$ 198) outperforms the others. When all the vectors are used (as LSI does), performance degrades to below Medium. 'Low' gains almost no benefits from spectral analysis. The results are in line with our prediction.

6 Conclusion

We show that spectral analysis is useful for overcoming data sparseness on the task of classifying words into their entity classes. In a series of experiments, the proposed method compares favorably with a number of methods that employ techniques such as EM and co-training.

We formalize the notion of harmful portions of the commonly used feature vectors for linear classifiers, and seek to factor out them via spectral analysis of unlabeled data. This process does not use any class information. By contrast, the process of bootstrapping is generally driven by class label prediction. As future work, we are interested in combining these somewhat orthogonal approaches.

Acknowledgements

I would like to thank colleagues at IBM Research for helpful discussions, and anonymous reviewers for useful comments. This work was supported by the Advanced Research and Development Activity under the Novel Intelligence and Massive Data (NIMD) program PNWD-SW-6059.

References

Steven Abney. 2002. Bootstrapping. In *Proceedings of ACL'02*.

Rie Kubota Ando and Lillian Lee. 2001. Iterative Residual Rescaling: An analysis and generalization of LSI. In *Proceedings of SIGIR'01*, pages 154–162.

Yossi Azar, Amos Fiat, Anna Karlin, Frank McSherry, and Jared Saia. 2001. Spectral analysis of data. In *Proceedings of STOC 2001*.

Matthew Berland and Eugene Charniak. 1999. Finding parts in very large corpora. In *Proceedings of ACL'99*.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unalbeled data with co-training. In *Proceedings of COLT-98*.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of EMNLP/VLC'99*.

Chandler Davis and W. M. Kahan. 1970. The rotation of eigenvectors by a perturbation. III. *SIAM Journal on Numerical Analysis*, 7(1):1–46, March.

Scott Deerwester, Susan T. Dumais, Geroge W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the Society for Information Science*, 41:391–407.

A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38.

Gene H. Golub and Charles F. Van Loan. 1996. Matrix computations third edition.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato's problem. *Psychological Review*, 104:211–240.

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of Information and Knowledge Management*.

Christos H. Papadimitriou, Prabhakar Raghavan, Hisao Tamaki, and Santosh Vempala. 2000. Latent Semantic Indexing: A probabilistic analysis. *Journal of Computer and System Sciences*, 61(2):217–235.

William Phillips and Ellen Riloff. 2002. Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In *Proceedings of EMNLP'02*.

David Pierce and Claire Cardie. 2001. Limitations of co-training for natural language learning from large datasets. In *Proceedings of EMNLP'01*.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*.

Ellen Riloff and Jessica Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of EMNLP'97*.

Brian Roark and Eugene Charniak. 1998. Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In *Proceedings of COLING-ACL'98*.

Hinrich Schüetze. 1992. Dimensions of meaning. In *Proceedings of Supercomputing'92*, pages 787–796.

Michael Thelen and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extracting pattern contexts. In *Proceedings of EMNLP'02*.

Vladimir Vapnik. 1998. *Statistical Learning Theory*. Wiley Interscience, New York.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of ACL'95*, pages 189–196.

Appendix

Estimation error, dependency, and feature mingling

Using the notation in Section 3.1, for simplicity, assume that all the seeds and words are non-polysemous. Suppose that label prediction is done by choosing the most similar seed where similarity is measured by inner products of corresponding count vectors. Set $\mu_{\mathcal{F}} = ||\mathbf{M}||$ where $\mathbf{M}$ is a matrix whose $[i, j]$-element is $\vec{p}_{c_i}^T \vec{p}_{c_i} - 1$ if $i = j$; $\vec{p}_{c_i}^T \vec{p}_{c_j}$ otherwise. Intuitively, $\mu_{\mathcal{F}}$ quantifies 'feature mingling'; it is larger when feature distributions over classes are uniform (i.e., useless for label prediction). Let S be a set of given seeds. Set

$$\epsilon = ||\vec{\varepsilon}_w + \vec{\delta}_w|| + \max_{s \in S}(||\vec{\varepsilon}_s + \delta_s||)(1 + ||\vec{\varepsilon}_w + \vec{\delta}_w||) \ .$$

Using properties of the matrix norm, it is easy to show that *for arbitrary $w \in W$, if*

$$1/2 \geq \epsilon + \mu_{\mathcal{F}}/\sqrt{2} \ ,$$

then, w's label is predicted correctly.

Since the condition is sufficient but not necessary, the proportion of the words that satisfy this condition gives the lower bound of the label prediction accuracy.

Background: spectral analysis

Singular value decomposition (SVD) factors a matrix into the product: $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, such that $\mathbf{U}$ and $\mathbf{V}$ are orthonormal and $\mathbf{\Sigma}$ is diagonal. Columns of $\mathbf{U}$ are called *left singular vectors*, and diagonal entries of $\mathbf{\Sigma}$ are called *singular values*. Also note that left singular vectors of $\mathbf{A}$ are eigenvectors of $\mathbf{A}\mathbf{A}^T$. Let $\mathcal{X}_h$ be the subspace spanned by left singular vectors corresponding to the h largest singular values of matrix $\mathbf{A}$. In this paper, we call this process of computing $\mathcal{X}_h$ *spectral analysis*. Among all possible h-dimensional subspaces, $\mathcal{X}_h$ is the subspace that maximizes orthogonal projections of $\mathbf{A}$'s column vectors in terms of the sum of squares of vector lengths. In that sense, we say that *spectral analysis captures the most prominent vector directions*. More details are found in e.g., (Golub and Loan, 1996).