Part of Speech Induction using Non-negative Matrix Factorization

Sravana Reddy

May 2009

Abstract

Unsupervised part-of-speech induction involves the discovery of syntactic categories in a text, given no additional information other than the text itself. One requirement of an induction system is the ability to handle multiple categories for each word, in order to deal with word sense ambiguity. We construct an algorithm for unsupervised part-of-speech induction, treating the problem as one of soft clustering. The key technical component of the algorithm is the application of the recently developed technique of non-negative matrix factorization to the task of category discovery, using word contexts and morphology as syntactic cues.
## Contents

Abstract 1

1 Introduction 5  
1.1 What is Part-of-speech Induction? 5  
1.2 Ambiguity in the Lexicon 6  
1.3 Our Methods and Results 7  

2 Background 8  
2.1 Motivation for Part-of-Speech Tagging 8  
2.2 Existing Work 8  
2.2.1 Hard Clustering 9  
2.2.2 Dealing with Ambiguity 12  
2.2.3 Category Induction with Morphology 13  
2.3 Evaluation Methods 14  
2.3.1 Gold Standard Evaluation 14  
2.3.2 Model-based Evaluation 17  

3 Category Induction with NMF 18  
3.1 Preliminaries on Non-negative Matrix Factorization 18  
3.2 Part-of-Speech Induction Using NMF 20  
3.2.1 Outline of Algorithm 20  
3.2.2 Step 1a: First run of NMF 23  
3.2.3 Step 1b: Low Frequency Words 23  
3.2.4 Part-of-Speech Tagging 24  

4 Results 25  
4.1 The Setup 25  
4.1.1 Data 25  
4.1.2 Evaluation 26  
4.1.3 Baseline Algorithms 26
List of Tables

4.1 Results: high-frequency word types, different sparseness and feature spaces (WSJ) . 27
4.2 Results: high-frequency word types, different sparseness and feature spaces (Brown) 28
4.3 Results: all word types, over different feature spaces, and constant sparseness (WSJ and Brown) ................................................................. 29
4.4 Baseline comparison: word types after NMF and SVDClust (WSJ) .................. 30
4.5 Baseline comparison: word types after NMF and SVDClust (Brown) ............ 30
4.6 Results: all word tokens, over different feature spaces ......................... 32
4.7 Baseline comparison: word tokens after NMF, SVDClust, and HMM (WSJ) .... 32
4.8 Baseline comparison: word tokens after NMF, SVDClust, and HMM (Brown) .. 33

5.1 Sample of WSJ clusters after SVDClust, step 1b .......................... 34
5.2 Sample of WSJ clusters after NMF, step 1b .................................. 35
5.3 Detection of ambiguous words (WSJ) ........................................ 35
5.4 Distribution after POS induction of common ambiguous words ................. 36
5.5 Overview of some ‘good’ clusters induced from the WSJ, part 1 ............. 38
5.6 Overview of some ‘good’ clusters induced from the WSJ, part 2 ............. 39
5.7 Overview of some ‘bad’ clusters induced from the WSJ ....................... 40
Chapter 1

Introduction

1.1 What is Part-of-speech Induction?

The main motivation for part-of-speech induction comes from the problem of part-of-speech tagging, which involves labeling each word in a given text with its correct part-of-speech category. Corpora annotated with part-of-speech categories are essential for a variety of language processing applications, including algorithms for learning language structure, such as morphological analyzers or parsers, as well as higher-level tasks like information retrieval, machine translation, and speech recognition.

Procedures for part-of-speech tagging fall into three broad classes. *Supervised tagging* uses training information; namely, data labeled with syntactic categories. In other cases, only a limited amount of labeled data (or none at all) is available; however, a lexicon is given which specifies the possible categories that can be assigned to each word in the text. This is known as *partly-supervised tagging*.

There are several situations where a lexicon may not be available, or the one available is not necessarily appropriate. For instance, lexicons may not exist for low-resource languages – languages lacking substantial corpora with linguistic annotations. Further, there is no universally correct syntactic classification system or tagset for any given language: the traditional European grammatical system posits 8 parts of speech in English, the Penn Treebank includes 45, and the CLAWS1, 132. This discrepancy is inevitable – some applications require more specific and fine-grained distinctions among categories than others. Existing tagsets may hence not suffice for a particular task or language. This necessitates *inducing syntactic categories*\(^1\) from an unannotated corpus. Part-of-speech tagging using no labeled data or a lexicon – in general, *no external information* besides the text itself – is known as *unsupervised tagging*.\(^2\)

---

\(^1\)In this paper, we use the terms ‘part-of-speech’, ‘word category’, and ‘syntactic category’ interchangeably.

\(^2\)There is no consensus in the literature on these terms – very often, ‘unsupervised tagging’ is taken to mean the case when there is no labeled data, but there is a lexicon listing all the valid word-category assignments. We follow the terminology given by Johnson (2007).
Unsupervised tagging can be thought of as involving two steps:

1. **(unsupervised) part-of-speech induction**: This is the process of bootstrapping the lexicon – that is, learning the syntactic categories of the language (which are not known \(a\ p\ i\ r\ i\ o\)) as well as assigning categories to each word type in the data.

2. **tagging**: Using the above lexicon to tag the data. This can be accomplished by a standard partly-supervised tagging algorithm. The non-trivial problem here is essentially one of disambiguation – discovering which of the valid categories (as given by the lexicon) should be assigned to each occurrence of a word.

This thesis focuses on the first step – namely, **unsupervised part-of-speech induction**.

### 1.2 Ambiguity in the Lexicon

While several algorithms exist for unsupervised part-of-speech tagging, few tackle the problem of ambiguity – the assignment of words to multiple categories. Schütze (1995) skips forward to assigning syntactic categories to word *tokens* (individual occurrences of words) in the text, without first creating a lexicon mapping word types to categories. Such an approach runs the risk of overfitting, or of having too little information for generalization to unseen data. In particular, the resulting model may not be able to predict the category of a token that occurs in a context that is highly dissimilar to any of the data seen during training. Further, one gains flexibility and modularity by treating part-of-speech tagging as a two-step procedure, where the word types are first assigned probabilistic memberships to categories (i.e, *part-of-speech induction*), and the resulting lexicon then used in an independent partly-supervised tagging algorithm.

One way of dealing with ambiguity at the type-level is to use a hidden Markov model on raw text (Johnson, 2007; Goldwater and Griffiths, 2007; Freitag, 2004). However, HMMs have several limitations: it is non-trivial to add other syntactic cues (such as long-range contextual information or morphology) as features, and training is expensive and does not scale well to large numbers of categories.

The other option is to use distributional features to cluster word types, where each word is a member of one or more clusters. This approach has been used by Clark (2000); however, only words which display an extremely high degree of ambiguity are assigned to more than one cluster, with low-frequency words being presumed to belong to exactly one category, which is too simplifying an assumption. For example, in the (shortened) Wall Street Journal corpus that we use later, 142 of the 831 words that occur exactly 3 times belong to multiple categories.

Our method aims to find soft clusters with no assumptions on which words can belong to more than one cluster. Further, no previous work has been done with effective results on type-level category induction using *vector space* models. Given the flexibility of such models in representing information, it is desirable to explore soft clustering approaches in this framework.
1.3 Our Methods and Results

The key contribution of this paper is in detailing a procedure for part-of-speech induction that allows for ambiguity in category assignments. In particular, our method discovers clusters of word types from the data, such that every word has a probabilistic membership to multiple clusters. This is achieved by applying the technique of non-negative matrix factorization to a vector space representation of the input text.

Non-negative Matrix Factorization (NMF) is a rank-reduction method that lends itself naturally to soft clustering in a feature space. The reduced space is not necessarily orthogonal, and all points in the space have non-negative values in each dimension. If each dimension is taken to correspond to a cluster, the membership probability of a point in a cluster is proportional to its value in the corresponding dimension. While NMF has been applied to a variety of problems, it has generally not been exploited for the problem of soft clustering.

Our results show a significant overall improvement compared to related algorithms. More importantly, it is found that a large part of this improvement is due to improved detection of word ambiguity, leading to the conclusion that NMF is a step in the right direction.
Chapter 2

Background

2.1 Motivation for Part-of-Speech Tagging

Text tagged with part-of-speech categories finds use in:

1. **Dealing with data sparsity**: For example, a speech recognition system could use an \(n\)-gram language model trained over *words*, but doing so would require a huge amount of training data in order to come close to accounting for the range of constructions seen in deployment. A more robust approach is to augment the system with an \(n\)-gram model trained over *word* *categories*.

2. **Word sense and pronunciation disambiguation**: While sense extends beyond part-of-speech, the latter is a major cue in narrowing down the possible senses of a word in context. Several words with ambiguous parts of speech may be pronounced differently – information that is needed by a speech recognizer or synthesizer. One example in English is the set of words whose noun and verb forms are distinguished only by the placement of stress: (\*object/N, object/V), (\*record/N, record/V), (\*combat/N, combat/V).

3. As a first step in **syntactic parsing** or chunking, which are in turn used for a variety of applications like machine translation and semantic parsing. Part-of-speech tagging is also useful in applications like **information retrieval** – finding named entities in raw text generally requires filtering words of certain categories or phrases of a given category-sequence (noun phrases, proper names, etc).

2.2 Existing Work

This section provides an overview of the major work in unsupervised part-of-speech induction and tagging. All major work in this area uses *distributional information* as a cue to syntactic categories.
In particular, this information takes into account context distributions – words that occur adjacent to or near the same words are likely to belong to the same category.

Existing part-of-speech induction algorithms fall under two broad classes:

1. **Hard clustering**: partitioning of word types, so that each word is assigned to exactly one cluster.

2. **Ambiguity-aware induction**: this may either involve
   
   (a) Inducing soft clusters over word types
   
   (b) Clustering word tokens

In addition, an orthogonal category of methods uses morphological information in addition to distributional cues.

### 2.2.1 Hard Clustering

One approach in this area is to use the *mutual information* between adjacent words. The best example of this is the agglomerative clustering algorithm of Brown et al. (1992). The program iteratively merges words into classes in such a way that the loss in average bigram mutual information computed over classes is minimized at each merge. Word classes are hence taken to reflect generalizations of word-level bigrams; the idea being that if each word in the class \( m = \{m_1, m_2, m_3, \ldots, m_q\} \) most often occurs immediately to the right of one or more words in the class \( l = \{l_1, l_2, l_3, \ldots, l_p\} \), then \( m \) and \( l \) are coherent lexical categories. This algorithm is used to cluster the words of the Brown corpus (Francis and Kucera, 1982) into 1000 categories. The categories vary in structure from semantically consistent (days of the week, nationalities, units of measure) to those resembling traditional parts of speech (gerunds, possessives, modals), to irregular, meaningless classes.

Knesser and Ney (1993) develop a similar program that uses a slightly different objective function at each merge, and produce comparable results on English and German data. McMahon and Smith (1996) creates a *top-down hierarchical clustering* of words, by initializing a random hierarchy, and moving words between clusters at each iteration by seeking to maximize the average mutual information between clusters.

Elman (1990) performs a toy experiment using a *neural network*. He picks 29 words categorized into noun and verb classes (NOUN-ANIM, VERB-EAT, NOUN-HUM, etc), and randomly generates sample sentences based on 15 hand-crafted 2-3 word sentence templates (NOUN-ANIM VERB-EAT NOUN-FOOD, NOUN-HUM VERB-INTRAN, etc). A stream of 10000 such sentences (with no breaks between them) is fed to a neural network, one word at a time, whose task is to predict the next word in the stream. The 29 words are then hierarchically clustered using the hidden unit activation vectors produced by inputting them to the trained network. The clustering is shown to
be successful both in correlating with the original classes and learning a natural hierarchy of these classes.

Another, more widely used approach, is to compare words based on the explicit contexts in which they occur. Early work in this area includes Kiss (1973), who used the context to the immediate right, and induced a hierarchical clustering of 31 words in a corpus of transcribed child-directed speech. The clusters were found to correspond well to syntactic classes, although it is difficult to evaluate the work given the small size of the data. Brill et al. (1990) use bigram statistics to compute a score that two words $w_1$ and $w_2$ are in the same category. If $w_2$ can be substituted for $w_1$ in a large percentage of bigrams containing $w_1$, and vice versa, $w_1$ and $w_2$ are taken to belong to the same class. This process is repeated for all word pairs, and the transitive closure of word classes is taken to get the final categories. The algorithm is tested on the Brown corpus, and the description of the results show that syntactic categories such as possessive pronouns, determiners, nouns, and modals are found.

Ney et al. (1994) devise an algorithm that finds the assignment of words to the set of categories $C$ in such a way as to maximize the likelihood of the text, given by:

$$F(T) = \sum_{w \in T} Pr(w) \log Pr(w) + \sum_{(c_0, c_1) \in C} \frac{Pr(c_0, c_1)}{Pr(c_0)Pr(c_1)}$$ (2.1)

where $T$ is the set of types in the data, and $Pr(c_0, c_1)$ is the relatively frequency of a word in the category $c_0$ (according to the current assignment of words to categories) being followed by a word in the category $c_1$. The algorithm starts with a random assignment, and then loops over the words in $T$. For each word, it moves it to the category $c$ such that this assignment $c$ will maximize $F(T)$. The system keeps looping over the words and moving them between categories until the change in $F(T)$ becomes sufficiently small. It is tested on the LOB English corpus (Kuhn and de Mori, 1990). The 50000 words of the corpus are assigned to 120 categories, which are found to correspond more or less to syntactic parts of speech.

A more sophisticated method is to use a vector-space approach. A context vocabulary $F = [f_1, f_2, \ldots, f_n]$ of the $n$ most frequent words in a corpus is extracted, and counts are taken of the number of times each $f_i$ occurs to the left of a word $w$, and the number of times it occurs to the right. The context vector $CV(w)$ is taken to be the normalized vector of counts $[left(f_1, w), left(f_2, w), \ldots, left(f_n, w), right(f_1, w), right(f_2, w), \ldots, right(f_n, w)]$.

A similarity metric between two words can now be defined by computing the similarity between their context vectors. Variations on the context vectors can be achieved by changing the size of the vocabulary $F$ or by increasing the size of the context window. The context window is the range on either side of a word to be counted into the context vector; a window of size $2p$ counts the $p$ words immediately preceding and following the reference word. Context vectors are built for every word in $W$, the set of word types in the text, projecting the words into a feature space of $2p|F|$ dimensions.
Finch and Chater (1994) use a context window of size 4, and hierarchical clustering to discover categories – pairs of words with the most similarity between their context vectors are merged at each step. The resulting clusters resemble those of Brown et al, corresponding to classes showing fine-grained semantic distinctions as well as traditional syntactic classes.

Finding the similarity between context vectors in the $2p|F|$-dim feature space can be highly inefficient, especially when using a large vocabulary $F$. Further, certain generalizations that underly some of the desired word categories may not not apparent. For instance, the contexts to the right of ‘a’ and ‘an’ are disjoint. It is hence unlikely that a clustering algorithm will put them in the same category since the similarity between their context vectors will be fairly low. Schütze (1993, 1995) addresses these issues by applying singular value decomposition (SVD) to the sparse matrix $M_{|W| \times 2k|F|}$, where the context vector of $w_i \in W$ forms the $i^{th}$ row. SVD projects the words of $W$ onto 50 dimensions. The words in the new, reduced-dimension space are then grouped into $k$ classes using $k$-means clustering.

Schütze (1995) refines the above procedure by making a second pass over the data with SVD, using the induced clusters as context features. That is, the context vocabulary referred to earlier is taken to be $C = [c_1, c_2, \ldots c_k]$, the $k$ clusters found in the first pass. The generalized context vector $CV(w)$ is the normalized count vector $[\text{left}(c_1, w), \text{left}(c_2, w), \ldots \text{left}(c_k, w), \text{right}(c_1, w), \text{right}(c_2, w), \ldots, \text{right}(c_n, w)]$ where $\text{left}(c_i, w)$ is the number of times that any word in the cluster $c_i$ occurs to the left of $w$ (and similarly for $\text{right}(c_i, w)$).

Belkin and Goldsmith (2002) take a similar approach in using context vectors (with a window of size 2), but apply spectral decomposition to the data in place of SVD. In particular, a graph of nearest neighbors is built with word types as nodes. The similarity between two words $w_1$ and $w_2$ is computed using the cosine measure

$$\text{Sim}(w_1, w_2) = \frac{CV(w_1) \cdot CV(w_2)}{|CV(w_1)||CV(w_2)|} \quad (2.2)$$

where $CV(w)$ is the context vector associated with $w$. For every word $w$ in the lexicon, the similarity between $w$ and every other word is computed. An edge is then added between $w$ and each $K$ most similar words $w_1, w_2, \ldots w_K$. The laplacian of this graph is then computed. The $N$ lowest-valued eigenvectors of the laplacian constitute the new space, with the words projected onto it in such a way that the more similar a pair of words are to each other (as defined by 2.2), the closer they are. The projected words can now be clustered by their co-ordinates in the new space, using a standard algorithm such as $k$-means.\footnote{For more on laplacian maps, cf. Belkin and Niyogi (2003).}
2.2.2 Dealing with Ambiguity

The methods described so far induce categorizations in which each word type belongs to exactly one category. This is obviously insufficient since most words are syntactically ambiguous (this is especially true in languages like English that don’t have a rich morphology). It is hence necessary to either (1) allow word types to belong to multiple categories, or (2) cluster word tokens (individual occurrences of words).

Soft Clustering of Word Types

Clark (2000) uses an iterative procedure, starting with $K$ one-word clusters over the $K$ most frequent words in the corpus. At each step, the context distributions of the clusters (the weighted average of the distribution of each member word) is computed. The next most frequent (unclustered) word is then assigned to one of the $K$ clusters whose context distribution is closest to its own, provided the similarity between the distributions is above a threshold value. This process is repeated until some proportion of the data is clustered.

In order to account for ambiguity in this algorithm, the context distributions of unclustered words are modeled as being a linear combination of the distributions of the existing clusters. That is, rather than finding the one cluster whose distribution is most similar to that of the word, the focus is on the relative likelihood of belonging to each cluster, thus creating a soft partitioning of words.

A natural way of assigning word types to multiple clusters is by using a hidden Markov model (HMM). The key advantage of HMMs is that they encapsulate a natural framework for ambiguity. Each hidden state may be treated as a part-of-speech category, and the HMM trained over an unannotated corpus. The final trained model assigns a probability distribution over word types for each state, and can be used for POS tagging using the Viterbi algorithm.  

The most widely used procedure for training the parameters of an HMM is the Baum-Welch expectation-maximization (EM) algorithm. However, as Higgins (2002) and Johnson (2007) note, HMMs trained with EM generally produce sub-optimal results for part-of-speech induction. One reason is simply that EM tends to get stuck in local maxima. This problem can be solved by using simulated annealing or experimenting with different starting configurations; however, the tagging results are still found to be poor.

Johnson posits that the main reason for EM’s poor performance is its tendency to create fairly flat distributions of words over states (the number of tokens that each state labels during Viterbi tagging is more or less the same across all states), in contrast to the ‘true’ distributions, which are highly skewed – 6 tags of the Penn Treebank account for more than half the tokens in the corpus.  

\footnote{HMMs are also applied widely in supervised (where labeled data is available for training) as well as partly supervised part-of-speech tagging (where the algorithm has access to some labeled data and/or a lexicon).}  

\footnote{Similarly, if an HMM initialized so that every state has a non-zero probability of transitioning to every other...}
With this in mind, Johnson (2007) and Goldwater and Griffiths (2007) describe a Bayesian method of re-estimating the parameters of an HMM, one of whose advantages is the ability to set priors on the sparsity of the HMM’s emission distributions. Briefly, Bayesian HMMs differ from HMMs with EM in that the multinomial distributions $\theta_y$ (transition probabilities to other states from state $y$) and $\phi_y$ (observation emissions from state $y$) are each conditioned on a prior.

Goldwater and Griffiths (2007) describes a partly supervised as well as unsupervised part-of-speech tagger (they experiment with dictionary knowledge of different degrees, including no knowledge), using Markov Chain Monte Carlo (MCMC) with Gibbs Sampling. Johnson (2007) augments this by testing both MCMC and the Variational Bayes (VB) method for parameter inference for learning an unsupervised HMM tagger.

Other HMM-based unsupervised taggers try to avoid the EM pitfalls by constraining the lexicon. Freitag (2004) first produces a hard clustering of high frequency word types using a method resembling that of Brown et al. (1992). He then initializes an HMM with as many states as clusters, with the emissions of each state being restricted to the words in it corresponding cluster, as well as all the unclustered low frequency words.

### Clustering of Word Tokens

Schütze (1995) explores extensions to his context vector + SVD framework in order to cluster tokens. Context vectors are first created as described earlier. The new vector of a token is then taken to be a concatenation of its type’s context vector, and the context vectors of the preceding and following words. The tokens are then clustered as described in the previous paragraph. A deeper level of information is integrated by clustering the context vocabulary $F$ using the first basic step, and then forming context vectors for the words in $W$ using derived word classes of $F$.

### 2.2.3 Category Induction with Morphology

Distributional statistics rely on the assumption that the text or language in question has rigid word order. For languages and domains for which this assumption must be relaxed, it is necessary to use another cue to part of speech information, like morphology, in place of or in addition to distributional methods.\footnote{Most of the methods described here have not actually been tested on languages other than English (which possesses simple morphology and relatively rigid word order). However, even English can be shown to benefit from incorporating morphological information, and it is assumed that such methods result in even greater improvements for languages with richer morphology.}

\[^4\text{state, EM is unlikely to cause it to converge to the sparser transition matrix that reflects true tag-to-tag transitions (only 683 non-zero tag-to-tag transitions among 37 non-punctuation tags occur in the Penn Treebank). It is fairly straightforward to account for this in EM: simply set the initial HMM topology (the non-zero ‘allowed’ transitions between states) to approximately reflect what is expected of the final model. The topology of an HMM remains more or less unchanged through training.}\]

\[^4\text{Most of the methods described here have not actually been tested on languages other than English (which possesses simple morphology and relatively rigid word order). However, even English can be shown to benefit from incorporating morphological information, and it is assumed that such methods result in even greater improvements for languages with richer morphology.}\]
Clark (2003) uses an HMM (with EM) over characters of a word to capture a notion of morphology. It is combined with the Ney-Essen model discussed in 2.2.1, and the objective function modified to include the probability of the data given the HMM, hence biasing the algorithm to place words with similar substring patterns in the same cluster. Clark also combines the character-level model with his previously described word-level HMM (to incorporate ambiguity), producing a hierarchical HMM (Fine et al., 1998), where the states at the top-level emit HMMs (corresponding to words), whose states then emit characters in order to generate a word.

Higgins (2002) uses a Hidden Neural Network – an HMM where the emission probability distribution of each state is defined by a neural network. The networks in this case are feed-forward perceptrons that predict lexical categories based on morphological features. In addition, the perceptron distributions are combined with raw emission probabilities that prevent overfitting to generalizations based solely on morphology.

2.3 Evaluation Methods

The quality of a model that induces categories and tags text can be evaluated in two general ways:

1. By comparison of the tags with an existing gold standard, or
2. Evaluating the probability that the model assigns to the data in conjunction with the complexity of the model.

2.3.1 Gold Standard Evaluation

Hard Clustering

We first review evaluation metrics for hard clustering, which is the case when dealing with token-level category induction (since every token is labeled with exactly one category, both by the algorithm and the gold standard).

Raw text that is tagged with induced categories/clusters can be compared with a gold standard file containing words tagged with part of speech classes. The key shortcoming of this method is that the resulting clusters do not have labels. One way of getting around the problem is to examine the induced clusters and hand-label each cluster with the gold standard tag that it most resembles. Once the clusters are labeled with the gold standard tags, evaluation is simply a matter of computing precision and recall for each tag. To avoid human intervention, the process of labeling clusters can be automated by matching tags to clusters based on how many words they have in common. The two simplest\(^5\) approaches are:

\(^5\)An algorithm for automatically generating labels based on certain features of the clusters is explained in Schone and Jurafsky (2001), but it has not been evaluated extensively, nor does it appear to scale to languages other than English.
1. Label a cluster with the tag with which it has the highest correspondence (measured by looking at the proportion of words the cluster and the tag share in common). This may result in one part-of-speech tag being used to label several clusters. Johnson (2007) refers to the gold standard evaluation metric using this labeling procedure as the many-to-one accuracy of POS tagging.

2. Johnson notes that the above can easily be undermined by a model that assigns each word to its own class. This would result in perfect many-to-one accuracy. An alternative, proposed by Haghighi and Klein (2006) is to map tags to clusters (as before), but constrain the mapping to be one-to-one by allowing a tag to label at most one cluster. The mapping is done greedily until either the clusters or the tags are exhausted. Johnson refers to the resulting metric as one-to-one accuracy.

A measure related to many-to-one, but not involving precision/recall is to simply count the number of items that the cluster and its matching tag have in common, and take the sum over all clusters, normalized by the total number of items. This is known as purity (Manning et al., 2008):

\[
Purity(G, C) = \frac{1}{\# \text{items}} \sum_{c \in C} \max_{g \in G} |c \cap g|
\]

where \(G\) and \(C\) are the sets of gold standard tags and induced clusters respectively, and \(|c \cap g|\) is the number of items that occur in both the cluster \(c\) and the gold standard tag \(g\). This measure has the same drawback as many-to-one – namely, that assigning each item to its own class would result in a perfect score.

Even the one-to-one method is not ideal, in that it presupposes that the generated clusters are of the same structure and quality as the gold standard tags.\(^6\) This is hardly ever the case – some clusters are more or less granular than traditional tags (for instance, many algorithms separate out names of days and names of places into separate categories, but assign particles, conjunctions, ‘to’, ‘in’, etc which are distinct Treebank classes) to the same category.

An obvious alternative is to measure the information-theoretic correspondence between clusters and tags. One approach is to simply compute the mutual information\(^7\) between the two based on the number of shared items (Ibid.) –

\(^6\)The other shortcoming is that the greedy assignment does not guarantee the best matching between \(G\) and \(C\). Consider a clustering \(C = \{ c_1 = \{1, 2, 5\}, c_2 = \{3, 4\}\}\), with \(G\) being \(\{g_1 = \{1, 2, 3, 4\}, g_2 = \{5\}\}\). The greedy approach would match \(c_1\) with \(g_1\) and \(c_2\) with \(g_2\), whereas the matching with highest overall precision and recall is \(c_1\) with \(g_2\) and \(c_2\) with \(g_1\).

\(^7\)A classic and more well-known measure is that of Rand (1971) which takes counts of the relative distributions of pairs of items in the induced and gold standard clusters, giving the ‘Rand index’, which is very similar to the mutual information metrics shown here. However, to our knowledge, the Rand index has not been used to evaluate part of speech induction.
where $N$ is the total number of items in the data. A higher value of mutual information indicates a better correspondence between the gold standard and the induced categorization.

A related alternative is normalized mutual information (Ibid.):

$$NMI(G; C) = \frac{2 \cdot I(G; C)}{H(G) \cdot H(C)}$$

Clark (2003) measures the conditional entropy of the gold standard tags given the clusters –

$$H(G|C) = H(G) - I(G; C)$$

where

$$H(G) = -\sum_{g} \frac{|g|}{N} \log \frac{|g|}{N}$$

The metric of Eq. 2.6 brings us back to the same trap of the many-to-one accuracy method: assigning each word to its own class will optimize $H(G|C)$. Similarly, $H(C|G) = H(C) - I(G; C)$ can be optimized by assigning most words to the same class. Goldwater and Griffiths (2007) propose a balance between the two metrics by using the variation of information (VI) metric (Meila, 2003):

$$VI(G, C) = H(G|C) + H(C|G)$$

VI effectively measures the amount of information lost when the data is moved from the gold standard clustering $G$ to the obtained clustering $C$. Hence, a lower VI score indicates closer correspondence between the induced and true clusters.

Fuzzy Clustering

It is necessary to evaluate the intermediate task of category induction: the assignment of word types to categories, which is a soft clustering task when allowing for ambiguity. The above mutual information measures can be adapted to problems of probabilistic membership, simply by replacing the integral counts in the mutual information equation with soft counts. Eq. 2.4 hence becomes

$$I(G; C) = \sum_{c \in C} \sum_{g \in G} \sum_{w \in W} |w| \cdot Pr(w \in g) \cdot Pr(w \in c) \log \frac{N \cdot |c \cap g|}{|c| \cdot |g|}$$

where $W$ is the set of all word types, and $|w|$ the frequency of the type $w$ in the data. $Pr(w \in g)$ and $Pr(w \in c)$ denote the membership weight of $w$ in $g$ and $c$ respectively. Variation of information
can be extended as an evaluation measure to soft clustering using the above equation for mutual information.

2.3.2 Model-based Evaluation

Often, we are interested not in how well the induced categories match a gold standard, but in how useful the information provided by the categories is to certain natural language processing tasks or tools. The simplest of these, and one with the most wide-ranging applicability, is the \textit{n-gram language model}. A model that assigns a high probability to a string of words $S$ is deemed a better model than one which assigns a lower probability to the same string. To quantify this intuition, the entropy of a string $S$ is assigned by the model $M$ is first computed –

$$H_M(S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \log Pr(S_i|S_{i-1}S_{i-2} \ldots S_{i-n+1}, M)$$ (2.10)

The strength of a model $M$ is given by its \textit{perplexity} with respect to $S$:

$$\text{Perplexity}(M, S) = 2^{H_M(S)}$$ (2.11)

To evaluate the assignment of categories, Brown et al. (1992) compare the perplexity assigned by a word-level linear interpolated trigram model\footnote{An interpolated n-gram model (Jelinek and Mercer, 1980) is a linear combination of n-gram, (n-1)-gram, (n-2)-gram, ... etc models, to prevent overfitting or to guard against the case of insufficient data. An interpolated trigram model $\tilde{Pr}(w_i|w_{i-1}w_{i-2})$ is used in the paper of Brown et al.: $\tilde{Pr}(w_i|w_{i-1}w_{i-2}) = \lambda_2 \cdot Pr(w_i|w_{i-1}w_{i-2}) + \lambda_1 \cdot Pr(w_i|w_{i-1}) + \lambda_0 \cdot Pr(w_i)$. The weights, $\lambda_2, \lambda_1, \lambda_0$ are estimated using expectation maximization on some data held-out from the text.} to a sample of text with the perplexity of a category-level interpolated trigram model with respect to the same text. That is, each word in the text $S_1S_2 \ldots S_{|S|}$ is replaced by part of speech categories, giving $C_{S_1}C_{S_2} \ldots C_{S_{|S|}}$, and the perplexity $\text{Perplexity}(MC, S)$ is computed:

$$\text{Perplexity}(MC, S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \log Pr(C_{S_i}|C_{S_{i-1}}C_{S_{i-2}} \ldots C_{S_{i-n+1}}, MC)$$ (2.12)

The difference between $\text{Perplexity}(MC, S)$ and $\text{Perplexity}(M, S)$ – in other words, the \textit{gain in perplexity} from using the assigned part of speech tags in place of words – is taken to be a measure of the quality of tagging.
Chapter 3

Category Induction with NMF

Our main motivation in testing non-negative matrix factorization for the part-of-speech induction problem follows from the results of singular value decomposition and related techniques that are applied to this problem. We observe one drawback to such methods based on principal components analysis: they implicitly assume a more or less hard partitioning of the data, making it difficult to recover any underlying soft clusterings. In particular, the space is distorted so that words that might lie at the intersection of two or more clusters are no longer distributed in a way that reflects their ambiguity.

We apply non-negative matrix factorization to our problem since it lends itself naturally to soft clustering of data in a feature space. This chapter first discusses the NMF algorithm, and then describes how we apply it to word category induction.

3.1 Preliminaries on Non-negative Matrix Factorization

Algorithms for non-negative matrix factorization were first developed for use in computer vision (Lee and Seung, 1999). NMF seeks to “approximately factor” a non-negative $m$-by-$n$ matrix $V_{m\times n}$ into the product of two non-negative matrices – an $m$-by-$r$ matrix $W_{m\times r}$ and an $r$-by-$n$ matrix $H_{r\times n}$, for some value of $r < m$. A common application of such methods is to treat the matrix $V$ as a representation of $n$ points in an $m$-dimensional space, with the $i^{th}$ column being the vector associated with the $i^{th}$ point.

The constraint on the factors derived from NMF – namely, that they are non-negative matrices – leads to some interesting properties. The example demonstrated in Lee and Seung (1999) is the task of clustering facial images, where the columns of $V$ are vectors representing a face. In this case, the columns of the matrix $W$ constitute the ‘basis faces’. Every face is a linear combination of the basis faces; the coefficients of this combination for the $i^{th}$ face are given by the $i^{th}$ column vector of $H$. Unlike PCA-based methods, where the coefficients may be negative (and hence

\[4.3.1\] illustrates the failure of SVD to reflect ambiguity in word distribution.
difficult to interpret conceptually), the non-negative coefficients in $H$ derived from NMF allow the interpretation of every 'basis face' as simply an additive component of the images. The authors demonstrate that this is indeed the case; the basis faces resemble variations of facial features.

NMF has subsequently been applied to a range of high-dimensional clustering applications, most notably in term-document matrices (Xu et al., 2003; Shahnaz et al., 2006; Berry and Browne, 2005) for a combination of document clustering and information retrieval. An advantage of the method in these domains is that it allows a document to be represented as a linear additive combination of hidden semantics (as indicated by the term distributions), similar to the parts-based representation of faces.

**An Algorithm for NMF.** Lee and Seung (2001) present an algorithm for NMF which iteratively updates the factors $W$ and $H$ until the product $WH$ is approximately equal to $V$. This approximation can be quantified by defining a convergence condition or cost function that measures the difference between $V$ and $WH$. The authors suggest two cost functions: the squared distance $||V - WH||^2 = \sum_{i,j} (V_{ij} - (WH)_{ij})^2$ and the Kullback-Leibler divergence $D(V||WH)$. We use the squared distance function in our system.\(^2\)

The update steps in the algorithm guarantee that the cost function is non-increasing. The update rule for the squared distance cost function is the multiplicative update which minimizes the squared distance error:

$$H_{ij} \leftarrow H_{ij} \cdot \frac{(W^TV)_{ij}}{(W^TWH)_{ij}}; \quad W_{ij} \leftarrow W_{ij} \cdot \frac{(VH^T)_{ij}}{(WH^TH)_{ij}}$$ (3.1)

Lee and Seung (2001) use a proof analogous to the proof of convergence of expectation-maximization to show that the multiplicative update rule guarantees at least a local minimum for $||V - WH||^2$. The proof is outlined in Appendix A.

**NMF with Sparseness Constraints.** For our purposes, we want to minimize the number of clusters that a word belongs to. Since we determine cluster membership by the coefficients of the corresponding column in $H$, each of the columns of $H$ should be relatively sparse. This is achieved by a procedure proposed by Hoyer (2004), which constrains the sparseness of one or both the matrix factors at each update step. First, a 'sparseness measure' of a vector $v$ of dimensionality $n$ is defined, based on its $L_1$ and $L_2$ norms. This measure evaluates to 1 when $v$ contains exactly one non-zero element, and 0 when all its elements are equal.

$$sparseness(v) = \sqrt{n} - \frac{\sum_{i=1}^{n} |v_i|}{\sqrt{\sum_{i=1}^{n} v_i^2}} \frac{\sqrt{n} - 1}{\sqrt{n} - 1}$$ (3.2)

\(^2\)Preliminary experiments using the Kullback-Leibler divergence indicate that there is almost no difference in clustering results between the two cost functions.
The update step is changed to constrain the columns of $H$ to be sparse, which can brought about by “approximating” each column (after the multiplicative update of 3.1) to the nearest vector that has the desired sparseness value. The approximation in this step is performed via a gradient-descent-type algorithm (Hoyer, 2004). In our case, it is convenient to require the $L_1$ norm of each column of $H$ to be 1, since the coefficients are interpreted to be the probabilistic membership of the word to each of the clusters.\(^3\) We can now set the desired sparseness by choosing a value for the $L_2$ norm.

The algorithm may begin with random positive matrices $W$ and $H$, and iterate the above updates (including the sparseness constraints) until convergence. However, it is vulnerable to getting stuck at a local minimum. To avoid this, we seed the initialization by first performing k-means on the original data (the columns of the matrix $V$), with $k$ set to be the number of desired final clusters, as described by Wild et al. (2004). This is done by setting the columns of $H$ to correspond to the cluster assignments: i.e, the entry $H[i, j]$ is set to be 0 if and only if the $j^{th}$ word is assigned by k-means to the $i^{th}$ cluster.

### 3.2 Part-of-Speech Induction Using NMF

#### 3.2.1 Outline of Algorithm

We use the MATLAB library for NMF with sparseness\(^4\) that implements the algorithm of Hoyer (2004), and modify it to fit our system. In particular, we constrain the columns rather than the rows of $H$ to be sparse, and set the $L_1$ norms to be unity, with the value of the $L_2$ norm being a parameter of the algorithm. The algorithm is seeded with k-means as described earlier, in place of random initialization of the matrix factors.

We represent the word types in the data as a matrix of column feature vectors, analogous to the SVD method of Schütze (1995). NMF is then applied to the matrix, using squared distance as the cost function. Category membership of the $i^{th}$ word is determined by the coefficients in the $i^{th}$ column vector of $H$. We first apply NMF on those word types that occur with a frequency of at least 3 in the corpus, and deal with the low frequency words later by using the category information of the first step, giving the final induced clusters. For tagging purposes, and better comparison to related methods, the induced clusters are used as a lexicon for an HMM trained on the data. A sketch of this procedure is shown, with the following sections elaborating on each step.

**Input:** list of words, typelist, feature set $F$, parameter sparseval, frequency threshold $q$.

**Step 1a**

$\text{highfreq} = \{ w \in \text{typelist} \text{ s.t. } \text{frequency}(w) > q \}$

for $w \in \text{highfreq}$ do

$V_1[:, w] \leftarrow \text{feature vector of } w$, using set $F$ with contexts = word types

---

\(^3\)This is slightly different from Hoyer’s paper, which sets the $L_2$ norm to be unity.

end for

\[ [W_1, H_1] = \text{NMF}(V_1, k, \text{sparseval}) \] \{Run NMF on \( V \), reducing to \( k \) dimensions, with columns of \( H \) being constrained to have sparseness = \text{sparseval}.\}

\( \text{Clust1a} = C_1, C_2, \ldots, C_k \)

for \( w \in \text{highfreq} \) do
  for \( i = 1 \) to \( k \) do
    \( C_i(w) = H[i, w] \) \{\( C_i(w) \) is the probability that \( w \) belongs to \( C_i \).\}
  end for
end for

\[ \text{Step 1b} \]
for \( w \in \text{typelist} \) do
  \( V_2[:, w] \leftarrow \) feature vector of \( w \), using set \( F \) with contexts = word clusters \( \text{Clust1a} \)
end for

\[ [W_2, H_2] = \text{NMF}(V_2, k, \text{sparseval}) \]

\( \text{Clust1b} = C_1, C_2, \ldots, C_k \)

for \( w \in \text{typelist} \) do
  for \( i = 1 \) to \( k \) do
    \( C_i(w) = H_2[i, w] \)
  end for
end for

\[ \text{Step 2} \]
Initialize HMM with states \( X = X_1, X_2, \ldots, X_k \)
for \( i = 1 \) to \( k \) do
  for \( w \in \text{typelist} \) do
    if \( C_i(w) > 0 \) then
      \( \Pr(w|X_i) \leftarrow \) a random nonzero value
    else
      \( \Pr(w|X_i) \leftarrow 0 \)
    end if
  end for
end for
Train HMM with Baum-Welch
Tag using Viterbi

A few different feature sets were experimented with, using some combination of contexts and morphological information. Contexts are simply the words occurring immediately to the left and the right of the reference word (including sentence boundaries). The left context vector of a word \( w \) contains the frequencies with which each context appears to the left of \( w \). A right context vector is similarly defined.

Morphological features are derived from Linguistica (Goldsmith, 2001, 2006), and include the
set of all suffixes, as well as all the signatures resulting from analysis of the data. A signature is a set of stems and a set of suffixes, such that the concatenation of every pair consisting of a stem and a suffix in the signature is a word that is present in the data. For instance, the words [jump, jumping, jumped, jumps, walk, walking, walked, walks] will be analyzed as a single signature with the stems \{walk, jump\} and the suffixes \{null, -ing, -ed, -s\}. Both suffix and signature features are binary.

Morphology is added to compensate for noise and sparsity of information due to free word order. Suffixes are the primary indicators of categories (e.g., most -ing words are gerunds, -ion words are nouns, -ly adverbs, and so on). However, many suffixes are common across categories, like ‘-s’ which occurs on both plural nouns and continuous verbs.

A feature which provides a more fine-grained cue to the grammatical category of a word is the signature that the word belongs to. A signature is defined as a tuple \((T, S)\), where \(T\) is a set of stems and \(S\) a set of suffixes such that the combination of every \(t \in T\) and \(s \in S\) forms a word in the data.

A feature vector of a word \(w\) containing all the contextual and distributional features is of the form

\[
[c(l_1), \ldots, c(l_{|L|}), c(r_1), \ldots, c(r_{|R|}), \ldots, c(t_1), \ldots, c(t_{|T|}), c(s_1), \ldots, c(s_{|S|})]
\]

where \(L, R, S, T\) are the sets of left contexts, right contexts, suffixes, and signatures respectively. \(c(l_x)\) is the number of times that the word \(l_x \in L\) occurred to the left of \(w\), \(c(r_x)\) the number of times that \(r_x \in R\) occurred to the right of \(w\), \(c(t_x) = 1\) if the suffix \(t_x\) is part of \(w\), and \(c(s_x) = 1\) if \(w\) is analyzed as part of the signature \(s_x\) (that is, \(w\) is a combination of a stem and a suffix in \(s(x)\)).

In additional to the above (complete) feature set, denoted by LRST, we also ran experiments on each of the following feature spaces –

1. **LR**: Concatenation of the left and right context vectors
2. **LR\_p**: Concatenation of the left and right context vectors, with the contexts being limited to the \(p\) most common words\(^5\) in the text (for some threshold \(p\)).
3. **LLRR**: Concatenation of the vector that counts contexts that occur two steps to the left of the word, the left and right context vectors as before, and the vector counting contexts that occurs two steps to the right.
4. **LLRR\_p**: Same as the above, except that the contexts are limited to the most frequent words.
5. **LR\_pS**: Concatenation of left and right context vectors (using the most frequent contexts), and suffix features.

\(^5\)Clearly, a word that only occurs a few times does not indicate much information about its neighbor.
6. **LRₚST**: Concatenation of the above with signature features.

We also experimented with different degrees of sparseness in the update step, in the range of 0.7 to 1.0. For a clearer idea of what the parameter indicates, Fig. 3.1 illustrates the changes in the column vector of the word ‘million’ (from the Brown corpus) in the matrix $H$, over sparseness values from 0.05 to 0.75. The 29 values correspond to the membership weight of the word in each of the 29 categories.

Figure 3.1: Projection of a vector to fit different values of sparseness (Eq. 3.2), with $L₁$ norm constrained to be 1.

### 3.2.2 Step 1a: First run of NMF

A matrix containing the feature vectors of high-frequency words is created, for each feature set. NMF is then run on this matrix, with the desired sparseness value. The induced categories form the contexts for the next step of the algorithm.

### 3.2.3 Step 1b: Low Frequency Words

Since there must be sufficiently many contextual cues for them to be both informative and robust against noise, all low frequency words are left out of the main NMF clustering. To process these words, we use the information gained from their more frequent cousins. This step also effectively serves as a second processing pass over the data, refining the quality of results.

For every word type $w$ in the data (both low and high frequency), we first build a context vector over the categories induced from Step 1a. For example, if the feature space is $LR$, the context vector of $w$ holds the counts of the number of times each category contains a word that occurs to the left and right of $w$. (When the context is a low frequency word that was not clustered in Step 1a, it is assigned its own category.) This is similar to the generalized context vectors model of Schütze (1995).
The suffix and signature features are then added to the category-based context vector space, and NMF run to cluster the data in this space. In order to account for noise, the sparseness constraint in the update step in the algorithm for every low frequency word is relaxed slightly. The cluster memberships of the low frequency words are hence relatively less concentrated than those of the other words.

3.2.4 Part-of-Speech Tagging

We also include this step of the process (tagging using the bootstrapped lexicon) in order to better evaluate the induction step. In order to disambiguate the categories to be assigned to individual tokens, the traditional step of partly-supervised (lexicon-aided) part-of-speech tagging is carried out using a first-order hidden Markov model. The states \( \{s_1, s_2, \ldots, s_n\} \) of the model are taken to map to categories \( \{c_1, c_2, \ldots, c_n\} \) respectively. Each state \( s_i \) is randomly initialized with a probability distribution over all word types, with non-zero emission probabilities being assigned only to those words that belong to the state’s corresponding category \( c_i \). (This precludes \( s_i \) from ever emitting a word that does not have some membership weight in \( c_i \), at all training stages of the model.) The HMM is then trained using Baum-Welch (Rabiner, 1989) over the text, and used to label word tokens of sentences in the same data.
Chapter 4

Results

To recap, the broad steps of our procedure are:

1. Part-of-speech induction
   (a) First pass of NMF over high-frequency words
   (b) Second pass over all words

2. Part-of-speech tagging using a hidden Markov model initialized with lexicon from above induced parts of speech

4.1 The Setup

4.1.1 Data
We used two English language corpora – the first 3914 lines (100,676 tokens) of the Wall Street Journal corpus with Penn Treebank tags, and the first 57340 lines (1,161,192 tokens)\(^1\) of the Brown corpus, henceforth referred to as WSJ and Brown respectively. All punctuation categories in the gold standard are collapsed into a single tag\(^2\), and the data lowercased. Since the WSJ tags UH (interjection), FW (foreign word), SYM (symbol), and LS (list item) are only used for a few (<10), very low-frequency tokens, we replace such instances manually with one the other tags (['&', SYM] by ['&', CC], ['perestroika', FW] with ['perestroika', NN], etc).

The resulting tagsets of the WSJ and the Brown corpora are of size 33 and 29 respectively. Besides differences in length and tag size, the two texts come from different domains – one being focused on finance and the other a general news corpus – and are hence likely to show slightly different behaviors under the same algorithm.

\(^1\)We restrict ourselves to fairly small corpora since some of the steps of the NMF and baseline algorithms used here are slow.

\(^2\)These include the uncertain tags, LRB, RRB, and NONE.
The gold standard of category induction (at the type-level) is constructed by simply counting the relative frequency of assignment of a tags to a word. For example, the tags VB and NN are assigned to ‘drop’ 4 and 15 times in the WSJ test; the word’s membership is hence (0.21, VB), (0.79, NN).

4.1.2 Evaluation

The quality of the part of speech tags over the word tokens in the data are primarily evaluated using the variation of information measure described in §2.3.1, Eq. 2.8. Variation of information (VI) is also used to evaluate the soft clustering of word types as produced by NMF. The VI scores reported are gotten by averaging the scores of 10 runs, using different random initializations through k-means seeding.

One issue to consider is the number of categories that we choose to induce. Given no knowledge of the language or data (as is the assumption here), any choice for the number is equally arbitrary. To make comparisons with the gold standard more straightforward and intuitive, we use the same number as the gold standard tagset (33 for experiments on WSJ, and 29 for those on the Brown corpus).

4.1.3 Baseline Algorithms

We implemented two other previously described algorithms for the part of speech induction task. The first is the SVD-based method of Schütze (1995), henceforth referred to as SVDClust, that treats words as points in a vector-space, with the main difference from NMF being the method of dimensionality reduction. It is hence illustrative to compare the specifics of where non-negative matrix factorization is advantageous over the former method. The second baseline algorithm we implement is a hidden Markov model with expectation-maximization, trained over sentences in data, similar to what is discussed as a baseline by Johnson (2007) and Higgins (2002).

The results from each step of our method are compared against (the corresponding steps of) one or more of the above algorithms that accomplish the same task.

In particular, the results at the end of step 1b (category induction) are compared to the generalized context vectors step of SVDClust. Since our performance focus is on soft clustering, the post-projection data of the SVD method is clustered using fuzzy c-means (Dunn, 1973), rather than regular k-means as in the original paper. For consistency, the final membership probabilities for each word as given by fuzzy c-means after SVD are also projected to sparse vectors, using the same algorithm as used in our NMF procedure. The results are also compared to the category assignments derived from the HMM, where a state corresponds to a category, and the membership

3 One area of future work is to examine how the quality of clusters changes as the number is lowered or increased. Another is to use gold-standard-independent evaluation metrics, such as perplexity (Eq. 2.11), within the algorithm to automatically estimate the best number of clusters.
weight of a word \( w \) to a category \( c \) (corresponding to a state \( S_c \)) is given by the ratio of the emission probabilities \( P(w|S_c)/\sum_{d\in\text{categories}} P(w|S_d) \).

Similarly, the numbers at the end of step 2 are compared to the token-level tagging step of SVDClust, as well as to the Viterbi parse of the data given by the HMM used in the previous step.

### 4.2 Step 1a: First Pass of NMF – High-frequency Words

#### WSJ Corpus

Table 4.1 shows the value of variation of information (VI) after NMF (to a reduced space of 33 dimensions, equal to the number of gold standard categories) on all word types of frequency \( \geq 3 \) (3745 words out of 11387), with different feature sets and sparseness constraints. In spaces where limited contextual information was used, the contexts were restricted to the top 1000 words.

<table>
<thead>
<tr>
<th>Features/Sparseness</th>
<th>0.7</th>
<th>0.75</th>
<th>0.8</th>
<th>0.85</th>
<th>0.9</th>
<th>0.95</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Morphology</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LR</td>
<td>4.60</td>
<td>4.59</td>
<td>4.53</td>
<td>4.50</td>
<td>4.34</td>
<td>4.31</td>
<td>4.37</td>
</tr>
<tr>
<td>LR_{1000}</td>
<td>4.58</td>
<td>4.55</td>
<td>4.49</td>
<td>4.44</td>
<td>4.26</td>
<td>4.24</td>
<td>4.33</td>
</tr>
<tr>
<td>LLRR</td>
<td>4.62</td>
<td>4.62</td>
<td>4.54</td>
<td>4.50</td>
<td>4.33</td>
<td>4.33</td>
<td>4.39</td>
</tr>
<tr>
<td>LLRR_{1000}</td>
<td>4.60</td>
<td>4.57</td>
<td>4.50</td>
<td>4.42</td>
<td>4.34</td>
<td>4.29</td>
<td>4.36</td>
</tr>
<tr>
<td>With Morphology</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LR_{1000}S</td>
<td>4.46</td>
<td>4.44</td>
<td>4.31</td>
<td>4.27</td>
<td>4.20</td>
<td>4.18</td>
<td>4.21</td>
</tr>
<tr>
<td>LR_{1000}ST</td>
<td>4.37</td>
<td>4.34</td>
<td>4.31</td>
<td>4.25</td>
<td>4.19</td>
<td>4.17</td>
<td>4.20</td>
</tr>
</tbody>
</table>

As might be expected, adding the morphological features \( S \) and \( T \) produces a huge improvement. Limiting the contextual information proves to be more effective, the scores drop slightly when using a context window of size 4 rather than 2 (namely, LLRR rather than LR).

To understand the effect of sparseness, we plot the sparseness values (as given by category membership) in the gold standard for the 3745 words tested here (Fig. 4.1). The peak at sparseness = 0.95, and the steep increase across all feature sets between sparseness = 0.90 and 0.85 can be explained by the fact that most of the words in the corpus have true sparseness values somewhere between 0.95 and 0.90.

#### Brown Corpus

Table 4.2 shows the same information on the Brown corpus. To account for the larger size of the corpus, the high frequency threshold was set at 4 rather than 3, giving 16689 types out of
49815. The VI scores are slightly higher than those on WSJ (while a larger corpus provides more information, it is most likely offset by the greater amount of noise introduced).

Table 4.2: VI between induced clusters over high-frequency words and gold standard lexicon (Brown), averaged over 10 runs.

<table>
<thead>
<tr>
<th>Features/Sparseness</th>
<th>0.7</th>
<th>0.75</th>
<th>0.8</th>
<th>0.85</th>
<th>0.9</th>
<th>0.95</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Morphology</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LR</td>
<td>4.91</td>
<td>4.90</td>
<td>4.87</td>
<td>4.85</td>
<td>4.72</td>
<td>4.70</td>
<td>4.78</td>
</tr>
<tr>
<td>LR1000</td>
<td>4.81</td>
<td>4.79</td>
<td>4.75</td>
<td>4.72</td>
<td>4.64</td>
<td>4.64</td>
<td>4.68</td>
</tr>
<tr>
<td>LLRR</td>
<td>4.05</td>
<td>4.04</td>
<td>4.00</td>
<td>3.94</td>
<td>4.83</td>
<td>4.79</td>
<td>4.88</td>
</tr>
<tr>
<td>LLRR1000</td>
<td>4.92</td>
<td>4.89</td>
<td>4.85</td>
<td>4.83</td>
<td>4.69</td>
<td>4.67</td>
<td>4.72</td>
</tr>
<tr>
<td>With Morphology</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LR1000S</td>
<td>4.82</td>
<td>4.79</td>
<td>4.75</td>
<td>4.70</td>
<td>4.561</td>
<td>4.59</td>
<td>4.63</td>
</tr>
<tr>
<td>LR1000ST</td>
<td>4.74</td>
<td>4.74</td>
<td>4.71</td>
<td>4.66</td>
<td>4.63</td>
<td>4.57</td>
<td>4.59</td>
</tr>
</tbody>
</table>

4.3 Step 1b: Second Pass of NMF – Accounting for the Low-frequency Words

The main differences of this step from the previous application of NMF are that the context information is at the level of the previously induced clusters, and that the value of the sparseness constraint in the update step is made lower for low-frequency words.

Table 4.3 shows the results of this step on all words in the data, for both the WSJ and Brown
corpora. The sparseness constraint of high-frequency words is set at 0.95, and that of low-frequency (3 or fewer tokens) at 0.80. The score dips slightly when compared to the best results after the first pass; however, the change is not much considering that thousands of low-frequency words (with weak contextual information) have been added to the data.

Table 4.3: VI between induced clusters over the full corpus and gold standard lexicon in both corpora. As before, the scores are the average over 10 runs.

<table>
<thead>
<tr>
<th>Features</th>
<th>WSJ</th>
<th>Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>4.44</td>
<td>4.81</td>
</tr>
<tr>
<td>LR1000</td>
<td>4.36</td>
<td>4.78</td>
</tr>
<tr>
<td>LLRR</td>
<td>4.48</td>
<td>4.91</td>
</tr>
<tr>
<td>LLRR1000</td>
<td>4.41</td>
<td>4.76</td>
</tr>
<tr>
<td>LR1000S</td>
<td>4.29</td>
<td>4.73</td>
</tr>
<tr>
<td>LR1000ST</td>
<td><strong>4.29</strong></td>
<td><strong>4.71</strong></td>
</tr>
</tbody>
</table>

4.3.1 Comparison with Baselines

The same features are now used to cluster the data using singular value decomposition. First, word clusters are found by using contexts over words, analogous to step 1a. These clusters are then used to create context vectors – that is, a word in the feature space $LR$ is characterized by the number of words in each of the categories that occurs to the left and the right of its occurrences. (See the description of the method in §2.2.1.) As in NMF, if the adjacent word is a low-frequency item that has not been clustered, it is assigned its own category in the context vector.

The resulting space is reduced to 50 dimensions (following the number chosen by Schütze (1995). fuzzy c-means, with $c =$ the number of gold standard categories, is then used to cluster the 3745 words in the data.

Table 4.4 lists the VI scores for each feature set in the WSJ corpus (33 categories), using SVDClust and NMF. The score obtained from running a word-level HMM on the corpus is also shown. Similar results are found for the Brown corpus, as shown in Table 4.5.

NMF gives a clear improvement over SVDClust of 10-20% for every feature set. As postulated earlier, its biggest strength appears to be in reflecting the inherent structure of ambiguity in the data. For instance, consider the word ‘offering’, which the WSJ gold standard tags as a noun (NN) in 15 instances, and as a gerund verb (VBG) 11 times. However, SVDClust in the above step gives it a 1.0 membership probability in a cluster consisting mostly of gerunds.

To see that this is not just a failure on the part of c-means, consider the position of the point corresponding to ‘offering’ in the SVD-reduced space. Fig. 4.2 shows the three dimensions of the
Table 4.4: VI between induced clusters and WSJ gold standard, with the baseline algorithms, SVDClust with generalized context vectors and fuzzy c-means, and a word-level HMM. The sparseness for NMF is set to 0.95. All sets of results are averaged over 10 runs. Standard deviations are given in parentheses.

<table>
<thead>
<tr>
<th>Features</th>
<th>SVDClust</th>
<th>NMF</th>
<th>HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>5.39 (0.06)</td>
<td>4.44 (0.04)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000}</td>
<td>5.30 (0.05)</td>
<td>4.36 (0.03)</td>
<td></td>
</tr>
<tr>
<td>LLRR</td>
<td>5.42 (0.05)</td>
<td>4.48 (0.02)</td>
<td></td>
</tr>
<tr>
<td>LLRR_{1000}</td>
<td>5.38 (0.04)</td>
<td>4.41 (0.02)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000}S</td>
<td>4.60 (0.05)</td>
<td>4.29 (0.03)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000}ST</td>
<td>4.57 (0.05)</td>
<td>4.29 (0.03)</td>
<td>4.45 (0.06)</td>
</tr>
</tbody>
</table>

Table 4.5: VI between induced clusters and Brown gold standard, using the baseline algorithms, SVDClust with generalized context vectors and fuzzy c-means, and a word-level HMM. The sparseness for the NMF algorithm is set to 0.95. All sets of results are averaged over 10 runs. Standard deviations are given in parentheses.

<table>
<thead>
<tr>
<th>Features</th>
<th>SVDClust</th>
<th>NMF</th>
<th>HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>5.53 (0.07)</td>
<td>4.81 (0.03)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000}</td>
<td>5.52 (0.06)</td>
<td>4.78 (0.03)</td>
<td></td>
</tr>
<tr>
<td>LLRR</td>
<td>5.59 (0.07)</td>
<td>4.91 (0.05)</td>
<td></td>
</tr>
<tr>
<td>LLRR_{1000}</td>
<td>5.45 (0.05)</td>
<td>4.76 (0.03)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000}S</td>
<td>4.99 (0.04)</td>
<td>4.73 (0.02)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000}ST</td>
<td>4.99 (0.04)</td>
<td>4.71 (0.03)</td>
<td>4.78 (0.09)</td>
</tr>
</tbody>
</table>
data space, corresponding to the three highest singular values. (The co-ordinates of the points in
the remaining dimensions do not vary as much.) We labeled each point using the gold standard,
and found that the verbs (and in this case, the gerunds) and the nouns group themselves together
fairly coherently, with the word ‘offering’ comfortably placed among the gerund verbs. It is hence
unlikely that any clustering algorithm would place ‘offering’ in a cluster with nouns.

Figure 4.2: Data after SVD, showing the position of ‘offering’ relevant to the nouns and the gerund
verbs.

A more detailed analysis of the performance of the algorithm on ambiguous words is given in
§5.1.

4.4 Step 2: Tagging using the Induced Lexicon

The lexicon built from step 1b is now used by a hidden Markov model, with states corresponding
to categories, to assign part-of-speech tags to tokens in the input data. The HMM is initialized so
that the emissions of a state $S_c$ (corresponding to category $c$) have support over all those words
that belong to $c$ in the lexicon. The VI scores at the end of this step for each corpus are shown in
Table 4.6.
Table 4.6: VI between tagged text and gold standard for both corpora, averaged over 10 runs.

<table>
<thead>
<tr>
<th>Features</th>
<th>WSJ</th>
<th>Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>4.23</td>
<td>4.65</td>
</tr>
<tr>
<td>LR(_{1000})</td>
<td>4.19</td>
<td>4.63</td>
</tr>
<tr>
<td>LLRR</td>
<td>4.23</td>
<td>4.74</td>
</tr>
<tr>
<td>LLRR(_{1000})</td>
<td>4.20</td>
<td>4.68</td>
</tr>
<tr>
<td>LR(_{1000S})</td>
<td>4.02</td>
<td>4.56</td>
</tr>
<tr>
<td>LR(_{1000ST})</td>
<td><strong>4.02</strong></td>
<td><strong>4.55</strong></td>
</tr>
</tbody>
</table>

4.4.1 Comparison with Baselines

The final step of Schütze’s algorithm (see §2.2.2) and an HMM with 33 states (with random initialization) are both implemented and run on the data. Table 4.7 and 4.8 list the scores for each feature set using the HMM and the token tagging step of SVDClust, on the WSJ and Brown corpora respectively.

Table 4.7: VI between tagged text and gold standard in WSJ for the NMF algorithm, SVDClust, and HMM. Results are averaged over 10 runs, with standard deviations shown in parentheses.

<table>
<thead>
<tr>
<th>Features</th>
<th>NMF</th>
<th>SVDClust</th>
<th>HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>4.23 (0.04)</td>
<td>5.30 (0.05)</td>
<td></td>
</tr>
<tr>
<td>LR(_{1000})</td>
<td>4.19 (0.03)</td>
<td>5.25 (0.05)</td>
<td></td>
</tr>
<tr>
<td>LLRR</td>
<td>4.23 (0.04)</td>
<td>5.26 (0.04)</td>
<td></td>
</tr>
<tr>
<td>LLRR(_{1000})</td>
<td>4.20 (0.03)</td>
<td>5.29 (0.05)</td>
<td></td>
</tr>
<tr>
<td>LR(_{1000S})</td>
<td>4.02 (0.02)</td>
<td>4.94 (0.04)</td>
<td></td>
</tr>
<tr>
<td>LR(_{1000ST})</td>
<td><strong>4.02</strong> (0.02)</td>
<td><strong>4.94</strong> (0.04)</td>
<td></td>
</tr>
</tbody>
</table>

4.5 Comparison with Other Previous Work

Objective comparison with related work is difficult, since published results are reported on different corpora, or use other evaluation metrics. To estimate where our results stand in relation to other algorithms, we extrapolate from the survey of Headden et al. (2008), which implements and evaluates 6 different procedures, and finds that the Ney-Essen with morphology (Clark, 2003), henceforth referred to as NEMorph, gets the highest VI scores.
Table 4.8: VI between tagged text and gold standard in the Brown corpus for the NMF algorithm, SVDClust, and HMM. Results are averaged over 10 runs, with standard deviations shown in parentheses.

<table>
<thead>
<tr>
<th>Features</th>
<th>NMF</th>
<th>SVDClust</th>
<th>HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>4.65 (0.03)</td>
<td>5.72 (0.06)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000}</td>
<td>4.63 (0.03)</td>
<td>5.70 (0.05)</td>
<td></td>
</tr>
<tr>
<td>LLRR</td>
<td>4.74 (0.02)</td>
<td>5.70 (0.05)</td>
<td>4.77 (0.09)</td>
</tr>
<tr>
<td>LLRR_{1000}</td>
<td>4.68 (0.02)</td>
<td>5.71 (0.04)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000S}</td>
<td>4.56 (0.02)</td>
<td>5.42 (0.03)</td>
<td></td>
</tr>
<tr>
<td>LR_{1000ST}</td>
<td>4.55 (0.02)</td>
<td>5.41 (0.03)</td>
<td></td>
</tr>
</tbody>
</table>

We ran the implementation of NEMorph\(^4\) on our WSJ corpus, with \(k = 37\) (see §2.2.3 for a description of the algorithm), and 10 states for its embedded morphology HMM. The results have a VI of 3.71, which is a little better than that of our model (4.02). However, note that the two algorithms are not quite comparable, since NEMorph does not make allowances for soft clustering. Further, since NEMorph outperforms all other algorithms by a margin of around 1.00 as reported in Headden et al. (2008), it is likely that our model also does relatively better than those algorithms.

Chapter 5

Discussion

Note: All results of our NMF-based algorithm (referred to as NMF in the chapter), as well as those of SVDClust, are taken from runs of the respective algorithms using the feature set LR_{1000}.

5.1 Detection of Ambiguity

The main advantage of NMF, as hypothesized, seems to be in the way it deals with ambiguity: despite using fuzzy clustering, the first baseline algorithm SVDClust tends to mostly assign words to single categories. We picked a representative sample of the clustered data in both sets of results. Table 5.1 shows (extracts of) some of the discovered clusters after step 1b$^1$ on the WSJ corpus. Most of the words shown are ambiguous in the gold standard, but that isn’t reflected in the results. However, the clusters after NMF (Table 5.2) show a good deal of overlap.

Table 5.1: Sample of WSJ clusters after SVDClust, step 1b. The left column is the tag that most corresponds to the cluster on the right. Words that appear in more than one category are in bold.

<table>
<thead>
<tr>
<th>VBP</th>
<th>want, change, lead, stand, call, care, provide, accept, ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>VBN</td>
<td>reported, found, failed, appeared, <strong>sold</strong>, proposed, ...</td>
</tr>
<tr>
<td>VBD</td>
<td>made, <strong>sold</strong>, posted, used, named, paid, noted, prompted, ...</td>
</tr>
<tr>
<td>VB</td>
<td><strong>fall</strong>, be, see, make, support, let, find, happen, die, yield, ...</td>
</tr>
<tr>
<td>NN</td>
<td>increase, issue, <strong>fall</strong>, limit, motive, report, income, end, ...</td>
</tr>
</tbody>
</table>

One the other hand, the HMM tends to overestimate the number of ambiguous words. Table 5.3 shows the detection of the most frequent ambiguous words in the WSJ corpus using NMF as compared to the two baselines. An ambiguous word in the gold standard is taken as one that is

$^1$We focus on the results of this step since it is the most basic level for evaluation of NMF on type-level soft clustering.
Table 5.2: Sample of WSJ clusters after NMF, step 1b. Words that are in different clusters from SVDClust, or are added to new ones are in italics. Words that are no longer in the same cluster are struck out.

| VBP | want, change, lead, stand, call, make, care, provide, accept, see, ... |
| VBN | reported, found, failed, appeared, sold, proposed, made, posted, ... |
| VBD | made, sold, posted, used, named, paid, noted, prompted, proposed, ... |
| VB  | fall, be, see, make, support, let, find, report, happen, die, yield, end, ... |
| NN  | increase, issue, fall, limit, motive, report, income, end, support, change, ... |

tagged with more than one category in the gold standard. (To compensate for errors in annotation, a word type is said to belong to category \( x \) if at least 2 of its tokens are tagged with \( x \).)

As indicated, SVDClust finds too few ambiguous words. On the other hand, the HMM overestimates them, while NMF gets the highest F-score.

Table 5.3: Detection of ambiguous words (WSJ)

<table>
<thead>
<tr>
<th>Number of ambiguous words among top 1000 types</th>
<th>Gold standard</th>
<th>NMF</th>
<th>SVDClust</th>
<th>HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>100%</td>
<td>89%</td>
<td>64%</td>
<td>49%</td>
</tr>
<tr>
<td>Recall</td>
<td>100%</td>
<td>68%</td>
<td>14%</td>
<td>75%</td>
</tr>
<tr>
<td>F-score</td>
<td>100%</td>
<td>77%</td>
<td>22%</td>
<td>50%</td>
</tr>
</tbody>
</table>

To get a better insight into the performance on ambiguous words, we analyze the distribution of certain common words in both corpora after the NMF algorithm, as well as the two baselines. The words examined are:

1. **to**, which occurs almost equally frequently as a preposition (43% of the instances), and as an infinitive marker (57%) in the Brown corpus,

2. **’s**, which, in the WSJ corpus, is used as a possessive marker about 87% of the time and a verb (VBZ, being the contracted form of ‘is’) the rest of the time, and

3. **before** which occurs in the Brown corpus as an adverb (16%), preposition (40%), and subordinating conjunction (44%).

Table 5.4 shows the clusters that the NMF and the baselines assign to each of the three words.
Each cluster is denoted by a label that shows the gold standard category/categories that it most corresponds to.

Table 5.4: Distribution after POS induction (using NMF, SVDClust, and HMM) of common ambiguous words. Clusters are denoted by labels which are assigned by hand based on their characteristics. Membership probabilities of the word to each cluster are shown in parentheses. Also shown is the distribution of each word among the gold standard categories in the corpus used.

<table>
<thead>
<tr>
<th>Word</th>
<th>Gold Standard</th>
<th>Clusters Assigned</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>NMF</td>
</tr>
<tr>
<td>to</td>
<td>infinitive marker (0.57) prepositions (0.43)</td>
<td>modals (0.32) prepositions (0.59) pronouns+proper nouns (0.08)</td>
</tr>
<tr>
<td>(in Brown)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>’s</td>
<td>possessives (0.88) present tense verbs (0.12)</td>
<td>possessives +determiners(0.94) present tense (0.06)</td>
</tr>
<tr>
<td>(in WSJ)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>before</td>
<td>prepositions (0.40) adverbs (0.16) subordinating conjunctions (0.44)</td>
<td>conjunctions (0.93) other (0.07)</td>
</tr>
<tr>
<td>(in Brown)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NMF is able to deduce that to functions as a preposition as well as another part of speech, with more or less equal membership. However, it does not find the ‘infinitive marker’ category (which in the gold standard, consists of a single word); rather, it groups the word along with modals (can, will, etc.) – most likely because the right contexts of modals are similar to those of to when it occurs in an infinitive. SVDClust picks up the fact that to functions as preposition, but as expected, only assigns the word to a one cluster. The HMM gets a more reasonable distribution shape, giving a 48% membership probability to a cluster of prepositions. However, due to the fact the clusters it induces are large and fairly imprecise, the rest of the membership weight is given to unrelated clusters including nouns and adverbs.

The baseline results for the words ’s and before are similar, with SVDClust pigeonholing the words into a single cluster, and the HMM getting close to approximating the ambiguity, but spreading membership over too many categories. NMF is less effective in categorizing before than it was in the case of to – it assigns the word mainly to the cluster of conjunctions, with some weight on a few unrelated clusters. On the other hand, it manages to pick the correct two categories for
's, although it shifts most of the weight to the class of possessives, and combines the class with the determiners.

5.2 Summary of Discovered Clusters

To assess the general performance of our algorithm, it is useful to take a look at the quality of clusters induced. About half the clusters (in each corpus) are coherent and correspond to either a gold standard part of speech category, or some other syntactically sensible group. The other clusters do maintain some coherence, but tend to span multiple traditional categories; for example, 2 of the clusters from the WSJ consist of a mixture of pronouns, plural nouns, adjectives, and miscellaneous words.

Table 5.5 and 5.6 shows some of the ‘good’ clusters of the 33 categories induced by NMF from the total 11387 types in the WSJ corpus. The words shown in each clusters are restricted to the top 50 or so, and are ordered by frequency. (Recall that we converted the corpus to lowercase before applying the algorithms.)

Table 5.7 shows some of the clusters that don’t correspond to natural categories. As above, the top most frequent words of each cluster are shown.

Besides the improved detection of ambiguity, the quality of clusters appear more or less similar to those found by some of the other existing part-of-speech induction algorithms. In particular, categories that are easily given by contextual and/or morphological information show up as coherent clusters. For instance, most prepositions are preceded by a verb and followed by a determiner, numbers are followed by plural nouns, past tense verbs usually have the suffix ed, etc. On the other hand, syntactic categories such as common nouns, pronouns, and certain adjectives, adverbs, and prepositions, occur in varied contexts, with no distinct morphology. This makes it difficult to induce such categories when using a vector space model with purely contextual/morphological information.

5.3 Future Work

There are several avenues of further work in the details of using NMF for part-of-speech induction. The algorithm that implements the projection of vertices for the sparseness constraint is very slow; any practical uses of NMF would require a more efficient procedure. It is also worth examining how alternate cost functions, like the Kullback-Leibler divergence or cosine similarity, affect results. As in all category induction tasks, it will also be interesting to test this method on data in other languages.

Another point about evaluating the method is a general one concerning the choice of the number of clusters. For simplicity, we picked the number to be equal to the number of gold standard tags. However, there is no clean one-to-one correspondence of the induced clusters with the gold
Table 5.5: Overview of some good clusters induced from the WSJ. The middle column shows the gold standard categories (if any) that each cluster corresponds to. Words that appear in multiple clusters are in italics.

<table>
<thead>
<tr>
<th>Cluster sample</th>
<th>Matching category</th>
<th># members</th>
</tr>
</thead>
<tbody>
<tr>
<td>on from because so after over only out under before just through</td>
<td>IN (prepositions)</td>
<td>30</td>
</tr>
<tr>
<td>until where between without though cut despite veto almost act</td>
<td></td>
<td></td>
</tr>
<tr>
<td>once across include something anything within nor neither</td>
<td></td>
<td></td>
</tr>
<tr>
<td>of in and for that said by at as with have but about if</td>
<td>IN</td>
<td>18</td>
</tr>
<tr>
<td>into even say while 're</td>
<td></td>
<td></td>
</tr>
<tr>
<td>two three 10 30 1 100 500 six 2 five 40 four 3 18 6 4 31 17 60</td>
<td>CD (numbers)</td>
<td>182</td>
</tr>
<tr>
<td>70 16 200 55 130 seven 24 300 45 19 23 75 900 65 62 1.5 250</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400 150 12.5 49 120 1.1 straight 20,000 47 29 61 3.1 180 7.3 ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bush cray hahn nixon phelan ward thomas mcgovern coleman lane</td>
<td>NNP (last names)</td>
<td>168</td>
</tr>
<tr>
<td>bernstein simmons spiegel dinkins courter martin baum wilder</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mcalpine steinberg newhouse dingell trudeau chase black</td>
<td></td>
<td></td>
</tr>
<tr>
<td>neuberger murray markey reupke van ackerman madison veraldi</td>
<td></td>
<td></td>
</tr>
<tr>
<td>yamamoto Giuliani harper sullivan ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mr. mrs. john robert william james a. michael r. j. d. david</td>
<td>NNP (first names)</td>
<td>162</td>
</tr>
<tr>
<td>ms. richard dr. mary t. charles c. peter cosby mitsui m.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>standardized kent e. b. edward g. l. goldman smith donald</td>
<td></td>
<td></td>
</tr>
<tr>
<td>douglas paul jack eastern miles george h. p. preparation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beth s. jim jerry f. frank christopher joseph ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>made fell reported used approved increased buying held further</td>
<td>VBD (past tense verbs)</td>
<td>246</td>
</tr>
<tr>
<td>survey led review study followed largely rejected offset hurt produced cited</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sought schools boosted provided considered</td>
<td></td>
<td></td>
</tr>
<tr>
<td>created claim emissions lines passed branch formed owned directly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lives covered crude showing funded signed felt thought jumped</td>
<td></td>
<td></td>
</tr>
<tr>
<td>property carried determined code challenge heard ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>made said added gives sold reported used based closed cut</td>
<td>VBN (past participle verbs)</td>
<td>199</td>
</tr>
<tr>
<td>offered approved noted ended found increased received estimated proposed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>followed given issued turned rejected read earned offset</td>
<td></td>
<td></td>
</tr>
<tr>
<td>consented hurt produced aimed won required sought hit introduced</td>
<td></td>
<td></td>
</tr>
<tr>
<td>limited bought listed allowed boosted fixed opened reduced valued</td>
<td></td>
<td></td>
</tr>
<tr>
<td>created placed seen spent surged determined ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>be make buy take get sell help pay see force face support go</td>
<td>VB (imperative/subjunctive)</td>
<td>111</td>
</tr>
<tr>
<td>give raise continue seek find keep receive slow reduce</td>
<td></td>
<td></td>
</tr>
<tr>
<td>meet remain improve prevent boost win watch consider</td>
<td></td>
<td></td>
</tr>
<tr>
<td>begin build acquire approve bring prove block extend findings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>let allow produce operate print turn ease avoid complete send ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>the a ‘s that an this u.s. market their program all her most big economic</td>
<td>DT (determiners)</td>
<td>19</td>
</tr>
<tr>
<td>each early fiscal great neither either</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 5.6: Overview of some good clusters induced from the WSJ (continued from Table 5.5). The middle column shows the gold standard categories (if any) that each cluster corresponds to. Words that appear in multiple clusters are in italics.

<table>
<thead>
<tr>
<th>Cluster sample</th>
<th>Matching category</th>
<th># members</th>
</tr>
</thead>
<tbody>
<tr>
<td>'s is are will was has n’t does rates remains plan adds includes seem appears seems needs causing wants sells forces asks admits enters tied holds seeks explains considers thinks signals cites quota tend employs spent represented contends spends fails combines complain sees puts . . .</td>
<td>VBZ (present tense verbs)</td>
<td>78</td>
</tr>
<tr>
<td>% million billion rates income less contract students charge sugar proposal limit better away wines evidence levels profits valley units benefits percentage acquisitions tons institutional transplants air expenses reduction range barrels itself protection assistance sanctions shipyard admitting deaths cabernet warrants pension stay county bone deposit mortgages trillion band pickers . . .</td>
<td>NNS (plural nouns)</td>
<td>391</td>
</tr>
<tr>
<td>to will would were could can do may did might added should wo must got decided ca believed shall ’ll</td>
<td>MD (modals)</td>
<td>20</td>
</tr>
</tbody>
</table>

standard – which does not necessarily mean that the induced categories are not reasonable parts of speech, but simply that they sometimes more specific or general than those in the gold standard tagset. For example, both our algorithm and the baselines clump the three adjective classes JJ, JJR (comparative), and JJS (superlative) together, and do not recognize the ‘WH’ classes (WDT, WP, WP$, WRB), but distribute their words among other related clusters. On the other hand, the algorithm makes finer distinctions than are reflected in the standard tagsets – first names, last names, dates, and so on. While one can experiment with different numbers of clusters and report the results on each (as has been done in some previous work), there isn’t a clear understanding of what the optimal number of categories are, given the data. Automatic discovery of the number of clusters $k$ may be done by picking $k$ that maximizes the log-likelihood of a held-out set of data.

Further, recall that the columns of $W$ are feature vectors that represent prototypical ‘words’, which form the basis of the original data. One possible direction of work is to use these feature vectors for clustering or syntactic category identification in other applications.
Table 5.7: Overview of some of the bad clusters induced from the WSJ. The gold standard categories (if any) that each cluster corresponds to is shown on the right column. Words that appear in multiple clusters are in italics.

<table>
<thead>
<tr>
<th>Cluster sample</th>
<th>Matching categories</th>
<th># members</th>
</tr>
</thead>
<tbody>
<tr>
<td>atlanta basic mae pharmaceutical aba junk-bond blue <em>malaysia</em></td>
<td>NNP + NNS + adjectives</td>
<td>328</td>
</tr>
<tr>
<td>philippines buyer model various deputy creditors auctions <em>pricing</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ftc legislative lawsuit investment-grade <em>excess</em> conventional</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fast-food <em>sweeping</em> democrat buy-outs authors <em>salary</em> older</td>
<td></td>
<td></td>
</tr>
<tr>
<td>downturn <em>police</em> utility economics founder sizable 1982 <em>flow</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>qualified regarding <em>pending</em> bronx hungary possibility harvest ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>they</em> which <em>we</em> he year <em>i</em> you prices she <em>business</em> exchange</td>
<td>NNS + pronouns+ other</td>
<td>173</td>
</tr>
<tr>
<td>them stocks funds <em>money</em> higher month growth yeargin however</td>
<td></td>
<td></td>
</tr>
<tr>
<td>operations <em>safety</em> close work division volume dividend fees legislation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>results questions transactions <em>product</em> 15,000 <em>rise</em> <em>employees</em> bells</td>
<td></td>
<td></td>
</tr>
<tr>
<td>consulting <em>german</em> actually employer sounds reruns lake politics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>subsidiaries railing explained gordon steelmakers suppliers ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>what</em> like <em>because</em> since yield among during use against below around</td>
<td></td>
<td></td>
</tr>
<tr>
<td>finance put result become making above <em>saying</em> <em>cost</em> meanwhile risk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sometimes <em>target</em> wrote indeed raising approach substantially</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sure <em>assuming</em> figure design larger argue alone contained <em>requires</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>means suffer denying whom expand 1980s <em>decide</em> retin-a throughout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>suggest version reflect reach ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>trading years time issue common well industry vice computer service</td>
<td></td>
<td></td>
</tr>
<tr>
<td>campbell costs move fund fine stake day <em>georgia</em> weeks ii stock-index</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wine america gain express korea thrift sector sea trucks aid herald</td>
<td></td>
<td></td>
</tr>
<tr>
<td>west nine crash partly 5,000 <em>georgia-pacific</em> utilities testing southeast</td>
<td></td>
<td></td>
</tr>
<tr>
<td>competition associates guild treatment ratners brazil cancer premium</td>
<td></td>
<td></td>
</tr>
<tr>
<td>giant staff separate heating 1992 <em>taiwan</em> metals courts segments</td>
<td></td>
<td></td>
</tr>
<tr>
<td>training name successor machine suggests ...</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>they</em> there <em>we</em> investors yesterday companies <em>bonds</em> banks days</td>
<td>NNS + pronouns + other</td>
<td>312</td>
</tr>
<tr>
<td>managers firms contracts terms materials here countries life union</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chipsterling executives <em>lawyers</em> computers increases <em>meeting</em> rider</td>
<td></td>
<td></td>
</tr>
<tr>
<td>imports why concerns <em>corp</em> risks ratings stations withdraw</td>
<td></td>
<td></td>
</tr>
<tr>
<td>responsibilities hopefully chateau encouraging quick crisis suspend</td>
<td></td>
<td></td>
</tr>
<tr>
<td>changing 2009 homelessness candidates ...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 6

Conclusion

This paper deals with an important problem in part-of-speech induction – namely, inducing a *soft clustering* of word types into syntactic categories, so that each word is assigned to one or more categories. We adapt the rank-reducing technique of non-negative matrix factorization to this problem, taking advantage of its natural applicability to soft clustering.

Part-of-speech induction is accomplished by treating words as vectors in some feature space that provides cues to the words’ syntactic categories. Matrix factorization decomposes this space into some number of basis components, and a set of coefficients corresponding to each word that allows it to be expressed as a linear combination of these components. If every element of the component and coefficient vector is non-negative, and the $L_1$ norm of each coefficient vector equal to 1, a natural interpretation of this setup is that the basis components are *categories*, with the coefficient vector of a word specifying the membership probability of the word to each of the categories. Another constraint on the coefficient vectors requires them to be sparse, this ensuring that each word can only belong to a few of total set of categories.

We apply the above procedure to category induction using two English language corpora, experimenting with different sparseness values and feature sets containing different types of contextual and morphological information. As seen in §5.1, it is found that NMF significantly improves on related methods in being able to correctly assign words to multiple clusters. Further, NMF performs consistently better than the baseline algorithms, both in the main task of part-of-speech induction, and in part-of-speech tagging, which makes use of the induced categories.

Acknowledgments

Many thanks to John Goldsmith, Partha Niyogi, Jason Riggle and Anne Rogers for their comments and guidance.
Appendix A

Proof of Convergence of the NMF
Update Rule

The paper by Lee and Seung (2001) treats non-negative matrix factorization as a constraint optimization problem. When using the squared distance cost function, this amounts to minimizing \( \|V - WH\|^2 \), under the constraint that \( W \) and \( H \) are non-negative. While gradient descent is one simple method of finding the local minimum of the cost function, it tends to be very slow and sensitive to the step size chosen. The authors hence propose an alternative algorithm using *multiplicative updates*. \( W \) and \( H \) are chosen to be arbitrary non-negative matrices of the required sizes, and the following step iterated until convergence.

\[
H_{ij} \leftarrow H_{ij} \cdot \frac{(W^T V)_{ij}}{(W^T WH)_{ij}}; \quad W_{ij} \leftarrow W_{ij} \cdot \frac{(VH^T)_{ij}}{(WHH^T)_{ij}} \quad (A.1)
\]

This will result in \( \|V - WH\|^2 \) converging to at least a local minimum as long as it is *non-increasing* under the update steps. To prove the convergence, the authors borrow the concept of an *auxiliary function* from Expectation-Maximization.

A function \( G(h, h') \) is said to be an auxiliary function for \( F(h) \) if \( G(h, h') \geq F(h) \) and \( G(h, h) = F(h) \). Then, \( F \) is non-increasing under the update rule

\[
h^{t+1} = \arg \min_h G(h, h') \quad (A.2)
\]

since \( F(h^{t+1}) \leq G(h^{t+1}, h^t) \) (by definition), \( G(h^{t+1}, h^t) \leq G(h^t, h^t) \) (from Eq. A.2), and \( G(h^t, h^t) = F(h) \). Thus, by iterating the update rule, the local minimum \( h_{\text{min}} \) can be found starting from some \( h_0 \):

\[
F(h_0) \geq F(h_1) \geq F(h_2) \geq \ldots \geq F(h_t) \geq F(h_{t+1}) \ldots \geq F(h_{\text{min}}) \quad (A.3)
\]

The next step is to find an auxiliary function \( G \) for the cost function \( \|V - WH\|^2 \), and the corresponding update rule given by Eq. A.2. The construction of \( G \) follows from Lemma 1.
Lemma 1 For a diagonal matrix $K(h^t)$ where $K_{ab}(h^t) = \frac{\delta_{ab}(W^TW h^t)_a}{h_a^t}$

$$G(h, h^t) = F(h^t) + (h - h^t)^T \nabla F(h^t) + 0.5(h - h^t)^T K(h^t)(h - h^t)$$ \hspace{1cm} (A.4)

is an auxiliary function for $F(h) = 0.5 \sum_i (v_i - \sum_a W_{ia} h_a)^2$.

**Proof:** We need to show that $G(h, h) = F(h)$ and $G(h, h^t) \geq F(h)$.

$G(h, h) = F(h)$ follows from Eq. A.4.

Now note that $F(h) = F(h^t) + (h - h^t)^T \nabla F(h^t) + 0.5(h - h^t)^T (W^T W) (h - h^t)$. Hence, to prove that $G(h, h^t) \geq F(h)$, we must show that

$$(h - h^t)^T (K(h^t) - W^T W)(h - h^t)$$ \hspace{1cm} (A.5)

or equivalently, that $K(h^t) - W^T W$ is positive semidefinite. To prove this, consider the matrix $M_{ab}(h^t) = h_a^t (K(h^t) - W^T W)_{ab} h_b^t$. $K(h^t) - W^T W$ is positive semidefinite iff $M$ is positive semidefinite.

**Lemma 2** $M_{ab}(h^t)$ is positive semidefinite.

**Proof:**

$$v^T M v = \sum_{ab} v_a M_{ab} v_b = \sum_{ab} h_a^t (W^T W)_{ab} h_b^t v_a^2 - v_a h_a^t (W^T W)_{ab} h_b^t v_b$$ \hspace{1cm} (A.6)

$$= \sum_{ab} h_a^t (W^T W)_{ab} h_b^t [0.5v_a^2 + 0.5v_b^2 - v_a v_b]$$ \hspace{1cm} (A.7)

$$= 0.5 \sum_{ab} h_a^t (W^T W)_{ab} h_b^t (v_a - v_b)^2 \geq 0$$ \hspace{1cm} (A.8)

Therefore, $G(h, h^t)$ as given in Eq. A.4 is an auxiliary function for $F$, giving the update rule

$$h^{t+1} = h^t - K(h^t)^{-1} \nabla F(h^t)$$ \hspace{1cm} (A.9)

that is:

$$h_{a}^{t+1} = h_{a}^{t} - \frac{(W^T x)_a}{(W^T W h^t)_a}$$ \hspace{1cm} (A.10)

This proves that the update rule for $H$ results in a non-increasing $||V - WH||^2$. The same proof can be applied to show that the cos functions is non-increasing under the update rule for $W$ as well.
Bibliography


46