Spectral Methods for Natural Language Processing

Jang Sun Lee (Karl Stratos)

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2016

ABSTRACT

Spectral Methods for Natural Language Processing

Jang Sun Lee (Karl Stratos)

Many state-of-the-art results in natural language processing (NLP) are achieved with statistical models involving latent variables. Unfortunately, computational problems associated with such models (for instance, finding the optimal parameter values) are typically intractable, forcing practitioners to rely on heuristic methods without strong guarantees. While heuristics are often sufficient for empirical purposes, their de-emphasis on theoretical aspects has certain negative ramifications. First, it can impede the development of rigorous theoretical understanding which can generate new ideas and algorithms. Second, it can lead to black art solutions that are unreliable and difficult to reproduce.

In this thesis, we argue that spectral methods—that is, methods that use singular value decomposition or other similar matrix or tensor factorization—can effectively remedy these negative ramifications. To this end, we develop spectral methods for two unsupervised language processing tasks. The first task is learning lexical representations from unannotated text (e.g., hierarchical clustering of a vocabulary). The second task is estimating parameters of latent-variable models used in NLP applications (e.g., for unsupervised part-of-speech tagging). We show that our spectral algorithms have the following advantages over previous methods:

1. The algorithms provide a new theoretical framework that is amenable to rigorous analysis. In particular, they are shown to be statistically consistent.

2. The algorithms are simple to implement, efficient, and scalable to large amounts of data. They also yield results that are competitive with the state-of-the-art.

Table of Contents

List of Figures **vii**

List of Tables **x**

1 Introduction **1**

1.1 Motivation 1

1.2 Learning Lexical Representations 4

1.2.1 Hierarchical Word Clusters 4

1.2.2 Word Embeddings 5

1.3 Estimating Parameters of Latent-Variable Models 6

1.3.1 Unsupervised POS Tagging 6

1.3.2 Phoneme Recognition 7

1.4 Thesis Overview 8

1.5 Notation 8

2 Related Work **10**

2.1 Latent-Variable Models in NLP 10

2.2 Representation Learning in NLP 11

2.3 Spectral Techniques 12

I The Spectral Framework **15**

3 A Review of Linear Algebra **16**

3.1 Basic Concepts 16

3.1.1 Vector Spaces and Euclidean Space 16
3.1.2 Subspaces and Dimensions 17
3.1.3 Matrices 18
3.1.4 Orthogonal Matrices 21
3.1.5 Orthogonal Projection onto a Subspace 21
3.1.6 Gram-Schmidt Process and QR Decomposition 22
3.2 Eigendecomposition 24
3.2.1 Square Matrices 24
3.2.2 Symmetric Matrices 27
3.2.3 Variational Characterization 28
3.2.4 Semidefinite Matrices 30
3.2.5 Numerical Computation 33
3.3 Singular Value Decomposition (SVD) 39
3.3.1 Derivation from Eigendecomposition 39
3.3.2 Variational Characterization 42
3.3.3 Numerical Computation 43
3.4 Perturbation Theory 44
3.4.1 Perturbation Bounds on Singular Values 44
3.4.2 Canonical Angles Between Subspaces 44
3.4.3 Perturbation Bounds on Singular Vectors 46

4 Examples of Spectral Techniques **52**
4.1 The Moore–Penrose Pseudoinverse 52
4.2 Low-Rank Matrix Approximation 53
4.3 Finding the Best-Fit Subspace 55
4.4 Principal Component Analysis (PCA) 55
4.4.1 Best-Fit Subspace Interpretation 56
4.5 Canonical Correlation Analysis (CCA) 57
4.5.1 Least Squares Interpretation 59
4.5.2 New Coordinate Interpretation 60
4.5.3 Dimensionality Reduction with CCA 60

4.6 Spectral Clustering 66
4.7 Subspace Identification 68
4.8 Alternating Minimization Using SVD 71
4.9 Non-Negative Matrix Factorization 73
4.10 Tensor Decomposition 75

II Inducing Lexical Representations **79**

5 Word Clusters Under Class-Based Language Models **80**
5.1 Introduction 81
5.2 Background 82
5.2.1 The Brown Clustering Algorithm 82
5.2.2 CCA and Agglomerative Clustering 84
5.3 Brown Model Definition 85
5.4 Clustering Under the Brown Model 86
5.4.1 An Overview of the Approach 86
5.4.2 Spectral Estimation of Observation Parameters 86
5.4.3 Estimation from Samples 89
5.4.4 Agglomerative Clustering 91
5.5 Experiments 93
5.5.1 Experimental Settings 93
5.5.2 Comparison to the Brown Algorithm: Quality 94
5.5.3 Comparison to the Brown Algorithm: Speed 95
5.5.4 Effect of the Choice of κ and Context 98
5.6 Conclusion 98

6 Word Embeddings from Decompositions of Count Matrices **102**
6.1 Introduction 103
6.2 Background in CCA 104
6.2.1 CCA Objective 104
6.2.2 Exact Solution via SVD 105

6.2.3 Using CCA for Word Representations . . . 105
6.3 Using CCA for Parameter Estimation . . . 106
6.3.1 Clustering under a Brown Model . . . 107
6.3.2 Spectral Estimation . . . 108
6.3.3 Choice of Data Transformation . . . 109
6.4 A Template for Spectral Methods . . . 110
6.5 Related Work . . . 113
6.6 Experiments . . . 113
6.6.1 Word Similarity and Analogy . . . 113
6.6.2 As Features in a Supervised Task . . . 117
6.7 Conclusion . . . 117

III Estimating Latent-Variable Models 119

7 Spectral Learning of Anchor Hidden Markov Models 120
7.1 Introduction . . . 121
7.2 The Anchor Hidden Markov Model . . . 122
7.3 Parameter Estimation for A-HMMs . . . 123
7.3.1 NMF . . . 123
7.3.2 Random Variables . . . 125
7.3.3 Derivation of a Learning Algorithm . . . 125
7.3.4 Construction of the Convex Hull Ω . . . 128
7.4 Experiments . . . 132
7.4.1 Background on Unsupervised POS Tagging . . . 133
7.4.2 Experimental Setting . . . 134
7.4.3 Practical Issues with the Anchor Algorithm . . . 135
7.4.4 Tagging Accuracy . . . 137
7.4.5 Qualitative Analysis . . . 138
7.5 Related Work . . . 139
7.5.1 Latent-Variable Models . . . 139

7.5.2 Unsupervised POS Tagging . . . 139
7.6 Conclusion . . . 141

8 Spectral Learning of Refinement Hidden Markov Models 144
8.1 Introduction . . . 144
8.2 Related Work . . . 146
8.3 The R-HMM Model . . . 147
8.3.1 Definition of an R-HMM . . . 147
8.4 The Forward-Backward Algorithm . . . 148
8.5 Spectral Estimation of R-HMMs . . . 151
8.5.1 Random Variables . . . 151
8.5.2 Estimation of the Operators . . . 154
8.6 The Spectral Estimation Algorithm . . . 156
8.7 Experiments . . . 157
8.8 Conclusion . . . 161

9 Conclusions 163
9.1 Limitations of the Existing Spectral Framework . . . 165
9.2 Future Work . . . 166
9.2.1 Flexible Framework for Spectral Optimization . . . 166
9.2.2 Online/Randomized Spectral Methods . . . 167
9.2.3 Spectral Methods for Other NLP Tasks . . . 167

IV Bibliography 169

Bibliography 170

V Appendices 187

A Appendix for Chapter 5 188
A.1 Clustering Algorithm of Brown *et al.* [1992] . . . 188
A.2 Incorporating Richer Context . . . 190

A.3 Consistency of Clusters: Proof of Theorem 5.4.4 191
A.4 Sample Complexity: Proof of Theorem 5.4.6 193

B Appendix for Chapter 6 **197**
B.1 Proof of Theorem 6.3.1 197

List of Figures

1.1 Two forms of lexical representations for the vocabulary: `coffee`, `tea`, `dog`, `cat`, `walk`, `run`, `walked`, `ran`. (a) Hierarchical word clusters. (b) Word embeddings. 5

3.1 The Gram-Schmidt process. 23
3.2 QR decomposition. 24
3.3 A basic version of the power iteration method. 35
3.4 A basic version of the subspace iteration method. 36
3.5 A basic version of the Lanczos method. 38

5.1 A standard implementation of the Brown clustering algorithm. See Figure A.1 for more details. 83
5.2 An example of a Brown word-cluster hierarchy. Each node in the tree is labeled with a bit string indicating the path from the root node to that node, where 0 indicates a left branch and 1 indicates a right branch. 83
5.3 Illustration of our clustering scheme. (a) Original rows of $\sqrt{O}$. (b) After row-normalization. 87
5.4 Estimation of M from samples. 89
5.5 Effect of the choice of κ and context on (a) MI and (b) NER dev F1 score. We used 1,000 clusters on RCV1 with vocabulary size 50k. In (a), the horizontal line is the MI achieved by Brown clusters. In (b), the top horizontal line is the F1 score achieved with Brown clusters and the bottom horizontal line is the baseline F1 score achieved without using clusters. 96

5.6 A $O(1)$ function that is called $O(nm^2)$ times in Liang's implementation of the Brown algorithm, accounting for over 40% of the runtime. Similar functions account for the vast majority of the runtime. The values in the arrays $\texttt{L2}, \texttt{q2}, \texttt{p2}, \texttt{p1}$ are precomputed. $\texttt{p2}[v][w] = p(v, w)$, i.e, the probability of cluster v being followed by cluster w; $\texttt{p1}[v] = p(v)$ is the probability of cluster v; $\texttt{q2}[v][w] = p(v, w) \log((p(v)p(w))^{-1} p(v, w))$ is the contribution of the mutual information between clusters v and w. The function recomputes $\texttt{L2}[v][w]$, which is the loss in log-likelihood if clusters v and w are merged. The function updates L2 after clusters s and t have been merged to form a new cluster u. There are many operations involved in this calculation: 6 divisions, 12 multiplications, 36 additions (26 additions and 10 subtractions), and 6 log operations. 97

5.7 Variant of Ward's algorithm from Section 5.4.4. 100

6.1 Visualization of the representational scheme under a Brown model with 2 hidden states. (a) Normalizing the original rows of O. (b) Normalizing the scaled and rotated rows of O. 107

6.2 A template for spectral word embedding methods. 112

7.1 Non-negative matrix factorization algorithm of Arora *et al.* (2012). 124

7.2 NMF-based learning algorithm for A-HMMs. The algorithm **Anchor-NMF** is given in Figure 7.1. 127

7.3 Algorithm for constructing a valid Ω with different construction methods. For simplicity, we only show the bigram construction (context size $L = 1$), but an extension for larger context ($L > 1$) is straightforward. 131

8.1 (a) An R-HMM chain. (b) An equivalent representation where labels and hidden states are intertwined. 145

8.2 The forward-backward algorithm (in matrix form) for an R-HMM. 149

8.3 Given an R-HMM sequence, we define random variables over observed quantities so that conditioning on the current node, (a) the future F_1 is independent of the past P and (b) the present R is independent of the density D. 153

8.4 Accuracy of the spectral algorithm and EM on TIMIT development data for varying numbers of hidden states m. For EM, the highest scoring iteration is shown. 158

8.5 The feature templates for phoneme recognition. The simplest features look only at the current label and observation. Other features indicate the previous phoneme type used before a_i (`pp`), the next phoneme type used after a_i (`np`), and the relative position (beginning, middle, or end) of a_i within the current phoneme (`pos`). The figure gives a typical segment of the phoneme sequence $a_1 \dots a_N$. 159

8.6 Feature ablation experiments on TIMIT development data for the best spectral model ($m = 24$) with comparisons to the best EM model ($m = 4$) and EM with $m = 24$. 160

8.7 Performance of baselines and spectral R-HMM on TIMIT test data. Number of hidden states m optimized on development data (see Figure 8.4). The improvement of the spectral method over the EM baseline is significant at the $p \leq 0.05$ level (and very close to significant at $p \leq 0.01$, with a precise value of $p \leq 0.0104$). 161

8.8 The spectral estimation algorithm for an R-HMM. 162

A.1 The $O(N + nm^2)$ clustering algorithm of Brown *et al.* [1992]. The variables are explained in the main text. 191

List of Tables

5.1 Performance gains in NER. 94

5.2 Mutual information computed as in Eq. (5.3) on the RCV1 corpus. 94

5.3 Speed and performance comparison with the Brown algorithm for different numbers of clusters and vocabulary sizes. In all the reported runtimes, we exclude the time to read and write data. We report the F1 scores on the NER dev set; for the spectral algorithm, we report the best scores. 101

6.1 Performance of CCA (1000 dimensions) on the development portion of data with different data transformation methods ($\alpha = 0.75$, $\beta = 0$). 114

6.2 Performance of various spectral methods on the development portion of data. 115

6.3 Performance of different word embedding methods on the test portion of data. See the main text for the configuration details of spectral methods. . 116

6.4 NER F1 scores when word embeddings are added as real-valued features to the baseline (—). For comparison, we also derive 1000 Brown clusters (BROWN) on the same vocabulary and use the resulting bit strings as features Brown *et al.* [1992]. 118

7.1 Numbers of word tokens and types across 10 languages in the universal treebank dataset (version 2.0). 134

7.2 Many-to-one accuracy on the English data with different choices of the convex hull Ω (Figure 7.3). These results do not use spelling features. 136

7.3 Many-to-one accuracy on each language using 12 universal tags. The first four models are HMMs estimated with the Baum-Welch algorithm (BW), the clustering algorithm of Brown *et al.* [1992], the anchor algorithm without (ANCHOR) and with (ANCHOR-FEAT) feature augmentation. LOG-LINEAR is the model of Berg-Kirkpatrick *et al.* [2010] trained with the direct-gradient method using L-BFGS. For BW and LOG-LINEAR, we report the mean and the standard deviation (in parentheses) of 10 random restarts run for 1,000 iterations. 137

7.4 Verifying model assumptions on the universal treebank. The anchor assumption is satisfied in every language. The Brown assumption (each word has exactly one possible tag) is violated but not by a large margin. The lower table shows the most frequent anchor word and its count under each tag on the English portion. 140

7.5 Log likelihood normalized by the number of words on English (along with accuracy). For BW, we report the mean of 10 random restarts run for 1,000 iterations. 140

7.6 Many-to-one accuracy on the English data with 45 original tags. We use the same setting as in Table 7.3. For BW and LOG-LINEAR, we report the mean and the standard deviation (in parentheses) of 10 random restarts run for 1,000 iterations. 141

7.7 Anchor words found in each language (model ANCHOR-FEAT). 142

7.8 Most likely words under each anchor word (English model ANCHOR-FEAT). Emission probabilities $o(x|h)$ are given in parentheses. 143

Acknowledgments

I am indebted to my advisor Michael Collins on so many levels. Most importantly, he shaped my identity as a researcher. I learned tremendously from his penetrating insight, bullet-proof rigor, and astonishing clarity in research. At the same time, I was able to pursue ideas on my own thanks to the right level of intellectual freedom he provided. If I achieved anything in the PhD program, it is due to his sage advising: all my shortcomings are due to my own limitation. I am also indebted to Daniel Hsu who was a good friend and an unofficial advisor to me during the program. The unfathomable depth of his knowledge consistently shocked me into new levels of research and is still what I strive for. The spectral methods developed in this thesis would not have been here if not for his kind and patient guidance. I also thank David Blei, Owen Rambow, and Slav Petrov for being on my dissertation committee.

I would like to thank my coauthors and collaborators all of who taught me so much. Special thanks go to Sham Kakade, Dean Foster, Lyle Ungar, and Shay Cohen for enlightening discussions on spectral methods. I would also like to thank Young-Bum Kim: I admire his passion and look forward to future collobaration. I was privileged to have summer internship opportunities with brilliant researchers. Sham Kakade and T. J. Hazen were the most gracious hosts at Microsoft Research New England in 2013. Slav Petrov and Emily Pitler were the most gracious hosts at Google New York in 2014. These internships expanded my horizon in research. I was also fortunate to participate in the summer workshop at the Johns Hopkins Center for Language and Speech Processing in 2011: I thank Alexander Berg, Tamara Berg, Hal Daumé III, Yejin Choi, Jesse Dodge, Amit Goyal, Xufeng Han, Alyssa Mensch, Margaret Mitchell, and Kota Yamaguchi for the productive work and great times together.

The NLP, speech, and machine learning groups at Columbia University were a wonderful

community and I would like to thank them collectively. I thank my lab members Avner May, Yin-Wen Chang, Mohammad Rasooli, Andrei Simion, and Sasha Rush for their friendship and many spontaneous discussions on NLP, machine learning, and life. I am especially grateful to Avner for learning together (chevruta): I deeply respect his down-to-earth attitude and incredible attention to details. My time at Columbia was also made pleasant by the friendship of Victor Soto (with whom I had many enjoyable discussions on movies and games), Or Biran, Chris Kedzie, Apoorv Agarwal, Kapil Thadani, Jessica Ouyang, Wei-Yun Ma, Cem Subakan, Chris Riederer, and many others sharing the workspace on the seventh floor of CEPSR.

Finally, I would like to thank my family in Korea. My parents provided critical support for my journey in the states in the last decade: they should take all credit for anything I achieved. And above all, I thank my brother Eung Sun Lee (이응선), who had been there for me from the very beginning.

Dedicated to my dad

Chapter 1

Introduction

1.1 Motivation

Many state-of-the-art results in natural language processing (NLP) are achieved with statistical models involving *latent* variables. This is a setting that arises in *unsupervised* and *semi-supervised* language processing tasks. For instance,

- Brown *et al.* [1992] use a variant of a hidden Markov model (HMM) to induce word clusters from unannotated text. These clusters are shown to be useful in a wide range of semi-supervised NLP tasks [Miller *et al.*, 2004; Koo *et al.*, 2008; Uszkoreit and Brants, 2008; Owoputi *et al.*, 2013].

- Matsuzaki *et al.* [2005] and Petrov *et al.* [2006] use a probablistic context-free grammar (PCFG) augmented with unobserved annotations. The resulting syntactic parser is much more accurate than a simple PCFG.

- Haghighi and Klein [2006] and Berg-Kirkpatrick *et al.* [2010] use feature-rich models with hidden variables and achieve excellent performance in a variety of label induction tasks in NLP.

- Brown *et al.* [1993] use latent-variable models to induce word alignments between translations. The so-called IBM models have been extremely influential in statistical machine translation.

Unfortunately, computational problems associated with such models (for instance, finding the optimal parameter values) are typically intractable. The difficulty is usually in the form of non-convex training objectives for which standard iterative optimization techniques, such as the expectation-maximization (EM) algorithm [Dempster *et al.*, 1977] or gradient-based search techniques, are not guaranteed to produce globally optimal solutions. In some cases, the difficulty is formalized by hardness results. For example, Terwijn [2002] shows that the maximum-likelihood estimation problem for HMMs is generally hard under cryptographic assumptions. Faced with this computational difficulty, practitioners are forced to rely on heuristic methods without strong guarantees; for instance, EM is only guaranteed to converge to a local optimum.

Nonetheless, heuristics are often sufficient for empirical purposes and it is tempting to overlook their lack of theoretical guarantees. But the de-emphasis on theoretical aspects has other negative ramifications. First, it can impede the development of rigorous theoretical understanding which can generate new ideas and algorithms. Second, it can lead to black art solutions that are unreliable and difficult to reproduce. This motivates a central question in this thesis:

> *Can we develop algorithms with provable guarantees for the challenging computational problems associated with latent-variable models in NLP?*

In this thesis, we give a positive answer to this question by turning to *spectral methods*. We broadly define a spectral method to be any algorithmic procedure that use singular value decomposition (SVD) or other similar matrix or tensor factorization. There has recently been a resurgence of interest in spectral methods as a principled framework for learning latent-variable models. The seminal work of Hsu *et al.* [2008] derives a spectral algorithm for learning HMMs with polynomial time and sample complexity, dodging the hardness result of Terwijn [2002] by exploiting a mild assumption with SVD. A number of other spectral algorithms have been proposed, including the tensor extension of Foster *et al.* [2012], the non-negative matrix factorization (NMF) method of Arora *et al.* [2012a], and the tensor decomposition method of Anandkumar *et al.* [2014]. This thesis aims to leverage and build on this recent progress in the context of NLP.

A high-level description of a spectral method is as follows:

Formulation We formulate a task into a suitable low-rank decomposition problem. The quantity to be decomposed is expected values of *observed* random variables (e.g., corresponding to words in unannotated text). Then we can trivially derive an empirical algorithm by equating theoretical moments with sample moments, that is, through the method of moments. The values for the rank is usually given by task-specific assumptions.

Decomposition We leverage powerful linear algebraic techniques (such as the SVD) to solve the low-rank decomposition problem.

Reconstruction We use the solution of the low-rank decomposition problem to reconstruct a solution to the original task.

An example makes this description concrete. Consider the task of finding a pair of vectors (a, b) that maximally correlate the scalar projections of (vector-valued) random variables (X, Y) with zero mean. This task is a special case of canonical correlation analysis (CCA) introduced by Hotelling [1936] and is useful for analyzing the linear relationship between X and Y. Using the Pearson correlation coefficient, we can formulate this as the following optimization problem:

$$(a, b) = \arg\max_{u,v} \frac{\boldsymbol{E}\left[\left(u^\top X\right)\left(v^\top Y\right)\right]}{\sqrt{\boldsymbol{E}\left[\left(u^\top X\right)^2\right]\boldsymbol{E}\left[\left(v^\top Y\right)^2\right]}}$$

where $\top$ denotes the transpose operation and $\boldsymbol{E}[Z]$ denotes the expected value of Z.[1] There are two unknowns u and v with nonlinear interactions, and it is unclear how to use a standard optimization technique to solve the problem. However, there is a spectral algorithm that gives a closed-form solution [Hotelling, 1936]:

1. **Formulation**: Define a matrix $\Omega = \boldsymbol{E}\left[XX^\top\right]^{-1/2}\boldsymbol{E}\left[XY^\top\right]\boldsymbol{E}\left[YY^\top\right]^{-1/2}$.
2. **Decomposition**: (Rank-1 SVD) Compute the left and right singular vectors (c, d) of Ω corresponding to the largest singular value.

[1] The solution is not unique. If (a, b) is a solution, so is $(\alpha a + \gamma, \beta b + \lambda)$ for any constants $\alpha, \beta, \gamma, \lambda \in \mathbb{R}$ where α and β are nonzero.

3. **Reconstruction**: Return $a = \boldsymbol{E}\left[XX^\top\right]^{-1/2} c$ and $b = \boldsymbol{E}\left[YY^\top\right]^{-1/2} d$.

Note that the algorithm requires the covariance matrices $\boldsymbol{E}\left[XX^\top\right]$ and $\boldsymbol{E}\left[YY^\top\right]$ to be invertible. This *non-degeneracy* requirement is common in spectral methods.

This thesis develops spectral methods for NLP with a focus on two tasks: learning lexical representations from unannotated text and estimating parameters of latent-variable models used in NLP applications.

1.2 Learning Lexical Representations

The first task considered in the thesis is inducing word representations that reflect distributional properties of words from an unlabeled corpus. These so-called distributional word representations are useful features in many downstream tasks in NLP such as part-of-speech (POS) tagging [Owoputi *et al.*, 2013; Schnabel and Schütze, 2014], named-entity recognition [Miller *et al.*, 2004; Dhillon *et al.*, 2011b], and syntactic parsing [Koo *et al.*, 2008; Chen and Manning, 2014]. They are also useful for certain lexical tasks such as measuring the similarity between words [Miller and Charles, 1991] and answering analogy questions [Mikolov *et al.*, 2013c].

More specifically, we consider two forms of lexical representations: hierarchical word clusters and word embeddings.

1.2.1 Hierarchical Word Clusters

Hierarchical word clusters are a form of lexical representations popularized by the clustering algorithm of Brown *et al.* [1992] which is now commonly called "Brown clustering". The Brown clustering algorithm performs greedy merging on a vocabulary to produce a binary tree over words such as the one shown in Figure 1.1(a). Each word is then represented as a bit string indicating the path from the root. We obtain clusters of different granularities by taking prefixes of the bit strings of different lengths. For the example in Figure 1.1(a), taking prefixes of length two yields four clusters: {`coffee`, `tea`}, {`dog`, `cat`}, {`walk`, `run`}, and {`walked`, `ran`}.

Brown clustering is a greedy heuristic to handle the associated computational difficulties

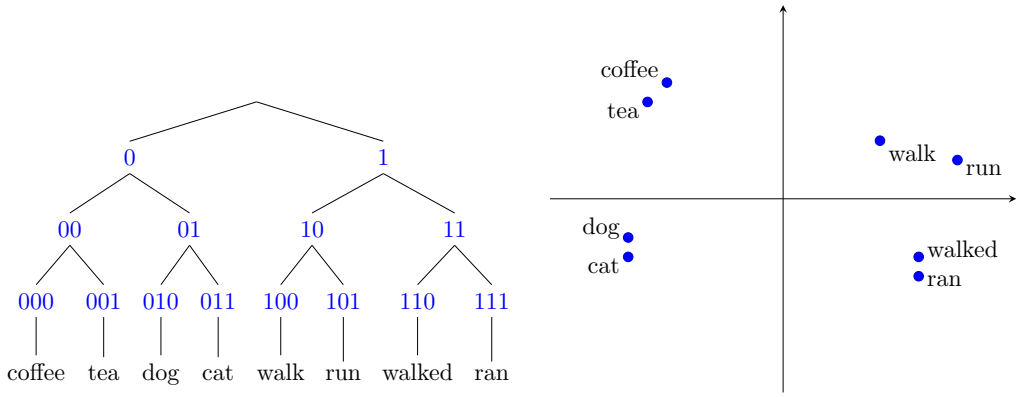


Figure 1.1: Two forms of lexical representations for the vocabulary: `coffee`, `tea`, `dog`, `cat`, `walk`, `run`, `walked`, `ran`. (a) Hierarchical word clusters. (b) Word embeddings.

underlying the clustering problem and hence has no theoretical guarantees. In Chapter 5, we develop a spectral algorithm which is the first provably correct algorithm for the clustering problem of Brown *et al.* [1992]. In practice, our algorithm is much more efficient than the Brown clustering algorithm and produces hierarchical word clusters of competitive quality.

1.2.2 Word Embeddings

Word embeddings refer to low-dimensional vector representations of words. Figure 1.1(b) shows a two-dimensional example visualized on a plane. The topology of words characterizes their relationship: for instance, the nearest neighbor of `dog` is `cat`.[2] There has been a surge of interest in word embeddings due to their ability to capture subtle syntactic and semantic patterns that are useful in NLP applications. Some successful existing methods for deriving word embeddings are the "word2vec" algorithm of Mikolov *et al.* [2013b], the "GloVe" algorithm of Pennington *et al.* [2014], and the "Eigenwords" algorithm of Dhillon *et al.* [2011a; 2012; 2015].

Many of these existing word embedding methods lack strong guarantees or involve steps

[2]Note that the two forms of lexical representations are closely related. We can represent clusters as vectors (e.g., indicator vectors) and convert word embeddings as hierarchical clusters using a distance-based clustering algorithm. We use this observation later in the thesis.

that are left unjustified. In Chapter 6, we develop a spectral algorithm using a novel model-based interpretation of CCA (which also underlies the method of Dhillon *et al.* [2015]). This algorithm gives a unified view of many spectral methods in the word embedding literature and yields state-of-the-art results on various lexical and semi-supervised tasks.

1.3 Estimating Parameters of Latent-Variable Models

The second task considered in the thesis is estimating parameters of latent-variable models used in NLP applications. When the unobserved variables themselves are the target labels, the task becomes unsupervised label induction. When the unobserved variables are not the target labels, these auxiliary variables are introduced to capture hidden correlation between observed variables and improve generalizability (e.g., see Matsuzaki *et al.* [2005]). We consider both settings in this thesis.

More specifically, we develop spectral algorithms for learning latent-variable models to tackle two NLP applications: unsupervised POS tagging and (supervised) phoneme recognition.

1.3.1 Unsupervised POS Tagging

In unsupervised POS tagging, we are given a set of unlabeled sentences and wish to tag each word with a label that faithfully reflects the word's POS tag. For instance, in the following two sentences,

`John has a light bag` `The light was blue`

a desirable tagging may look like

`John/1 has/2 a/3 light/4 bag/1` `The/3 light/1 was/2 blue/4`

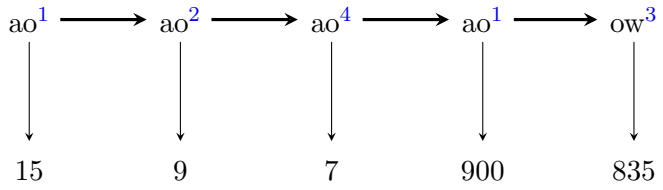
where the labels 1, 2, 3, and 4 respectively group words into nouns, verbs, determiners, and adjectives. Note that for the word `light`, we must disambiguate the adjective use from the noun use based on the context.

One approach is to learn an HMM with hidden states corresponding to (missing) POS tags. Learning HMMs is generally intractable [Terwijn, 2002]; furthermore, vanilla HMMs

are known to perform poorly in unsupervised POS tagging because of model misspecification. In Chapter 6, we design a certain family of restricted HMMs effective for POS tagging and develop a statistically consistent spectral learning algorithm. With the universal tagset, our method is competitive with the state-of-the-art.

1.3.2 Phoneme Recognition

In supervised phoneme recognition, we are given a training set of discretized speech signal sequences annotated with corresponding phoneme sequences and wish to learn a model to predict phonemes for unseen signals. A naive approach is to fit an HMM on the training set, but an HMM makes a strong assumption that each phoneme is independent of phonemes before the previous phoneme. To relax this assumption, we introduce auxilliary variables that further refine hidden states (in a similar manner Matsuzaki *et al.* [2005] introduce latent variables in probabilistic context-free grammars for syntactic parsing). We call this augmented HMM a refinement HMM (R-HMM). For instance, given a training instance (x, y) where $x = (15\ 9\ 7\ 900\ 835)$ is a sequence of signal classes and $y = (\text{ao ao ao ao ow})$ is the corresponding phoneme sequence, an R-HMM assumes an unobserved sequence of integers $z = (1\ 2\ 4\ 1\ 3)$ that refines y and defines the joint probability $p(x, y, z)$:



Estimating optimal parameters of an R-HMM is challenging due to the unobserved auxiliary variables. In Chapter 8, we develop a spectral algorithm which is guaranteed to find the true parameters under mild conditions on the singular values of the model. In experiments, R-HMMs perform significantly better than HMMs and the spectral algorithm is competitive with EM.

We argue that the spectral algorithms presented in this thesis have the following advantages over previous methods:

1. The algorithms provide a new theoretical framework that is amenable to rigorous analysis. In particular, they are shown to be statistically consistent: their outputs converge to the true underlying values given enough samples.

2. The algorithms are simple to implement, efficient, and scalable to large amounts of data. They also yield results that are competitive with the state-of-the-art.

1.4 Thesis Overview

The thesis is structured as follows. In Chapter 2, we give a broad overview of related works. In Part I, we provide a review of linear algebra (Chapter 3) and existing spectral techniques (Chapter 4). In Part II, we give spectral algorithms for learning lexical representations: hierarchical word clusters (Chapter 5) and word embeddings (Chapter 6). In Part III, we give spectral algorithms for estimating latent-variable models: for unsupervised POS tagging (Chapter 7) and for supervised phoneme recognition (Chapter 8). In Chapter 9, we provide concluding remarks and a discussion of future work.

1.5 Notation

We use the following notation throughout the thesis.

- $[n]$ denotes the set of positive integers $\{1, \ldots, n\}$.
- $[[\Gamma]]$ denotes the indicator of a predicate Γ, taking value 1 if Γ is true and 0 otherwise.
- $\boldsymbol{E}[X]$ denotes the expected value of a random variable X.
- $\top$ as a superscript denotes the transpose operation of a vector or a matrix.
- $\text{diag}(v)$ denotes the diagonal matrix of a vector $v = (v_1 \ldots v_n)$. That is,

$$\text{diag}(v) = \begin{bmatrix} v_1 & & \cdots & 0 \\ & v_2 & & \vdots \\ \vdots & & \ddots & \\ 0 & \cdots & & v_n \end{bmatrix}.$$

- $[v_1 \dots v_m]$ denotes a matrix whose i-th column is the column vector v_i.
- M^a denotes the matrix power of (square matrix) M by a scalar a. For the purposes of this thesis, it is sufficient to define $M^a = V \operatorname{diag}(\lambda_1^a \dots \lambda_n^a) V^\top$ where we always assume $M = V \operatorname{diag}(\lambda_1 \dots \lambda_n) V^\top$ is diagonalizable (see Section 3.2.2). Under this definition, $M^a = \prod_{i=1}^{a} M$ if a is a positive integer, M^{-1} is the inverse of M, and $M^{1/2} M^{1/2} = M$.
- $\sqrt{M}$ denotes the element-wise square-root of a matrix M, that is, $[\sqrt{M}]_{i,j} = \sqrt{M_{i,j}}$.
- M^+ denotes the Moore-Penrose pseudoinverse of a matrix M. See Section 4.1 for a description of the pseudoinverse.
- $I_{m \times m}$ denotes the $m \times m$ identity matrix.

Chapter 2

Related Work

This chapter gives a broad overview of the literature in natural language processing (NLP) and machine learning relavant to this thesis, covering the following topics:

1. The use of latent-variable models in NLP (Section 2.1)
2. Representation learning in NLP (Section 2.2)
3. Spectral techniques and their applications to learning latent-variable models (Section 2.3)

A more technical review of spectral techniques is provided in Chapter 4. Additionally, a chapter-specific discussion of related work is provided in each of Chapter 5, 6, 7, and 8.

2.1 Latent-Variable Models in NLP

Models with hidden variables are of enormous importance in NLP. Language processing tasks that involve learning these models range from part-of-speech (POS) tagging [Merialdo, 1994] to word clustering [Brown *et al.*, 1992], text classification [Nigam *et al.*, 1998], topic modeling [Blei *et al.*, 2003], syntactic parsing [Matsuzaki *et al.*, 2005], and machine translation [Brown *et al.*, 1993].

A highly influential algorithm for learning latent-variable models is the expectation-maximization (EM) algorithm. A general theory of EM is developed in the landmark work by Dempster *et al.* [1977] which shows that EM optimizes the likelihood objective from

incomplete data.[1] Two examples of EM for structured models particularly relavant to NLP are the forward-backward (or Baum-Welch) algorithm [Rabiner, 1989] for hidden Markov models (HMMs) [Baum *et al.*, 1967; Church, 1988] and the inside-outside algorithm [Lari and Young, 1990] for probabilistic context-free grammars (PCFGs) [Chomsky, 1956; Eddy and Durbin, 1994].

A common use of latent-variable models in NLP is inducing linguistic structure from unannotated text. For example, there is a large body of work on learning an HMM or its variant for unsupervised tagging [Merialdo, 1994; Johnson, 2007; Berg-Kirkpatrick *et al.*, 2010], tree models for unsupervised parsing [Carroll and Charniak, 1992; Pereira and Schabes, 1992; Brill, 1993; Klein and Manning, 2004], and word alignment models for machine translation [Brown *et al.*, 1993; Vogel *et al.*, 1996].

More recently, latent-variable models have been also used to improve performance in supervised tasks in NLP. An important work in this direction is the work by Matsuzaki *et al.* [2005] who propose introducing unobserved variables in a PCFG to relax the model's strong independence assumptions. The resulting model, called a latent-variable PCFG (L-PCFG), is trained with EM and has been shown to perform significantly better than a vanilla PCFG. A follow-up work by Petrov *et al.* [2006] introduces a split-and-merge enhancement in EM and achieves state-of-the-art parsing accuracy [Petrov and Klein, 2007; Petrov, 2010].

2.2 Representation Learning in NLP

There has been great interest in the NLP community in methods that derive linguistic representations from large quantities of unlabeled data [Brown *et al.*, 1992; Pereira *et al.*, 1993; Ando and Zhang, 2005; Turian *et al.*, 2010; Dhillon *et al.*, 2011b; Collobert *et al.*, 2011; Mikolov *et al.*, 2013b,c]. These representations can be used to improve accuracy on various NLP problems, or to give significant reductions in the number of training examples required for learning.

A classical example is the word clusters (so-called "Brown clusters") resulting from the

[1] The original proof of Dempster *et al.* [1977] contains an error and is corrected and strengthened by Wu [1983].

clustering algorithm of Brown *et al.* [1992] and its variants [Martin *et al.*, 1998]. These word clusters have been very useful in semi-supervised tasks in NLP, for instance, semi-supervised named-entity recognition (NER) [Miller *et al.*, 2004], dependency parsing [Koo *et al.*, 2008], translation [Uszkoreit and Brants, 2008], and POS tagging [Owoputi *et al.*, 2013].

Another form of lexical representation called word embeddings (meaning low-dimensional vector representations of words) has recently been a subject of intense focus in NLP research. Mikolov *et al.* [2013a] and Mikolov *et al.* [2013b] propose a series of language models inspired by neural networks whose parameters can be used as word embeddings. Their work is known as "word2vec" and has been very influential in many areas of representation learning [Zou *et al.*, 2013; Levy and Goldberg, 2014b; Arora *et al.*, 2015]. The "Eigenwords" algorithms by Dhillon *et al.* [2011a] and Dhillon *et al.* [2012] use various modifications of canonical correlation analysis (LR-MVL and two-step CCA) for deriving word embeddings. The "GloVe" algorithm by Pennington *et al.* [2014] performs a weighted low-rank factorization of log-transformed co-occurrence counts.

Efforts have been made in deriving representations for linguistic objects beyond individual words, primarily in the neural network community. Socher *et al.* [2010] use recursive neural networks to model sentences; the resulting vector representations have been useful in a number of tasks such as parsing [Socher *et al.*, 2013a, 2011] and sentiment analysis [Socher *et al.*, 2013b]. Other types of neural networks have been considered, such as convolutional neural networks [Kalchbrenner *et al.*, 2014] and models based on recurrent neural networks [Cho *et al.*, 2014].

2.3 Spectral Techniques

Algorithms that make use of spectral decomposition date far back. Perhaps the most prominent early examples are principal component analysis (PCA) [Pearson, 1901] and canonical correlation analysis (CCA) [Hotelling, 1936]. PCA computes orthogonal directions of maximal variance, and the solution is obtained from the eigenvectors of the covariance matrix. CCA computes projection operators that maximize the correlation between two random variables (with certain orthogonality constraints), and the solution is obtained

from the singular vectors of a linearly transformed covariance matrix.

Another important example of a spectral technique is spectral clustering: partitioning vertices in an undirected graph by matrix decomposition [Donath and Hoffman, 1973; Fiedler, 1973]. A spectral clustering algorithm typically proceeds by constructing a graph Laplacian matrix from the data and performing a standard clustering algorithm (e.g., k-means) on reduced-dimensional points that correspond to the top eigenvalues of the Laplacian. It can be seen as optimizing a nontrivial clustering objective [Von Luxburg, 2007].

A recent advance in spectral techniques is their use in learning latent-variable models. EM is a common approach for learning latent-variable models, but it generally has no guarantees of consistency or of sample complexity (e.g., within the PAC framework [Valiant, 1984]). This has led a number of researchers to consider alternatives to the EM algorithm, which do have PAC-style guarantees. For example, Dasgupta [1999] derive the first provably correct polynomial-time algorithm for learning Gaussian mixture models (GMMs). Vempala and Wang [2004] derive a spectral algorithm for learning GMMs.

One focus has been on spectral learning algorithms for HMMs and related models. This line of work started with the work of Hsu *et al.* [2008], who developed a spectral learning algorithm for HMMs which recovers an HMM's parameters, up to a linear transformation, using singular value decomposition and other simple matrix operations. The algorithm builds on the idea of observable operator models for HMMs due to Jaeger [2000]. Following the work of Hsu *et al.* [2008], spectral learning algorithms have been derived for a number of other models, including finite state transducers [Balle *et al.*, 2011]; split-head automaton grammars [Luque *et al.*, 2012]; reduced rank HMMs in linear dynamical systems [Siddiqi *et al.*, 2009]; kernel-based methods for HMMs [Song *et al.*, 2010]; tree graphical models [Song *et al.*, 2011b,a]; latent-variable probabilistic context-free grammars (L-PCFGs) [Cohen *et al.*, 2012]. There are also spectral learning algorithms for learning PCFGs in the unsupervised setting [Bailly *et al.*, 2013].

Anandkumar *et al.* [2012c] propose a general method of moments approach to estimating mixture models. This direction of research is further pursued in the tensor decomposition framework of Anandkumar *et al.* [2014], which has been used to recover the parameters of a variety of models such as latent Dirichlet allocation (LDA) [Anandkumar *et al.*, 2012b],

mixtures of linear regressions [Chaganty and Liang, 2013], and a more general class of latent-variable graphical models [Chaganty and Liang, 2014].

Other types of matrix factorization techniques (other than SVD) have also been used for recovering model parameters. Arora *et al.* [2012a] develop an exact NMF algorithm for recovering the parameters of topic models satisfying an "anchor" assumption: each topic has at least one word that can only appear under that topic. This NMF approach to learning is quite general and has been applied to learning other models, such as L-PCFGs [Cohen and Collins, 2014] and anchor HMMs described in Chapter 7 of the thesis.

Part I

The Spectral Framework

Chapter 3

A Review of Linear Algebra

3.1 Basic Concepts

In this section, we review basic concepts in linear algebra frequently invoked in spectral techniques.

3.1.1 Vector Spaces and Euclidean Space

A **vector space** $\mathcal{V}$ over a field $\mathcal{F}$ of scalars is a set of "vectors", entities with direction, closed under addition and scalar multiplication satisfying certain axioms. It can be endowed with an **inner product** $\langle \cdot, \cdot \rangle : \mathcal{V} \times \mathcal{V} \to \mathcal{F}$, which is a quantatative measure of the relationship between a pair of vectors (such as the angle). An inner product also induces a **norm** $||u|| = \sqrt{\langle u, u \rangle}$ which computes the magnitude of u. See Chapter 1.2 of Friedberg *et al.* [2003] for a formal definition of a vector space and Chapter 2 of Prugovečki [1971] for a formal definition of an inner product.

In subsequent sections, we focus on Euclidean space to illustrate key ideas associated with a vector space. The n-dimensional (real-valued) **Euclidean space** $\mathbb{R}^n$ is a vector space over $\mathbb{R}$. The **Euclidean inner product** $\langle \cdot, \cdot \rangle : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is defined as

$$\langle u, v \rangle := [u]_1[v]_1 + \cdots + [u]_n[v]_n \tag{3.1}$$

It is also called the **dot product** and written as $u \cdot v$. The standard vector multiplication notation $u^\top v$ is sometimes used to denote the inner product.

One use of the inner product is calculating the *length* (or *norm*) of a vector. By the Pythagorean theorem, the length of $u \in \mathbb{R}^n$ is given by $||u||_2 := \sqrt{[u]_1^2 + \ldots + [u]_n^2}$ and called the **Euclidean norm** of u. Note that it can be calculated as

$$||u||_2 = \sqrt{\langle u, u \rangle} \tag{3.2}$$

Another use of the inner product is calculating the *angle* θ between two nonzero vectors. This use is based on the following result.

Theorem 3.1.1. *For nonzero $u, v \in \mathbb{R}^n$ with angle θ, $\langle u, v \rangle = ||u||_2 \, ||v||_2 \cos\theta$.*

Proof. Let $w = u - v$ be the opposing side of θ. The law of cosines states that

$$||w||_2^2 = ||u||_2^2 + ||v||_2^2 - 2 \, ||u||_2 \, ||v||_2 \cos\theta$$

But since $||w||_2^2 = ||u||_2^2 + ||v||_2^2 - 2\langle u, v \rangle$, we conclude that $\langle u, v \rangle = ||u||_2 \, ||v||_2 \cos\theta$. □

The following corollaries are immediate from Theorem 3.1.1.

Corollary 3.1.2 (Orthogonality)**.** *Nonzero $u, v \in \mathbb{R}^n$ are orthogonal (i.e., their angle is $\theta = \pi/2$) iff $\langle u, v \rangle = 0$.*

Corollary 3.1.3 (Cauchy–Schwarz inequality)**.** $|\langle u, v \rangle| \leq ||u||_2 \, ||v||_2$ *for all $u, v \in \mathbb{R}^n$.*

3.1.2 Subspaces and Dimensions

A **subspace** S of $\mathbb{R}^n$ is a subset of $\mathbb{R}^n$ which is a vector space over $\mathbb{R}$ itself. A necessary and sufficient condition for $S \subseteq \mathbb{R}^n$ to be a subspace is the following (Theorem 1.3, Friedberg *et al.* [2003]):

1. $0 \in S$
2. $u + v \in S$ whenever $u, v \in S$
3. $au \in S$ whenever $a \in \mathbb{R}$ and $u \in S$

The condition implies that a subspace is always a "flat" (or linear) space passing through the origin, such as infinite lines and planes (or the trivial subspace $\{0\}$).

A set of vectors $u_1 \dots u_m \in \mathbb{R}^n$ are called **linearly dependent** if there exist $a_1 \dots a_m \in \mathbb{R}$ that are not all zero such that $au_1 + \cdots au_m = 0$. They are **linearly independent** if they are not linearly dependent. The **dimension** $\dim(S)$ of a subspace $S \subseteq \mathbb{R}^n$ is the number of linearly independent vectors in S.

The **span** of $u_1 \dots u_m \in \mathbb{R}^n$ is defined to be all their linear combinations:

$$\text{span}\{u_1 \dots u_m\} := \left\{ \sum_{i=1}^{m} a_i u_i \;\middle|\; a_i \in \mathbb{R} \right\} \tag{3.3}$$

which can be shown to be the smallest subspace of $\mathbb{R}^n$ containing $u_1 \dots u_m$ (Theorem 1.5, Friedberg *et al.* [2003]).

The **basis** of a subspace $S \subseteq \mathbb{R}^n$ of dimension m is a set of linearly independent vectors $u_1 \dots u_m \in \mathbb{R}^n$ such that

$$S = \text{span}\{u_1 \dots u_m\} \tag{3.4}$$

In particular, $u_1 \dots u_m$ are called an **orthonormal basis** of S when they are orthogonal and have length $||u_i||_2 = 1$. We frequently parametrize an orthonormal basis as an orthonormal matrix $U = [u_1 \dots u_m] \in \mathbb{R}^{n \times m}$ ($U^\top U = I_{m \times m}$).

Finally, given a subspace $S \subseteq \mathbb{R}^n$ of dimension $m \leq n$, the corresponding **orthogonal complement** $S^\perp \subseteq \mathbb{R}^n$ is defined as

$$S^\perp := \{u \in \mathbb{R}^n : \; u^\top v = 0 \;\; \forall v \in S\}$$

It is easy to verify that the three subspace conditions hold, thus $S^\perp$ is a subspace of $\mathbb{R}^n$. Furthermore, we always have $\dim(S) + \dim(S^\perp) = n$ (see Theorem 1.5, Friedberg *et al.* [2003]), thus $\dim(S^\perp) = n - m$.

3.1.3 Matrices

A matrix $A \in \mathbb{R}^{m \times n}$ defines a linear transformation from $\mathbb{R}^n$ to $\mathbb{R}^m$. Given $u \in \mathbb{R}^n$, the transformation $v = Au \in \mathbb{R}^m$ can be thought of as either a linear combination of the columns $c_1 \dots c_n \in \mathbb{R}^m$ of A, or dot products between the rows $r_1 \dots r_m \in \mathbb{R}^n$ of A and u:

$$v = [u]_1 c_1 + \cdots + [u]_n c_n = \begin{bmatrix} r_1^\top u \\ \vdots \\ r_m^\top u \end{bmatrix} \tag{3.5}$$

The **range** (or the **column space**) of A is defined as the span of the columns of A; the **row space** of A is the column space of $A^\top$. The **null space** of A is defined as the set of vectors $u \in \mathbb{R}^n$ such that $Au = 0$; the **left null space** of A is the null space of $A^\top$. We denote them respectively by the following symbols:

$$\text{range}(A) = \text{col}(A) := \{Au : \ u \in \mathbb{R}^n\} \subseteq \mathbb{R}^m \tag{3.6}$$

$$\text{row}(A) := \text{col}(A^\top) \subseteq \mathbb{R}^n \tag{3.7}$$

$$\text{null}(A) := \{u \in \mathbb{R}^n : \ Au = 0\} \subseteq \mathbb{R}^n \tag{3.8}$$

$$\text{left-null}(A) := \text{null}(A^\top) \subseteq \mathbb{R}^m \tag{3.9}$$

It can be shown that they are all subspaces (Theorem 2.1, Friedberg *et al.* [2003]). Observe that $\text{null}(A) = \text{row}(A)^\perp$ and $\text{left-null}(A) = \text{range}(A)^\perp$. In Section 3.3, we show that singular value decomposition can be used to find an orthonormal basis of each of these subspaces.

The **rank** of A is defined as the dimension of the range of A, which is the number of linearly independent columns of A:

$$\text{rank}(A) := \dim(\text{range}(A)) \tag{3.10}$$

An important use of the rank is testing the invertibility of a square matrix: $A \in \mathbb{R}^{n \times n}$ is invertible iff $\text{rank}(A) = n$ (see p. 152 of Friedberg *et al.* [2003]). The **nullity** of A is the dimension of the null space of A, $\text{nullity}(A) := \dim(\text{null}(A))$.

The following theorems are fundamental results in linear algebra:

Theorem 3.1.4 (Rank-nullity theorem)**.** *Let $A \in \mathbb{R}^{m \times n}$. Then*

$$\text{rank}(A) + \text{nullity}(A) = n$$

Proof. See p. 70 of Friedberg *et al.* [2003]. □

Theorem 3.1.5. *Let $A \in \mathbb{R}^{m \times n}$. Then*

$$\dim(\text{col}(A)) = \dim(\text{row}(A))$$

Proof. See p. 158 of Friedberg *et al.* [2003]. □

Theorem 3.1.5 shows that $\text{rank}(A)$ is also the number of linearly independent rows. Furthermore, the rank-nullity theorem implies that if $r = \text{rank}(A)$,

$$\begin{aligned} \text{rank}(A) = \dim(\text{col}(A)) = \dim(\text{row}(A)) = r \\ \dim(\text{null}(A)) = n - r \\ \dim(\text{left-null}(A)) = m - r \end{aligned}$$

We define additional quantities associated with a matrix. The **trace** of a square matrix $A \in \mathbb{R}^{n \times n}$ is defined as the sum of its diagonal entries:

$$\text{Tr}(A) := [A]_{1,1} + \cdots + [A]_{n,n} \tag{3.11}$$

The **Frobenius norm** $||A||_F$ of a matrix $A \in \mathbb{R}^{m \times n}$ is defined as:

$$||A||_F := \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |[A]_{i,j}|^2} = \sqrt{\text{Tr}(A^\top A)} = \sqrt{\text{Tr}(AA^\top)} \tag{3.12}$$

where the trace expression can be easily verified. The relationship between the trace and eigenvalues (3.23) implies that $||A||_F^2$ is the sum of the singular values of A. The **spectral norm** or the **operator norm** $||A||_2$ of a matrix $A \in \mathbb{R}^{m \times n}$ is defined as the maximizer of $||Ax||_2$ over the unit sphere,

$$||A||_2 := \max_{u \in \mathbb{R}^n:\ ||u||_2 = 1} ||Au||_2 = \max_{u \in \mathbb{R}^n:\ u \neq 0} \frac{||Au||_2}{||u||_2} \tag{3.13}$$

The variational characterization of eigenvalues (Theorem 3.2.7) implies that $||A||_2$ is the largest singular value of A. Note that $||Au||_2 \leq ||A||_2 \, ||u||_2$ for any $u \in \mathbb{R}^n$: this matrix-vector inequality is often useful.

An important property of $||\cdot||_F$ and $||\cdot||_2$ is their orthogonal invariance:

Proposition 3.1.1. *Let $A \in \mathbb{R}^{m \times n}$. Then*

$$||A||_F = ||QAR||_F \qquad\qquad ||A||_2 = ||QAR||_2$$

where $Q \in \mathbb{R}^{m \times m}$ and $R \in \mathbb{R}^{n \times n}$ are any orthogonal matrices (see Section 3.1.4).

Proof. Let $A = U\Sigma V^\top$ be an SVD of A. Then $QAR = (QU)\Sigma(R^\top V)^\top$ is an SVD of QAR since QU and $R^\top V$ have orthonormal columns. Thus A and QAR have the same set of singular values. Since $||\cdot||_F$ is the sum of singular values and $||\cdot||_2$ is the maximum singular value, the statement follows. □

3.1.4 Orthogonal Matrices

A square matrix $Q \in \mathbb{R}^{n\times n}$ is an **orthogonal matrix** if $Q^\top Q = I_{n\times n}$. In other words, the columns of Q are an orthonormal basis of $\mathbb{R}^n$; it follows that $QQ^\top = I_{n\times n}$ since $QQ^\top$ is an identity operator over $\mathbb{R}^n$ (see Section 3.1.5). Two important properties of Q are the following:

1. For any $u \in \mathbb{R}^n$, Qu has the same length as u:

$$||Qu||_2 = \sqrt{u^\top Q^\top Qu} = \sqrt{u^\top u} = ||u||_2$$

2. For any nonzero $u, v \in \mathbb{R}^n$, the angle $\theta_1 \in [0, \pi]$ between Qu and Qv and $\theta_2 \in [0, \pi]$ between u and v are the same. To see this, note that

$$||Qu||_2 \, ||Qv||_2 \cos\theta_1 = \langle Qu, Qv\rangle = u^\top Q^\top Qv = \langle u, v\rangle = ||u||_2 \, ||v||_2 \cos(\theta_2)$$

   It follows that $\cos\theta_1 = \cos\theta_2$ and thus $\theta_1 = \theta_2$ (since θ_1, θ_2 are taken in $[0, \pi]$).

Hence an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ can be seen as a *rotation* of the coordinates in $\mathbb{R}^n$.[1] This idea is used in Chapter 5 and 6.

3.1.5 Orthogonal Projection onto a Subspace

Theorem 3.1.6. *Let $S \subseteq \mathbb{R}^n$ be a subspace spanned by an orthonormal basis $u_1 \dots u_m \in \mathbb{R}^n$. Let $U := [u_1 \dots u_m] \in \mathbb{R}^{n\times m}$. Pick any $x \in \mathbb{R}^n$ and define*

$$y^* := \arg\min_{y\in S} ||x - y||_2 \tag{3.14}$$

Then the unique solution is given by $y^* = UU^\top x$.

Proof. Any element $y \in S$ is given by Uv for some $v \in \mathbb{R}^n$, thus $y^* = Uv^*$ where

$$v* = \arg\min_{v\in\mathbb{R}^n} ||x - Uv||_2 = (U^\top U)^{-1} U^\top x = U^\top x$$

is unique, hence y^* is unique. □

[1] Certain orthogonal matrices also represent *reflection*. For instance, the orthogonal matrix

$$Q = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

is a reflection in $\mathbb{R}^2$ (along the diagonal line that forms an angle of $\pi/4$ with the x-axis).

In Theorem 3.1.6, $x - y^*$ is orthogonal to the subspace $S = \text{span}\{u_1 \dots u_m\}$ since

$$\langle x - y^*, u_i \rangle = x^\top u_i - x^\top U U^\top u_i = 0 \qquad \forall i \in [m] \tag{3.15}$$

For this reason, the $n \times n$ matrix $\Pi := UU^\top$ is called the **orthogonal projection** onto the subspace $S \subseteq \mathbb{R}^n$. A few remarks on Π:

1. Π is unique. If Π' is another orthogonal projection onto S, then $\Pi x = \Pi' x$ for all $x \in \mathbb{R}^n$ (since this is uniquely given, Theorem 3.1.6). Hence $\Pi = \Pi'$.

2. Π is an identitiy operator for elements in S. This implies that the inherent dimension of $x \in S$ is m (not n) in the sense that the m-dimensional vector

   $$\tilde{x} := U^\top x$$

   can be restored to $x = U\tilde{x} \in \mathbb{R}^n$ without any loss of accuracy. This idea is used in subspace identification techniques (Section 4.7).

It is often of interest to compute the orthogonal projection $\Pi \in \mathbb{R}^{n \times n}$ onto the range of $A \in \mathbb{R}^{n \times m}$. If A already has orthonormal columns, the projection is given by $\Pi = AA^\top$. Otherwise, a convenient construction is given by

$$\Pi = A(A^\top A)^+ A^\top \tag{3.16}$$

To see this, let $A = U\Sigma V^\top$ be a rank-m SVD of A so that the columns of $U \in \mathbb{R}^{n \times m}$ are an orthonormal basis of range(A). Then

$$A(A^\top A)^+ A^\top = (U\Sigma V^\top)(V\Sigma^{-2}V^\top)(V\Sigma U^\top) = UU^\top \tag{3.17}$$

3.1.6 Gram-Schmidt Process and QR Decomposition

An application of the orthogonal projection yields a very useful technique in linear algebra called the **Gram-Schmidt process** (Figure 3.1).

Theorem 3.1.7. *Let $v_1 \dots v_m \in \mathbb{R}^n$ be linearly independent vectors. The output $\bar{v}_1 \dots \bar{v}_m \in \mathbb{R}^n$ of **Gram-Schmidt**$(v_1 \dots v_m)$ are orthonormal and satisfy*

$$\text{span}\{\bar{v}_1 \dots \bar{v}_i\} = \text{span}\{v_1 \dots v_i\} \qquad \forall 1 \leq i \leq m$$

Gram-Schmidt$(v_1 \dots v_m)$

Input: linearly independent $m \leq n$ vectors $v_1 \dots v_m \in \mathbb{R}^n$

1. Normalize $\bar{v}_1 = v_1 / ||v_1||_2$.
2. For $i = 2 \dots m$,
   (a) Remove the components of v_i lying in the span of $\bar{v}_1 \dots \bar{v}_{i-1}$,
   $$\bar{w}_i = v_i - [\bar{v}_1 \dots \bar{v}_{i-1}][\bar{v}_1 \dots \bar{v}_{i-1}]^\top v_i = v_i - \sum_{j=1}^{i-1} (\bar{v}_j^\top v_i)\bar{v}_j$$
   (b) Normalize $\bar{v}_i = \bar{w}_i / ||\bar{w}_i||_2$.

Output: orthonormal $\bar{v}_1 \dots \bar{v}_m \in \mathbb{R}^n$ such that $\text{span}\{\bar{v}_1 \dots \bar{v}_i\} = \text{span}\{v_1 \dots v_i\}$ for all $i = 1 \dots m$

Figure 3.1: The Gram-Schmidt process.

Proof. The base case $i = 1$ can be trivially verified. Assume $\text{span}\{\bar{v}_1 \dots \bar{v}_{i-1}\}$ equals $\text{span}\{v_1 \dots v_{i-1}\}$ and consider the vector $\bar{v}_i$ computed in the algorithm. It is orthogonal to the subspace $\text{span}\{\bar{v}_1 \dots \bar{v}_{i-1}\}$ by (3.15) and has length 1 by the normalization step, so $\bar{v}_1 \dots \bar{v}_i$ are orthonormal. Furthermore,

$$v_i = (\bar{v}_1^\top v_i)\bar{v}_1 + \dots + (\bar{v}_{i-1}^\top v_i)\bar{v}_{i-1} + ||\bar{w}_i||_2 \, \bar{v}_i$$

is in $\text{span}\{\bar{v}_1 \dots \bar{v}_i\}$, thus $\text{span}\{\bar{v}_1 \dots \bar{v}_i\} = \text{span}\{v_1 \dots v_i\}$. □

The Gram-Schmidt process yields one of the most elementary matrix decomposition techniques called **QR decomposition**. A simplified version (which assumes only matrices with linearly independent columns) is given in Figure 3.1.

Theorem 3.1.8. *Let $A \in \mathbb{R}^{n \times m}$ be a matrix with linearly independent columns $a_1 \dots a_m \in \mathbb{R}^n$. The output (Q, R) of $\boldsymbol{QR}(A)$ are an orthonormal matrix $Q \in \mathbb{R}^{n \times m}$ and an upper triangular matrix $R \in \mathbb{R}^{m \times m}$ such that $A = QR$.*

QR(A)

Input: $A \in \mathbb{R}^{n \times m}$ with linearly independent columns $a_1 \dots a_m \in \mathbb{R}^n$

1. $Q := [\bar{a}_1 \dots \bar{a}_m] \leftarrow$ **Gram-Schmidt**$(a_1 \dots a_m)$
2. Define an upper triangular matrix $R \in \mathbb{R}^{m \times m}$ by

$$[R]_{i,j} \leftarrow \bar{a}_i^\top a_j \qquad \forall i \in [1, m],\ j \in [i, m]$$

Output: orthonormal matrix $Q \in \mathbb{R}^{n \times m}$ and an upper triangular matrix $R \in \mathbb{R}^{m \times m}$ such that $A = QR$.

Figure 3.2: QR decomposition.

Proof. The columns $\bar{a}_1 \dots \bar{a}_m$ of Q are orthonormal by Theorem 3.1.7 and R is upper triangular by construction. The i-th column of QR is given by

$$(\bar{a}_1^\top \bar{a}_1)a_i + \dots + (\bar{a}_i^\top \bar{a}_i)a_i = [\bar{a}_1 \dots \bar{a}_i][\bar{a}_1 \dots \bar{a}_i]^\top a_i = a_i$$

since $a_i \in \text{span}\{\bar{a}_1 \dots \bar{a}_i\}$. □

The Gram-Schmidt process is also used in the non-negative matrix factorization algorithm of Arora *et al.* [2012a] (which is given in Figure 7.1 in this thesis).

3.2 Eigendecomposition

In this section, we develop a critical concept associated with a matrix called eigenvectors and eigenvalues. This concept leads to decomposition of a certain class of matrices called *eigendecomposition*. All statements (when not proven) can be found in standard introductory textbooks on linear algebra such as Strang [2009].

3.2.1 Square Matrices

Let $A \in \mathbb{R}^{n \times n}$ be a real square matrix. An **eigenvector** v of A is a nonzero vector that preserves its direction in $\mathbb{R}^n$ under the linear transformation defined by A: that is, for some

scalar λ,

$$Av = \lambda v \tag{3.18}$$

The scalar λ is called the **eigenvalue** corresponding to v. A useful fact (used in the proof of Theorem 3.2.3) is that eigenvectors corresponding to different eigenvalues are linearly independent.

Lemma 3.2.1. *Eigenvectors (v, v') of $A \in \mathbb{R}^{n\times n}$ corresponding to distinct eigenvalues (λ, λ') are linearly independent.*

Proof. Suppose $v' = cv$ for some scalar c (which must be nonzero). Then the eigen conditions imply that $Av' = A(cv) = c\lambda v$ and also $Av' = \lambda' v' = c\lambda' v$. Hence $\lambda v = \lambda' v$. Since $\lambda \neq \lambda'$, we must have $v = 0$. This contradicts the definition of an eigenvector. □

Theorem 3.2.2. *Let $A \in \mathbb{R}^{n\times n}$. The following statements are equivalent:*

- *λ is an eigenvalue of A.*
- *λ is a scalar that yields $\det(A - \lambda I_{n\times n}) = 0$.*

Proof. λ is an eigenvalue of A iff there is some nonzero vector v such that $Av = \lambda v$, and

$$\begin{aligned}
\exists v \neq 0 :\ (A - \lambda I_{n\times n})v = 0 \quad &\iff \quad \text{nullity}(A - \lambda I_{n\times n}) > 0 \\
&\iff \quad \text{rank}(A - \lambda I_{n\times n}) < n \qquad \text{(by the rank-nullity theorem)} \\
&\iff \quad A - \lambda I_{n\times n} \text{ is not invertible}
\end{aligned}$$

The last statement is equivalent to $\det(A - \lambda I_{n\times n}) = 0$. □

Since $\det(A - \lambda I_{n\times n})$ is a degree n polynomial in λ, it has n roots (counted with multiplicity[2]) by the fundamental theorem of algebra and can be written as

$$\det(A - \lambda I_{n\times n}) = (\lambda - \lambda_1)(\lambda - \lambda_2)\cdots(\lambda - \lambda_n) \tag{3.19}$$

[2]Recall that λ is a root of multiplicity k for a polynomial $p(x)$ if $p(x) = (x-\lambda)^k s(x)$ for some polynomial $s(x) \neq 0$.

Let λ be a distinct root of (3.19) and $a(\lambda)$ its multiplicity. Theorem 3.2.2 implies that λ is a distinct eigenvalue of A with a space of corresponding eigenvectors

$$E_{A,\lambda} := \{v : \ Av = \lambda v\} \tag{3.20}$$

(i.e., the null space of $A - \lambda I_{n\times n}$ and hence a subspace) which is called the **eigenspace** of A associated with λ. The dimension of this space is the number of linearly independent eigenvectors corresponding to λ. It can be shown that

$$1 \leq \dim(E_{A,\lambda}) \leq a(\lambda)$$

where the first inequality follows by the definition of λ (i.e., there is a corresponding eigenvector). We omit the proof of the second inequality.

Theorem 3.2.3. *Let $A \in \mathbb{R}^{n\times n}$ be a matrix with eigenvalues $\lambda_1 \dots \lambda_n$. The following statements are equivalent:*

- *There exist eigenvectors $v_1 \dots v_n$ corresponding to $\lambda_1 \dots \lambda_n$ such that*

$$A = V\Lambda V^{-1} \tag{3.21}$$

  *where $V = [v_1 \dots v_n]$ and $\Lambda = \operatorname{diag}(\lambda_1 \dots \lambda_n)$. (3.21) is called an **eigendecomposition** of A.*

- *The eigenspace of A associated with each distinct eigenvalue λ has the maximum dimension, that is, $\dim(E_{A,\lambda}) = a(\lambda)$.*

Proof. For any eigenvectors $v_1 \dots v_n$ corresponding to $\lambda_1 \dots \lambda_n$, we have

$$AV = V\Lambda \tag{3.22}$$

Thus it is sufficient to show that the existence of an invertible V is equivalent to the second statement. This is achieved by observing that we can find n linearly independent eigenvectors iff we can find $a(\lambda)$ linearly independent eigenvectors for each distinct eigenvalue λ (since eigenvectors corresponding to different eigenvalues are already linearly independent by Lemma 3.2.1). □

Theorem 3.2.3 gives the condition on a square matrix to have an eigendecomposition (i.e., each eigenspace must have the maximum dimension). A simple corollary is the following:

Corollary 3.2.4. *If $A \in \mathbb{R}^{n\times n}$ has n distinct eigenvalues $\lambda_1 \dots \lambda_n$, it has an eigendecomposition.*

Proof. Since $1 \leq \dim(E_{A,\lambda_i}) \leq a(\lambda_i) = 1$ for each (distinct) eigenvalue λ_i, the statement follows from Theorem 3.2.3. □

Since we can write an eigendecomposition of A as

$$V^{-1}AV = \Lambda$$

where Λ is a diagonal matrix, a matrix that has an eigendecomposition is called **diagonalizable**.[3] Lastly, a frequently used fact about eigenvalues $\lambda_1 \dots \lambda_n$ of $A \in \mathbb{R}^{n\times n}$ is the following (proof omitted):

$$\mathrm{Tr}(A) = \lambda_1 + \cdots + \lambda_n \tag{3.23}$$

3.2.2 Symmetric Matrices

A square matrix $A \in \mathbb{R}^{n\times n}$ always has eigenvalues but not necessarily an eigendecomposition. Fortunately, if A is additionally *symmetric*, A is guaranteed to have an eigendecomposition of a convenient form.

Lemma 3.2.5. *Let $A \in \mathbb{R}^{n\times n}$. If A is symmetric, then*

1. *All eigenvalues of A are real.*
2. *A is diagonalizable.*
3. *Eigenvectors corresponding to distinct eigenvalues are orthogonal.*

[3] While not every square matrix $A \in \mathbb{R}^{n\times n}$ is diagonalizable, it can be transformed into an *upper triangular form* $T = U^\top AU$ by an orthogonal matrix $U \in \mathbb{R}^{n\times n}$; see Theorem 3.3 of Stewart and Sun [1990]. This implies a decomposition $A = UTU^\top$ known the **Schur decomposition**. A can also always be transformed into a *block diagonal form* called a **Jordan canonical form**; see Theorem 3.7 of Stewart and Sun [1990].

Proof. For the first and second statements, we refer to Strang [2009]. For the last statement, let (v, v') be eigenvectors of A corresponding to distinct eigenvalues (λ, λ'). Then

$$\lambda v^\top v' = v^\top A^\top v' = v^\top A v' = \lambda' v^\top v'$$

Thus $v^\top v' = 0$ since $\lambda \neq \lambda'$. □

Theorem 3.2.6. *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix with eigenvalues $\lambda_1 \dots \lambda_n \in \mathbb{R}$. Then there exist orthonormal eigenvectors $v_1 \dots v_n \in \mathbb{R}^d$ of A corresponding to $\lambda_1 \dots \lambda_n$. In particular,*

$$A = V \Lambda V^\top \tag{3.24}$$

for orthogonal matrix $V = [v_1 \dots v_n] \in \mathbb{R}^{n\times n}$ and $\Lambda = \mathrm{diag}(\lambda_1 \dots \lambda_n) \in \mathbb{R}^{n\times n}$.

Proof. Since A is diagonalizable (Lemma 3.2.5), the eigenspace of λ_i has dimension $a(\lambda_i)$ (Theorem 3.2.3). Since this is the null space of a real matrix $A - \lambda_i I_{n\times n}$, it has $a(\lambda_i)$ orthonormal basis vectors in $\mathbb{R}^n$. The claim follows from the fact that the eigenspaces of distinct eigenvalues are orthogonal (Lemma 3.2.5). □

Another useful fact about the eigenvalues of a symmetric matrix is the following.

Proposition 3.2.1. *If $A \in \mathbb{R}^{n\times n}$ is symmetric, the rank of A is the number of nonzero eigenvalues.*

Proof. The dimension of $E_{A,0}$ is the multiplicity of the eigenvalue 0 by Lemma 3.2.5 and Theorem 3.2.3. The rank-nullity theorem gives $\mathrm{rank}(A) = n - \mathrm{nullity}(A) = n - a(0)$. □

Note that (3.24) can be equivalently written as a sum of weighted outer products

$$A = \sum_{i=1}^{n} \lambda_i v_i v_i^\top = \sum_{\lambda_i \neq 0} \lambda_i v_i v_i^\top \tag{3.25}$$

3.2.3 Variational Characterization

In Section 3.2.2, we see that any symmetric matrix has an eigendecomposition with real eigenvalues and orthonormal eigenvectors. It is possible to frame this decomposition as a *constrained optimization problem* (i.e., variational characterization).

Theorem 3.2.7. *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix with orthonormal eigenvectors $v_1 \dots v_n \in \mathbb{R}^n$ corresponding to its eigenvalues $\lambda_1 \geq \dots \geq \lambda_n \in \mathbb{R}$. Let $k \leq n$. Consider maximizing $v^\top Av$ over unit-length vectors $v \in \mathbb{R}^n$ under orthogonality constraints:*

$$v_i^* = \underset{\substack{v\in\mathbb{R}^n: \\ ||v||_2=1 \\ v^\top v_j^*=0\ \forall j<i}}{\arg\max} \quad v^\top Av \qquad\qquad \text{for } i = 1 \dots k$$

Then an optimal solution is given by $v_i^* = v_i$.[4]

Proof. The Lagrangian for the objective for v_1^* is:

$$L(v, \bar{\lambda}) = v^\top Av - \bar{\lambda}(v^\top v - 1)$$

Its stationary conditions $v^\top v = 1$ and $Av = \bar{\lambda}v$ imply that v_1^* is a unit-length eigenvector of A with eigenvalue $\bar{\lambda}$. Pre-multiplying the second condition by $v^\top$ and using the first condition, we have $\bar{\lambda} = v^\top Av$. Since this is the objective to maximize and has to be an eigenvalue of A, we must have $\bar{\lambda} = \lambda_1$. Thus any unit-length eigenvector in E_{A,λ_1} is an optimal solution for v_1^*, in particular v_1.

Suppose $v_i^* = v_i$ for $i < t \leq k$. The Lagrangian for the objective for v_t^* is:

$$L(v, \bar{\lambda}, \{\gamma_i\}_{i<t}) = v^\top Av - \bar{\lambda}(v^\top v - 1) - \sum_{i=1}^{t-1} \gamma_i v^\top v_i$$

Its stationary conditions are $v^\top v = 1$, $v_i^\top v = 0$ for $i < t$, and $Av = \bar{\lambda}v + (1/2)\sum_{i=1}^{t-1} \gamma_i v_i$. Pre-multiplying the last condition by $v_j^\top$ and using $Av_j = \lambda_j v_j$, we have $\gamma_j = 0$ for each $j < t$. Then $Av = \bar{\lambda}v$, so any stationary point is a unit-length eigenvector of A orthogonal to $v_1 \dots v_{t-1}$ corresponding to an eigenvalue $\bar{\lambda} = v^\top Av$. This implies that v_t is an optimal solution for v_t^*. □

Note that in Theorem 3.2.7,

$$\lambda_1 = \max_{v:\ ||v||_2=1} v^\top Av = \max_{v\neq 0} \left(\frac{v}{\sqrt{v^\top v}}\right)^\top A \left(\frac{v}{\sqrt{v^\top v}}\right) = \max_{v\neq 0} \frac{v^\top Av}{v^\top v}$$

The quantity in the last expression is called the **Rayleigh quotient**,

$$R(A, v) := \frac{v^\top Av}{v^\top v} \tag{3.26}$$

[4]Note that this solution also satisfies $(v_i^*)^\top Av_j^* = 0$ for all $i \neq j$ (even though this is not a constraint).

Thus the optimization problem can be seen as maximizing $R(A, v)$ over $v \neq 0$ (under orthogonality constraints):

$$v_i^* = \underset{\substack{v \in \mathbb{R}^n: \\ v \neq 0 \\ v^\top v_j^* = 0 \ \forall j<i}}{\arg\max} \frac{v^\top A v}{v^\top v} \qquad \text{for } i = 1 \dots k$$

Another useful characterization in matrix form is the following:

Theorem 3.2.8. *Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix with orthonormal eigenvectors $v_1 \dots v_n \in \mathbb{R}^d$ corresponding to its eigenvalues $\lambda_1 \geq \dots \geq \lambda_n \in \mathbb{R}$. Let $k \leq n$. Consider maximizing the trace of $\overline{V}^\top A \overline{V} \in \mathbb{R}^{k \times k}$ over orthonormal matrices $\overline{V} \in \mathbb{R}^{n \times k}$:*

$$V^* = \underset{\overline{V} \in \mathbb{R}^{n \times k}: \, \overline{V}^\top \overline{V} = I_{k \times k}}{\arg\max} \operatorname{Tr}(\overline{V}^\top A \overline{V})$$

Then an optimal solution is given by $V^* = [v_1 \dots v_k]$.

Proof. Denote the columns of $\overline{V}$ by $\bar{v}_1 \dots \bar{v}_k \in \mathbb{R}^n$. The Lagrangian for the objective is:

$$L(\{\bar{v}_1 \dots \bar{v}_k\}, \{\bar{\lambda}_i\}_{i=1}^k, \{\gamma_{ij}\}_{i \neq j}) = \sum_{i=1}^k \bar{v}_i^\top A \bar{v}_i - \sum_{i=1}^k \bar{\lambda}_i (\bar{v}_i^\top \bar{v}_i - 1) - \sum_{i \neq j} \gamma_{ij} \bar{v}_i^\top \bar{v}_j$$

It can be verified from stationary conditions that $\bar{v}_i^\top \bar{v}_i = 1$, $\bar{v}_i^\top \bar{v}_j = 0$ (for $i \neq j$), and $A\bar{v}_i = \bar{\lambda}_i \bar{v}_i$. Thus $\bar{v}_1 \dots \bar{v}_k$ are orthonormal eigenvectors of A corresponding to eigenvalues $\bar{\lambda}_1 \dots \bar{\lambda}_k$. Since the objective to maximize is

$$\operatorname{Tr}(\overline{V}^\top A \overline{V}) = \sum_{i=1}^k \bar{v}_i^\top A \bar{v}_i = \sum_{i=1}^k \bar{\lambda}_i$$

any set of orthonormal eigenvectors corresponding to the k largest eigenvalues $\lambda_1 \geq \dots \geq \lambda_k$ are optimal, in particular $V^* = [v_1 \dots v_k]$. □

3.2.4 Semidefinite Matrices

A symmetric matrix $A \in \mathbb{R}^{n \times n}$ always has an eigendecomposition with real eigenvalues. When all the eigenvalues of A are furthermore *non-negative*, A is called **positive semidefinite** or **PSD** and sometimes written as $A \succeq 0$. Equivalently, a symmetric matrix $A \in \mathbb{R}^{n \times n}$

is PSD if $v^\top Av \geq 0$ for all $v \in \mathbb{R}^n$; to see this, let $A = \sum_{i=1}^n \lambda_i v_i v_i^\top$ be an eigendecomposition and note that

$$v^\top Av = \sum_{i=1}^n \lambda_i (v^\top v_i)^2 \geq 0 \ \ \forall v \in \mathbb{R}^n \qquad \iff \qquad \lambda_i \geq 0 \ \ \forall 1 \leq i \leq n$$

A PSD matrix whose eigenvalues are strictly positive is called **positive definite** and written as $A \succ 0$. Similarly as above, $A \succ 0$ iff $v^\top Av > 0$ for all $v \neq 0$. Matrices that are **negative semidefinite** and **negative definite** are symmetrically defined (for non-positive and negative eigenvalues).

These matrices are important because they arise naturally in many settings.

Example 3.2.1 (Covariance matrix)**.** *The covariance matrix of a random variable $X \in \mathbb{R}^n$ is defined as*

$$C_X := \boldsymbol{E}\left[(X - \boldsymbol{E}[X])(X - \boldsymbol{E}[X])^\top\right]$$

which is clearly symmetric. For any $v \in \mathbb{R}^n$, let $Z := v^\top(X - \boldsymbol{E}[X])$ and note that $v^\top C_X v = \boldsymbol{E}[Z^2] \geq 0$, thus $C_X \succeq 0$.

Example 3.2.2 (Hessian)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function. The Hessian of f at x is defined as $\nabla^2 f(x) \in \mathbb{R}^{n \times n}$ where*

$$[\nabla^2 f(x)]_{i,j} := \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \qquad \forall i, j \in [n]$$

which is clearly symmetric. If x is stationary, $\nabla f(x) = 0$, then the spectral properties of $\nabla^2 f(x)$ determines the category of x.

- *If $\nabla^2 f(x) \succ 0$, then x is a local minimum. Consider any direction $u \in \mathbb{R}^n$. By Taylor's theorem, for a sufficiently small $\eta > 0$*

$$f(x + \eta u) \approx f(x) + \frac{\eta^2}{2} u^\top \nabla^2 f(x) u > f(x)$$

- *Likewise, if $\nabla^2 f(x) \prec 0$, then x is a local maximum.*

- *If $\nabla^2 f(x)$ has both positive and negative eigenvalues, then x is a saddle point. If $v_+ \in \mathbb{R}^n$ is an eigenvector corresponding to a positive eigenvalue,*

$$f(x + \eta v_+) \approx f(x) + \frac{\eta^2}{2} v_+^\top \nabla^2 f(x) v_+ > f(x)$$

If $v_- \in \mathbb{R}^n$ is an eigenvector corresponding to a negative eigenvalue,

$$f(x + \eta v_-) \approx f(x) + \frac{\eta^2}{2} v_-^\top \nabla^2 f(x) v_- < f(x)$$

Finally, if $\nabla^2 f(x) \succeq 0$ for all $x \in \mathbb{R}^n$, then f is convex. Given any $x, y \in \mathbb{R}^n$, for some z between x and y,

$$\begin{aligned} f(y) &= f(x) + \nabla f(x)^\top (y - x) + \frac{1}{2}(y - x)^\top \nabla^2 f(z)(y - x) \\ &\geq f(x) + \nabla f(x)^\top (y - x) \end{aligned}$$

Example 3.2.3 (Graph Laplacian)**.** *Consider an undirected weighted graph with n vertices $[n]$ and a (symmetric) adjacency matrix $W \in \mathbb{R}^{n \times n}$. The (i, j)-th entry of W is a non-negative weight $w_{ij} \geq 0$ for edge (i, j) where $w_{ij} = 0$ iff there is no edge (i, j). The degree of vertex $i \in [n]$ is defined as $d_i := \sum_{j=1}^n w_{ij}$ and assumed to be positive.*

The (unnormalized) graph Laplacian is a matrix whose spectral properties reveal the connectivity of the graph:

$$L := D - W \tag{3.27}$$

where $D := \operatorname{diag}(d_1, \ldots, d_n)$. Note that L does not depend on self-edges w_{ii} by construction. This matrix has the following properties (proofs can be found in Von Luxburg [2007]):

- *$L \succeq 0$ (and symmetric), so all its eigenvalues are non-negative.*
- *Moreover, the multiplicity of eigenvalue 0 is the number of connected components in the graph (so it is always at least 1).*
- *Suppose there are $m \leq n$ connected components $A_1 \ldots A_m$ (a partition of $[n]$). Represent each component $c \in [m]$ by an indicator vector $\mathbb{1}^c \in \{0, 1\}^n$ where*

$$\mathbb{1}_i^c = [\![\text{ vertex } i \text{ belongs to component } c \,]\!] \qquad \forall i \in [n]$$

Then $\{\mathbb{1}^1 \ldots \mathbb{1}^m\}$ is a basis of the zero eigenspace $E_{L,0}$.

3.2.5 Numerical Computation

Numerical computation of eigenvalues and eigenvectors is a deep subject beyond the scope of this thesis. Thorough treatments can be found in standard references such as Golub and Van Loan [2012]. Here, we supply basic results to give insight.

Consider computing eigenvectors (and their eigenvalues) of a diagonalizable matrix $A \in \mathbb{R}^{n\times n}$. A direct approach is to calculate the n roots of the polynomial $\det(A - \lambda I_{n\times n})$ in (3.19) and for each distinct root λ find an orthonormal basis of its eigenspace $E_{A,\lambda} = \{v : (A - \lambda I_{n\times n})v = 0\}$ in (3.20). Unfortunately, finding roots of a high-degree polynomial is a non-trivial problem of its own. But the following results provide more practical approaches to this problem.

3.2.5.1 Power Iteration

Theorem 3.2.9. *Let $A \in \mathbb{R}^{n\times n}$ be a nonzero symmetric matrix with eigenvalues $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$ and corresponding orthonormal eigenvectors $v_1 \ldots v_n$. Let $v \in \mathbb{R}^n$ be a vector chosen at random. Then $A^k v$ converges to some multiple of v_1 as k increases.*

Proof. Since $v_1 \ldots v_n$ form an orthonormal basis of $\mathbb{R}^n$ and v is randomly chosen from $\mathbb{R}^n$, $v = \sum_{i=1}^n c_i v_i$ for some nonzero $c_1 \ldots c_n \in \mathbb{R}$. Therefore,

$$A^k v = \lambda_1^k \left(c_1 v_1 + \sum_{i=2}^{n} c_i \left(\frac{\lambda_i}{\lambda_1} \right)^k v_i \right)$$

Since $|\lambda_i/\lambda_1| < 1$ for $i = 2 \ldots n$, the second term vanishes as k increases. □

A few remarks on Theorem 3.2.9:

- The proof suggests that the convergence rate depends on $\lambda_2/\lambda_1 \in [0, 1)$. If this is zero, $k = 1$ yields an exact estimate $Av = \lambda_1 c_1 v_1$. If this is nearly one, it may take a large value of k before $A^k v$ converges.

- The theorem assumes $|\lambda_1| > |\lambda_2|$ for simplicity (this is called a spectral gap condition), but there are more sophisticated analyses that do not depend on this assumption (e.g., Halko *et al.* [2011]).

- Once we have an estimate $\hat{v}_1 = A^k v$ of the dominant eigenvector v_1, we can calculate an estimate $\hat{\lambda}_1$ of the corresponding eigenvalue by solving

$$\hat{\lambda}_1 = \arg\min_{\lambda \in \mathbb{R}} ||A\hat{v}_1 - \lambda \hat{v}_1||_2 \tag{3.28}$$

  whose closed-form solution is given by the Rayleigh quotient:

$$\hat{\lambda}_1 = \frac{\hat{v}_1^\top A \hat{v}_1}{\hat{v}_1^\top \hat{v}_1} \tag{3.29}$$

- Once we have an estimate $(\hat{v}_1, \hat{\lambda}_1)$ of (v_1, λ_1), we can perform a procedure called **deflation**

$$A' := A - \hat{\lambda}_1 \hat{v}_1 \hat{v}_1^\top \approx A - \lambda_1 v_1 v_1^\top = \sum_{i=2}^{n} \lambda_i v_i v_i^\top \tag{3.30}$$

  If $\hat{v}_1 = v_1$, the dominant eigenvector of A' is exactly v_2 which can be estimated in a similar manner.

Theorem 3.2.9 suggests a scheme for finding eigenvectors of A, one by one, in the order of decreasing eigenvalues. This scheme is called the **power iteration method**; a basic version of the power method is given in Figure 3.3. Note that the eigenvector estimate is normalized in each iteration (Step 1(b)ii); this is a typical practice for numerical stability.

3.2.5.2 Orthogonal Iteration

Since the error introduced in deflation (3.30) propagates to the next iteration, the power method may be unstable for non-dominant eigen components. A natural generalization that remedies this problem is to find eigenvectors of A corresponding to the largest m eigenvalues simultaneously. That is, we start with m linearly independent vectors as columns of $\widetilde{V} = [\hat{v}_1 \dots \hat{v}_m]$ and compute

$$A^k \widetilde{V} = [A^k \hat{v}_1 \dots A^k \hat{v}_m]$$

Note that each column of $A^k \widetilde{V}$ converges to the dominant eigen component (the case $m = 1$ degenerates to the power method). As long as the columns remain linearly independent (which we can maintain by orthogonalizing the columns in every multiplication by A), their

Input: symmetric $A \in \mathbb{R}^{n \times n}$, number of desired eigen components $m \leq n$

Simplifying Assumption: the m dominant eigenvalues of A are nonzero and distinct, $|\lambda_1| > \cdots > |\lambda_m| > |\lambda_{m+1}| > 0$ ($\lambda_{m+1} = 0$ if $m = n$), with corresponding orthonormal eigenvectors $v_1 \dots v_m \in \mathbb{R}^n$

1. For $i = 1 \dots m$,
   (a) Initialize $\hat{v}_i \in \mathbb{R}^n$ randomly from a unit sphere.
   (b) Loop until convergence:
      i. $\hat{v}_i \leftarrow A\hat{v}_i$
      ii. $\hat{v}_i \leftarrow \hat{v}_i / ||\hat{v}_i||$
   (c) Compute the corresponding eigenvalue $\hat{\lambda}_i \leftarrow \hat{v}_i^\top A \hat{v}_i$.
   (d) Deflate $A \leftarrow A - \hat{\lambda}_i \hat{v}_i \hat{v}_i^\top$.

Output: estimate $(\hat{v}_i, \hat{\lambda}_i)$ of (v_i, λ_i) for $i = 1 \dots m$

Figure 3.3: A basic version of the power iteration method.

span converges to the subspace spanned by the eigenvectors of A corresponding to the largest m eigenvalues under certain conditions (Chapter 7, Golub and Van Loan [2012]). Thus the desired eigenvectors can be recovered by finding an orthonormal basis of range($A^k \widetilde{V}$). The resulting algorithm is known as the **orthogonal iteration method** and a basic version of the algorithm is given in Figure 3.4. As in the power iteration method, a typical practice is to compute an orthonormal basis in each iteration rather than in the end to improve numerical stability; in particular, to prevent the estimate vectors from becoming linearly dependent (Step 2b).

3.2.5.3 Lanczos Method

We mention a final algorithm which is particularly effective when the goal is to compute only a small number of dominant eigenvalues of a large, sparse symmetric matrix. The algorithm is known as the **Lanczos method**; details of this method can be found in Chapter 9 of

Input: symmetric $A \in \mathbb{R}^{n \times n}$, number of desired eigen components $m \leq n$

Simplifying Assumption: the m dominant eigenvalues of A are nonzero and distinct, $|\lambda_1| > \cdots > |\lambda_m| > |\lambda_{m+1}| > 0$ ($\lambda_{m+1} = 0$ if $m = n$), with corresponding orthonormal eigenvectors $v_1 \ldots v_m \in \mathbb{R}^n$

1. Initialize $\widehat{V} \in \mathbb{R}^{n \times m}$ such that $\widehat{V}^\top \widehat{V} = I_{m \times m}$ randomly.
2. Loop until convergence:
   (a) $\widetilde{V} \leftarrow A\widehat{V}$
   (b) Update $\widehat{V}$ to be an orthonormal basis of range($\widetilde{V}$) (e.g., by computing the QR decomposition: $[\widehat{V}, R] \leftarrow \mathrm{QR}(\widetilde{V})$).
3. Compute the corresponding eigenvalues $\widehat{\Lambda} \leftarrow \widehat{V}^\top A \widehat{V}$.
4. Reorder the columns of $\widehat{V}$ and $\widehat{\Lambda}$ in descending absolute magnitude of eigenvalues.

Output: estimate $(\widehat{V}, \widehat{\Lambda})$ of (V, Λ) where $V = [v_1 \ldots v_m]$ and $\Lambda = \mathrm{diag}(\lambda_1 \ldots \lambda_m)$.

Figure 3.4: A basic version of the subspace iteration method.

Golub and Van Loan [2012]. We give a sketch of the algorithm to illustrate its mechanics. This is the algorithm we use in in implementing our works presented in Chapter 5, Chapter 6, and Chapter 7. More specifically, we use the SVDLIBC package provided by Rohde [2007] which employs the single-vector Lanczos method.

Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix. The Lanczos method seeks an orthogonal matrix $Q_n \in \mathbb{R}^{n \times n}$ such that

$$T = Q_n^\top A Q_n = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 \\ \beta_1 & \alpha_2 & \beta_2 & \ddots & \vdots \\ 0 & \beta_2 & \ddots & & \\ \vdots & & & \ddots & \beta_{n-1} \\ 0 & \cdots & & \beta_{n-1} & \alpha_n \end{bmatrix} \tag{3.31}$$

is a **tridiagonal** matrix (i.e., $T_{i,j} = 0$ except when $i \in \{j-1, j, j+1\}$). We know that such

matrices exist since A is diagonalizable; for instance, Q_n can be orthonormal eigenvectors of A. Note that T is symmetric (since A is). From an eigendecompsition of $T = \widetilde{V}\widetilde{\Lambda}\widetilde{V}^\top$, we can recover an eigendecomposition of the original matrix $A = V\widetilde{\Lambda}V^\top$ where $V = Q_n\widetilde{V}$ since

$$A = Q_n T Q_n^\top = (Q_n\widetilde{V})\widetilde{\Lambda}(Q_n\widetilde{V})^\top$$

is an eigendecomposition. The Lanczos method is an iterative scheme to efficiently calculate Q_n and, in the process, simultaneously compute the tridiagonal entries $\alpha_1 \dots \alpha_n$ and $\beta_1 \dots \beta_{n-1}$.

Lemma 3.2.10. *Let $A \in \mathbb{R}^{n\times n}$ be symmetric and $Q_n = [q_1 \dots q_n]$ be an orthogonal matrix such that $T = Q_n^\top A Q_n$ has the tridiagonal form in* (3.31)*. Define $q_0 = q_{n+1} = 0$, $\beta_0 = 1$, and*

$$r_i := Aq_i - \beta_{i-1}q_{i-1} - \alpha_i q_i \qquad 1 \le i \le n$$

Then

$$\alpha_i = q_i^\top A q_i \qquad 1 \le i \le n \tag{3.32}$$

$$\beta_i = ||r_i||_2 \qquad 1 \le i \le n-1 \tag{3.33}$$

$$q_{i+1} = r_i/\beta_i \qquad 1 \le i \le n-1 \tag{3.34}$$

Proof. Since $AQ_n = Q_nT$, by the tridiagonal structure of T,

$$Aq_i = \beta_{i-1}q_{i-1} + \alpha_i q_i + \beta_i q_{i+1} \qquad 1 \le i \le n$$

Multiplying on the left by q_i and using the orthonormality of $q_1 \dots q_n$, we verify $\alpha_i = q_i^\top A q_i$. Rearranging the expression gives

$$\beta_i q_{i+1} = Aq_i - \beta_{i-1}q_{i-1} - \alpha_i q_i = r_i$$

Since q_{i+1} is a unit vector for $1 \le i \le n-1$, we have $\beta_i = ||r_i||$ and $q_{i+1} = r_i/\beta_i$. □

Input: symmetric $A \in \mathbb{R}^{n \times n}$ with dominant eigenvalues $|\lambda_1| \geq \cdots \geq |\lambda_m| > 0$ and corresponding orthonormal eigenvectors $v_1 \dots v_m \in \mathbb{R}^n$, number of desired eigen components $m \leq n$

1. Initialize $\hat{q}_1 \in \mathbb{R}^n$ randomly from a unit sphere and let $\hat{q}_0 = 0$ and $\hat{\beta}_0 = 1$.

2. For $i = 1 \dots m$,

   (a) Compute $\hat{\alpha}_i \leftarrow \hat{q}_i^\top A \hat{q}_i$. If $i < m$, compute:

   $$\hat{r}_i \leftarrow A\hat{q}_i - \hat{\beta}_{i-1}\hat{q}_{i-1} - \hat{\alpha}_i \hat{q}_i \qquad \hat{\beta}_i \leftarrow ||\hat{r}_i||_2 \qquad \hat{q}_{i+1} \leftarrow \hat{r}_i / \hat{\beta}_i$$

3. Compute the eigenvalues $|\hat{\lambda}_1| \geq \dots \geq |\hat{\lambda}_m|$ and the corresponding orthonormal eigenvectors $\hat{w}_1 \dots \hat{w}_m \in \mathbb{R}^m$ of the $m \times m$ tridiagonal matrix:

   $$\widehat{T} = \begin{bmatrix} \hat{\alpha}_1 & \hat{\beta}_1 & 0 & \cdots & 0 \\ \hat{\beta}_1 & \hat{\alpha}_2 & \hat{\beta}_2 & \ddots & \vdots \\ 0 & \hat{\beta}_2 & \ddots & & \\ \vdots & & & \ddots & \hat{\beta}_{m-1} \\ 0 & \cdots & & \hat{\beta}_{m-1} & \hat{\alpha}_m \end{bmatrix}$$

   (e.g., using the orthogonal iteration method).

4. Let $\hat{v}_i \leftarrow \widehat{Q}_m \hat{w}_i$ where $\widehat{Q}_m := [\hat{q}_1 \dots \hat{q}_m] \in \mathbb{R}^{n \times m}$.

Output: estimate $(\hat{v}_i, \hat{\lambda}_i)$ of (v_i, λ_i) for $i = 1 \dots m$

Figure 3.5: A basic version of the Lanczos method.

Lemma 3.2.10 suggests that we can seed a random unit vector q_1 and iteratively compute $(\alpha_i, \beta_{i-1}, q_i)$ for all $i \leq n$. Furthermore, it can be shown that if we terminate this iteration early at $i = m \leq n$, the eigenvalues and eigenvectors of the resulting $m \times m$ tridiagonal matrix are a good approximation of the m dominant eigenvalues and eigenvectors of the original matrix A. It can also be shown that the Lanczos method converges faster than the power iteration method [Golub and Van Loan, 2012].

A basic version of the Lanczos method shown in Figure 3.5. The main computation in each iteration is a matrix-vector product $A\hat{q}_i$ which can be made efficient if A is sparse.

3.3 Singular Value Decomposition (SVD)

Singular value decomposition (SVD) is an application of eigendecomposition to factorize *any* matrix $A \in \mathbb{R}^{m\times n}$.

3.3.1 Derivation from Eigendecomposition

SVD can be derived from an observation that $AA^\top \in \mathbb{R}^{m\times m}$ and $A^\top A \in \mathbb{R}^{n\times n}$ are symmetric and PSD, and have the same number of nonzero (i.e., positive) eigenvalues since $\text{rank}(A^\top A) = \text{rank}(AA^\top) = \text{rank}(A)$.

Theorem 3.3.1. *Let $A \in \mathbb{R}^{m\times n}$. Let $\lambda_1 \geq \ldots \geq \lambda_m \geq 0$ denote the m eigenvalues of $AA^\top$ and $\lambda'_1 \geq \ldots \geq \lambda'_n \geq 0$ the n eigenvalues of $A^\top A$. Then*

$$\lambda_i = \lambda'_i \qquad 1 \leq i \leq \min\{m, n\} \tag{3.35}$$

Moreover, there exist orthonormal eigenvectors $u_1 \ldots u_m \in \mathbb{R}^m$ of $AA^\top$ corresponding to $\lambda_1 \ldots \lambda_m$ and orthonormal eigenvectors $v_1 \ldots v_n \in \mathbb{R}^n$ of $A^\top A$ corresponding to $\lambda'_1 \ldots \lambda'_n$ such that

$$A^\top u_i = \sqrt{\lambda_i} v_i \qquad 1 \leq i \leq \min\{m, n\} \tag{3.36}$$

$$Av_i = \sqrt{\lambda_i} u_i \qquad 1 \leq i \leq \min\{m, n\} \tag{3.37}$$

Proof. Let $u_1 \ldots u_m \in \mathbb{R}^m$ be orthonormal eigenvectors of $AA^\top$ corresponding to eigenvalues $\lambda_1 \geq \ldots \geq \lambda_m \geq 0$. Pre-multiplying $AA^\top u_i = \lambda_i u_i$ by $A^\top$ and $u_i^\top$, we obtain

$$A^\top A(A^\top u_i) = \lambda_i (A^\top u_i) \tag{3.38}$$

$$\lambda_i = \left|\left| A^\top u_i \right|\right|_2^2 \tag{3.39}$$

The first equality shows that $A^\top u_i$ is an eigenvector of $A^\top A$ corresponding to an eigenvalue λ_i. Since this holds for all i and both $AA^\top$ and $A^\top A$ have the same number of nonzero eigenvalues, we have (3.35).

Now, construct $v_1 \dots v_m$ as follows. Let eigenvectors v_i of $A^\top A$ corresponding to nonzero eigenvalues $\lambda_i > 0$ be:

$$v_i = \frac{A^\top u_i}{\sqrt{\lambda_i}} \tag{3.40}$$

These vectors are unit-length eigenvectors of $A^\top A$ by (3.38) and (3.39). Furthermore, they are orthogonal: if $i \neq j$,

$$v_i^\top v_j = \frac{u_i^\top A A^\top u_j}{\sqrt{\lambda_i \lambda_j}} = \sqrt{\frac{\lambda_j}{\lambda_i}} u_i^\top u_j = 0$$

Let eigenvectors v_i of $A^\top A$ corresponding to zero eigenvalues $\lambda_i = 0$ be any orthonormal basis of $E_{A^\top A,0}$. Since this subspace is orthogonal to eigenvectors of $A^\top A$ corresponding to nonzero eigenvalues, we conclude that all $v_1 \dots v_m$ are orthonormal.

It remains to verify (3.36) and (3.37). For $\lambda_i > 0$, they follow immediately from (3.40). For $\lambda_i = 0$, $A^\top u_i$ and Av_i must be zero vectors since $\left|\left|A^\top u_i\right|\right|_2^2 = \lambda_i$ by (3.39) and also $||Av_i||_2^2 = v_i^\top A^\top A v_i = \lambda_i$; thus (3.36) and (3.37) hold trivially. □

The theorem validates the following definition.

Definition 3.3.1. *Let $A \in \mathbb{R}^{m \times n}$. Let $u_1 \dots u_m \in \mathbb{R}^m$ be orthonormal eigenvectors of $AA^\top$ corresponding to eigenvalues $\lambda_1 \geq \dots \geq \lambda_m \geq 0$, let $v_1 \dots v_n \in \mathbb{R}^n$ be orthonormal eigenvectors of $A^\top A$ corresponding to eigenvalues $\lambda'_1 \geq \dots \geq \lambda'_n \geq 0$, such that*

$$\begin{aligned} \lambda_i &= \lambda'_i && 1 \leq i \leq \min\{m,n\} \\ A^\top u_i &= \sqrt{\lambda_i} v_i && 1 \leq i \leq \min\{m,n\} \\ A v_i &= \sqrt{\lambda_i} u_i && 1 \leq i \leq \min\{m,n\} \end{aligned}$$

*The **singular values** $\sigma_1 \dots \sigma_{\max\{m,n\}}$ of A are defined as:*

$$\sigma_i := \begin{cases} \sqrt{\lambda_i} & 1 \leq i \leq \min\{m,n\} \\ 0 & \min\{m,n\} < i \leq \max\{m,n\} \end{cases} \tag{3.41}$$

*The vector u_i is called a **left singular vector** of A corresponding to σ_i. The vector v_i is called a **right singular vector** of A corresponding to σ_i. Define*

- *$U \in \mathbb{R}^{m \times m}$ is an orthogonal matrix $U := [u_1 \dots u_m]$.*

- $\Sigma \in \mathbb{R}^{m \times n}$ *is a rectangular diagonal matrix with* $\Sigma_{i,i} = \sigma_i$ *for* $1 \leq i \leq \min\{m, n\}$.
- $V \in \mathbb{R}^{n \times n}$ *is an orthogonal matrix* $V := [v_1 \dots v_n]$.

and note that $AV = U\Sigma$. *This gives a* ***singular value decomposition (SVD)*** *of* A:

$$A = U\Sigma V^\top = \sum_{i=1}^{\min\{m,n\}} \sigma_i u_i v_i^\top \tag{3.42}$$

If A is already symmetric, there is a close relation between an eigendecomposition of A and an SVD of A.

Proposition 3.3.1. *If* $A \in \mathbb{R}^{n \times n}$ *is symmetric and* $A = V \operatorname{diag}(\lambda_1 \dots \lambda_n) V^\top$ *is an orthonormal eigendecomposition of* A *with* $\lambda_1 \geq \cdots \geq \lambda_n$, *then* $A = V \operatorname{diag}(|\lambda_1| \dots |\lambda_n|) V^\top$ *is an SVD of* A.

Proof. Since $V \operatorname{diag}(\lambda_1^2 \dots \lambda_n^2) V^\top$ is an eigendecomposition of $AA^\top$ and $A^\top A$, the i-th singular value of A is $\sigma_i = \sqrt{\lambda_i^2} = |\lambda_i|$ and the left and right singular vectors corresponding to σ_i are both the i-th column of V. □

Corollary 3.3.2. *If* $A \in \mathbb{R}^{n \times n}$ *is symmetric, an eigendecomposition of* A *and an SVD of* A *are the same iff* $A \succeq 0$.

As emphasized in Chapter 6.7 of Strang [2009], given a matrix $A \in \mathbb{R}^{m \times n}$ with rank r, an SVD yields an orthonormal basis for each of the four subspaces assocated with A:

$$\begin{aligned} \operatorname{col}(A) &= \operatorname{span}\{u_1 \dots u_r\} \\ \operatorname{row}(A) &= \operatorname{span}\{v_1 \dots v_r\} \\ \operatorname{null}(A) &= \operatorname{span}\{v_{r+1} \dots v_n\} \\ \text{left-null}(A) &= \operatorname{span}\{u_{r+1} \dots u_m\} \end{aligned}$$

A typical practice, however, is to only find singular vectors corresponding to a few dominant singular values (in particular, ignore zero singular values).

Definition 3.3.2 (Low-rank SVD)**.** *Let $A \in \mathbb{R}^{m\times n}$ with rank r. Let $u_1 \dots u_r \in \mathbb{R}^m$ and $v_1 \dots v_r \in \mathbb{R}^n$ be left and right singular vectors of A corresponding to the (only) positive singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$. Let $k \leq r$. A* ***rank-k SVD*** *of A is*

$$\widehat{A} = U_k \Sigma_k V_k^\top = \sum_{i=1}^{k} \sigma_i u_i v_i^\top \tag{3.43}$$

where $U_k := [u_1 \dots u_k] \in \mathbb{R}^{m\times k}$, $\Sigma_k := \mathrm{diag}(\sigma_1 \dots \sigma_k) \in \mathbb{R}^{k\times k}$, and $V_k := [v_1 \dots v_k] \in \mathbb{R}^{n\times k}$. Note that $\widehat{A} = A$ if $k = r$.

3.3.2 Variational Characterization

As in eigendecomposition, we can view SVD as solving a constrained optimization problem.

Theorem 3.3.3. *Let $A \in \mathbb{R}^{m\times n}$ with left singular vectors $u_1 \dots u_p \in \mathbb{R}^m$ and right singular vectors $v_1 \dots v_p \in \mathbb{R}^n$ corresponding to singular values $\sigma_1 \geq \dots \geq \sigma_p \geq 0$ where $p := \min\{m, n\}$. Let $k \leq p$. Consider maximizing $u^\top A v$ over unit-length vector pairs $(u, v) \in \mathbb{R}^m \times \mathbb{R}^n$ under orthogonality constraints:*

$$(u_i^*, v_i^*) = \underset{\substack{(u,v)\in\mathbb{R}^m\times\mathbb{R}^n: \\ ||u||_2=||v||_2=1 \\ u^\top u_j^* = v^\top v_j^* = 0 \ \forall j<i}}{\arg\max} u^\top A v \qquad \text{for } i = 1 \dots k$$

Then an optimal solution is given by $(u_i^*, v_i^*) = (u_i, v_i)$.

Proof. The proof is similar to the proof of Theorem 3.2.7 and is omitted. □

A trace maximization interpretation of right (or left, by considering $A^\top$) singular vectors immediately follows from Theorem 3.2.8.

Corollary 3.3.4. *Let $A \in \mathbb{R}^{m\times n}$ with right singular vectors $v_1 \dots v_p \in \mathbb{R}^n$ corresponding to singular values $\sigma_1 \geq \dots \geq \sigma_p \geq 0$ where $p := \min\{m, n\}$. Let $k \leq p$. Consider maximizing the trace of $\overline{V}^\top A^\top A \overline{V} \in \mathbb{R}^{k\times k}$ over orthonormal matrices $\overline{V} \in \mathbb{R}^{n\times k}$:*

$$\begin{aligned} V^* &= \underset{\overline{V}\in\mathbb{R}^{n\times k}:\ \overline{V}^\top\overline{V}=I_{k\times k}}{\arg\max} \mathrm{Tr}(\overline{V}^\top A^\top A \overline{V}) \\ &= \underset{\overline{V}\in\mathbb{R}^{n\times k}:\ \overline{V}^\top\overline{V}=I_{k\times k}}{\arg\max} \left|\left|A\overline{V}\right|\right|_F^2 \end{aligned}$$

(the second equality is by definition). Then an optimal solution is $V^* = [v_1 \dots v_k]$.

Proof. This is equivalent to the optimization in Theorem 3.2.8 where the given symmetric matrix is $A^\top A$. The result follows from the definition of right singular vectors. □

Finally, a trace maximization interpretation jointly of left and right singular vectors is given below:

Theorem 3.3.5. *Let $A \in \mathbb{R}^{m\times n}$ with left singular vectors $u_1 \dots u_p \in \mathbb{R}^m$ and right singular vectors $v_1 \dots v_p \in \mathbb{R}^n$ corresponding to singular values $\sigma_1 \geq \dots \geq \sigma_p \geq 0$ where $p := \min\{m, n\}$. Let $k \leq p$. Consider maximizing the trace of $\overline{U}^\top A\overline{V} \in \mathbb{R}^{k\times k}$ over orthonormal matrices $\overline{U} \in \mathbb{R}^{m\times k}$ and $\overline{V} \in \mathbb{R}^{n\times k}$:*

$$(U^*, V^*) = \underset{\substack{\overline{U}\in\mathbb{R}^{m\times k},\ \overline{V}\in\mathbb{R}^{n\times k}:\\ \overline{U}^\top\overline{U}=\overline{V}^\top\overline{V}=I_{k\times k}}}{\arg\max} \operatorname{Tr}(\overline{U}^\top A\overline{V})$$

Then an optimal solution is given by $U^* = [u_1 \dots u_k]$ and $V^* = [v_1 \dots v_k]$.

Proof. The proof is similar to the proof of Theorem 3.2.8 and is omitted. □

3.3.3 Numerical Computation

Numerical computation of SVD is again an involved subject beyond the scope of this thesis. See Cline and Dhillon [2006] for references to a wide class of algorithms. Here, we give a quick remark on the subject to illustrate main ideas.

Let $A \in \mathbb{R}^{m\times n}$ with $m \leq n$ (if not, consider $A^\top$) and consider computing a rank-k SVD $U_k\Sigma_k V_k^\top$ of A. Since the columns of $U_k \in \mathbb{R}^{m\times k}$ are eigenvectors corresponding to the dominant k eigenvalues of $A^\top A \in \mathbb{R}^{m\times m}$ (which are squared singular values of A), we can compute an eigendecomposition of $A^\top A$ to obtain U_k and Σ_k, and finally recover $V_k = \Sigma_k^{-1}U_k^\top A$.

The core of many SVD algorithms is computing an eigendecomposition of $A^\top A$ efficiently without explicitly computing the matrix product. This can be done in various ways. For instance, we can modify the basic Lanczos algorithm in Figure 3.5 as follows: replace the matrix-vector product $A\hat{q}_i$ in Step 2a to $\hat{z}_i := A\hat{q}_i$ followed by $A^\top\hat{z}_i$. As another example, Matlab's sparse SVD (`svds`) computes an eigendecomposition of

$$B := \begin{bmatrix} 0 & A \\ A^\top & 0 \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}$$

and extracts the singular vectors and values of A from the eigendecomposition of B.

There is also a randomized algorithm for computing an SVD [Halko *et al.*, 2011]. While we do not use it in this thesis since other SVD algorithms are sufficiently scalable and efficient for our purposes, the randomized algorithm can potentially be used for computing an SVD of an extremely large matrix.

3.4 Perturbation Theory

Matrix perturbation theory is concerned with how properties of a matrix change when the matrix is perturbed by some noise. For instance, how "different" are the singular vectors of A from the singular vectors of $\widehat{A} = A + E$ where E is some noise matrix?

In the following, we let $\sigma_i(M) \geq 0$ denote the i-th largest singular value of M. We write $\angle\{u, v\}$ to denote the angle between nonzero vectors u, v taken in $[0, \pi]$.

3.4.1 Perturbation Bounds on Singular Values

Basic bounds on the singular values of a perturbed matrix are given below. They can also be used as bounds on eigenvalues for symmetric matrices.

Theorem 3.4.1 (Weyl [1912])**.** *Let $A, E \in \mathbb{R}^{m\times n}$ and $\widehat{A} = A + E$. Then*

$$\left|\sigma_i(\widehat{A}) - \sigma_i(A)\right| \leq ||E||_2 \qquad \forall i = 1 \ldots \min\{m, n\}$$

Theorem 3.4.2 (Mirsky [1960])**.** *Let $A, E \in \mathbb{R}^{m\times n}$ and $\widehat{A} = A + E$. Then*

$$\sum_{i=1}^{\min\{m,n\}} \left(\sigma_i(\widehat{A}) - \sigma_i(A)\right)^2 \leq ||E||_F^2$$

3.4.2 Canonical Angles Between Subspaces

To measure how "different" the singular vectors of A are from the singular vectors of $\widehat{A} = A + E$, we use the concept of an *angle* between the associated subspaces. This concept can be understood from the one-dimensional case. Suppose $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^n$ are subspaces of $\mathbb{R}^n$ with $\dim(\mathcal{X}) = \dim(\mathcal{Y}) = 1$. An acute angle $\angle\{\mathcal{X}, \mathcal{Y}\}$ between these subspaces can be

calculated as:

$$\angle\{\mathcal{X},\mathcal{Y}\} = \arccos \max_{\substack{x\in\mathcal{X},\, y\in\mathcal{Y}: \\ ||x||_2=||y||_2=1}} x^\top y$$

This is because $x^\top y = \cos\angle\{x,y\}$ for unit vectors x, y. Maximization ensures that the angle is acute. The definition can be extended as follows:

Definition 3.4.1. *Let $\mathcal{X},\mathcal{Y} \subset \mathbb{R}^n$ be subspaces of $\mathbb{R}^n$ with $\dim(\mathcal{X}) = d$ and $\dim(\mathcal{Y}) = d'$. Let $m := \min\{d,d'\}$. The **canonical angles** between $\mathcal{X}$ and $\mathcal{Y}$ are defined as*

$$\angle_i\{\mathcal{X},\mathcal{Y}\} := \arccos \max_{\substack{x\in\mathcal{X},\, y\in\mathcal{Y}: \\ ||x||_2=||y||_2=1 \\ x^\top x_j = y^\top y_j = 0 \ \forall j<i}} x^\top y \qquad \forall i = 1\ldots m$$

*The **canonical angle matrix** between $\mathcal{X}$ and $\mathcal{Y}$ is defined as*

$$\angle\{\mathcal{X},\mathcal{Y}\} := \operatorname{diag}(\angle_1\{\mathcal{X},\mathcal{Y}\}\ldots\angle_m\{\mathcal{X},\mathcal{Y}\})$$

Canonical angles can be found with SVD:

Theorem 3.4.3. *Let $X \in \mathbb{R}^{n\times d}$ and $Y \in \mathbb{R}^{n\times d'}$ be orthonormal bases for $\mathcal{X} := \operatorname{range}(X)$ and $\mathcal{Y} := \operatorname{range}(Y)$. Let $X^\top Y = U\Sigma V^\top$ be a rank-$(\min\{d,d'\})$ SVD of $X^\top Y$. Then*

$$\angle\{\mathcal{X},\mathcal{Y}\} = \arccos\Sigma$$

Proof. For all $1 \le i \le \min\{d,d'\}$,

$$\cos\angle_i\{\mathcal{X},\mathcal{Y}\} = \max_{\substack{x\in\operatorname{range}(X) \\ y\in\operatorname{range}(Y): \\ ||x||_2=||y||_2=1 \\ x^\top x_j = y^\top y_j = 0 \ \forall j<i}} x^\top y = \max_{\substack{u\in\mathbb{R}^d,\, v\in\mathbb{R}^{d'}: \\ ||u||_2=||v||_2=1 \\ u^\top u_j = v^\top v_j = 0 \ \forall j<i}} uX^\top Yv = \sigma_i$$

where we solve for u, v in $x = Xu$ and $y = Yv$ under the same constraints (using the orthonormality of X and Y) to obtain the second equality. The final equality follows from a variational characterization of SVD. □

Sine of the canonical angles The sine of the canoincal angles between subspaces is a natural measure of their difference partly because of its connection to the respective orthogonal projections (see Section 3.1.5).

Theorem 3.4.4 (Chapter 2, Stewart and Sun [1990]). *Let $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^n$ be subspaces of $\mathbb{R}^n$. Let $\Pi_{\mathcal{X}}, \Pi_{\mathcal{Y}} \in \mathbb{R}^{n \times n}$ be the (unique) orthogonal projections onto $\mathcal{X}, \mathcal{Y}$. Then*

$$||\sin \angle \{\mathcal{X}, \mathcal{Y}\}||_F = \frac{1}{\sqrt{2}} ||\Pi_{\mathcal{X}} - \Pi_{\mathcal{Y}}||_F$$

A result that connects the sine of canonical angles to singular values is the following:

Theorem 3.4.5 (Corollary 5.4, p. 43, Stewart and Sun [1990]). *Let $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^n$ be subspaces of $\mathbb{R}^n$ with the same dimension $\dim(\mathcal{X}) = \dim(\mathcal{Y}) = d$. Let $X, Y \in \mathbb{R}^{n \times d}$ be orthonormal bases of $\mathcal{X}, \mathcal{Y}$. Let $X_\perp, Y_\perp \in \mathbb{R}^{n \times (n-d)}$ be orthonormal bases of $\mathcal{X}^\perp, \mathcal{Y}^\perp$. Then the nonzero singular values of $Y_\perp^\top X$ or $X_\perp^\top Y$ are the sines of the nonzero canonical angles between $\mathcal{X}$ and $\mathcal{Y}$. In particular,*

$$||\sin \angle \{\mathcal{X}, \mathcal{Y}\}|| = \left|\left|Y_\perp^\top X\right|\right| = \left|\left|X_\perp^\top Y\right|\right| \tag{3.44}$$

where the norm can be $||\cdot||_2$ or $||\cdot||_F$.

3.4.3 Perturbation Bounds on Singular Vectors

Given the concept of canonical angles, We are now ready to state important bounds on the top singular vectors of a perturbed matrix attributed to Wedin [1972].

Theorem 3.4.6 (Wedin, spectral norm, p. 262, Theorem 4.4, Stewart and Sun [1990]). *Let $A, E \in \mathbb{R}^{m \times n}$ and $\widehat{A} = A + E$. Assume $m \geq n$. Let $A = U \Sigma V^\top$ and $\widehat{A} = \widehat{U} \widehat{\Sigma} \widehat{V}^\top$ denote SVDs of A and $\widehat{A}$. Choose the number of the top singular components $k \in [n]$ and write*

$$A = [U_1 U_2 U_3] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \\ 0 & 0 \end{bmatrix} [V_1 V_2]^\top \qquad \widehat{A} = \left[\widehat{U}_1 \widehat{U}_2 \widehat{U}_3\right] \begin{bmatrix} \widehat{\Sigma}_1 & 0 \\ 0 & \widehat{\Sigma}_2 \\ 0 & 0 \end{bmatrix} \left[\widehat{V}_1 \widehat{V}_2\right]^\top$$

where the matrices (U_1, Σ_1, V_1) with $\Sigma_1 \in \mathbb{R}^{k \times k}$, (U_2, Σ_2, V_2) with $\Sigma_2 \in \mathbb{R}^{(n-k) \times (n-k)}$, and a leftover $U_3 \in \mathbb{R}^{m \times (m-n)}$ represent a k-partition of $U \Sigma V$ (analogously for $\widehat{A}$). Let

$$\Phi := \angle \left\{ \text{range}(U_1), \text{range}(\widehat{U}_1) \right\}$$
$$\Theta := \angle \left\{ \text{range}(V_1), \text{range}(\widehat{V}_1) \right\}$$

If there exist $\alpha, \delta > 0$ *such that* $\sigma_k(\widehat{A}) \geq \alpha + \delta$ *and* $\sigma_{k+1}(A) \leq \alpha$*, then*

$$||\sin\Phi||_2 \leq \frac{||E||_2}{\delta} \qquad\qquad ||\sin\Theta||_2 \leq \frac{||E||_2}{\delta}$$

We point out some subtle aspects of Theorem 3.4.6. First, we can choose any matrices to bound $||\sin\Phi||_2$ as long as they have U_1 and $\widehat{U}_1$ as their top k left singular vectors (analogously for Θ). Second, we can flip the ordering of the singular value constraints (i.e., we can choose which matrix to treat as the original). For example, let $\widetilde{A} \in \mathbb{R}^{m \times n}$ be any matrix whose top k left singular vectors are $\widehat{U}_1$ (e.g., $\widetilde{A} = \widehat{U}_1 \widehat{\Sigma}_1 \widehat{V}_1^\top$). The theorem implies that if there exist $\alpha, \delta > 0$ such that $\sigma_k(A) \geq \alpha + \delta$ and $\sigma_{k+1}(\widetilde{A}) \leq \alpha$, then

$$||\sin\Phi||_2 \leq \frac{\left|\left|\widetilde{A} - A\right|\right|_2}{\delta}$$

There is also a Frobenius norm version of Wedin, which is provided here for completeness:

Theorem 3.4.7 (Wedin, Frobenius norm, p. 260, Theorem 4.1, Stewart and Sun [1990])**.** *Assume the same notations in Theorem 3.4.6. If there exists* $\delta > 0$ *such that* $\sigma_k(\widehat{A}) \geq \delta$ *and* $\min_{i=1\ldots k,\, j=k+1\ldots n} \left|\sigma_i(\widehat{A}) - \sigma_j(A)\right| \geq \delta$*, then*

$$||\sin\Phi||_F^2 + ||\sin\Theta||_F^2 \leq \frac{2\,||E||_F^2}{\delta^2}$$

Applications of Wedin to low-rank matrices Simpler versions of Wedin's theorem can be derived by assuming that A has rank k (i.e., our choice of the number of the top singular components exactly matches the number of nonzero singular values of A). This simplifies the condition in Theorem 3.4.6 because $\sigma_{k+1}(A) = 0$.

Theorem 3.4.8 (Wedin, Corollary 22, Hsu *et al.* [2012])**.** *Assume the same notations in Theorem 3.4.6. Assume* $\text{rank}(A) = k$ *and* $\text{rank}(\widehat{A}) \geq k$*. If* $\left|\left|\widehat{A} - A\right|\right|_2 \leq \epsilon\sigma_k(A)$ *for some* $\epsilon < 1$*, then*

$$||\sin\Phi||_2 \leq \frac{\epsilon}{1-\epsilon} \qquad\qquad ||\sin\Theta||_2 \leq \frac{\epsilon}{1-\epsilon}$$

Proof. For any value of $\alpha > 0$, define $\delta := \sigma_k(\widehat{A}) - \alpha$. Since $\sigma_k(\widehat{A})$ is positive, we can find a sufficiently small α such that δ is positive, thus the conditions $\sigma_k(\widehat{A}) \geq \alpha + \delta$ and

$\sigma_{k+1}(A) = 0 \leq \alpha$ in Theorem 3.4.6 are satisfied. It follows that

$$||\sin\Phi||_2 \leq \frac{\left|\left|\widehat{A} - A\right|\right|_2}{\delta} = \frac{\left|\left|\widehat{A} - A\right|\right|_2}{\sigma_k(\widehat{A}) - \alpha}$$

Since this is true for any $\alpha > 0$, we can take limit $\alpha \to 0$ on both sides to obtain

$$||\sin\Phi||_2 \leq \frac{\left|\left|\widehat{A} - A\right|\right|_2}{\sigma_k(\widehat{A})} \leq \frac{\epsilon\sigma_k(A)}{\sigma_k(\widehat{A})} \leq \frac{\epsilon\sigma_k(A)}{(1-\epsilon)\sigma_k(A)} = \frac{\epsilon}{1-\epsilon}$$

where the last inequality follows from Weyl's inequality: $\sigma_k(\widehat{A}) \geq (1-\epsilon)\sigma_k(A)$. The bound on the right singular vectors can be shown similarly. □

It is also possible to obtain a different bound by using an alternative argument.

Theorem 3.4.9 (Wedin)**.** *Assume the same notations in Theorem 3.4.6. Assume* $\text{rank}(A) = k$ *and* $\text{rank}(\widehat{A}) \geq k$*. If* $\left|\left|\widehat{A} - A\right|\right|_2 \leq \epsilon\sigma_k(A)$ *for some* $\epsilon < 1$*, then*

$$||\sin\Phi||_2 \leq 2\epsilon \qquad\qquad ||\sin\Theta||_2 \leq 2\epsilon$$

Proof. Define $\widetilde{A} := \widehat{U}_1\widehat{\Sigma}_1\widehat{V}_1^\top$. Note that $\left|\left|\widehat{A} - \widetilde{A}\right|\right|_2 \leq \left|\left|\widehat{A} - A\right|\right|_2$ since $\widetilde{A}$ is the optimal rank-k approximation of $\widehat{A}$ in $||\cdot||_2$ (Theorem 4.2.1). Then by the triangle inequality,

$$\left|\left|\widetilde{A} - A\right|\right|_2 \leq \left|\left|\widehat{A} - \widetilde{A}\right|\right|_2 + \left|\left|\widehat{A} - A\right|\right|_2 \leq 2\epsilon\sigma_k(A)$$

We now apply Theorem 3.4.6 with $\widetilde{A}$ as the original matrix and A as a perturbed matrix (see the remark below Theorem 3.4.6). Since $\sigma_k(A) > 0$ and $\sigma_{k+1}(\widetilde{A}) = 0$, we can use the same limit argument in the proof of Theorem 3.4.8 to have

$$||\sin\Phi||_2 \leq \frac{\left|\left|A - \widetilde{A}\right|\right|_2}{\sigma_k(A)} \leq \frac{2\epsilon\sigma_k(A)}{\sigma_k(A)} = 2\epsilon$$

The bound on the right singular vectors can be shown similarly. □

All together, we can state the following convenient corollary.

Corollary 3.4.10 (Wedin)**.** *Let* $A \in \mathbb{R}^{m\times n}$ *with rank* k*. Let* $E \in \mathbb{R}^{m\times n}$ *be a noise matrix and assume that* $\widehat{A} := A + E$ *has rank at least* k*. Let* $A = U\Sigma V^\top$ *and* $\widehat{A} = \widehat{U}\widehat{\Sigma}\widehat{V}^\top$ *denote*

rank-k SVDs of A and $\widehat{A}$. If $||E||_2 \leq \epsilon\sigma_k(A)$ for some $\epsilon < 1$, then for any orthonormal bases $U_\perp, \widehat{U}_\perp$ of $\text{range}(U)^\perp, \text{range}(\widehat{U})^\perp$ *and $V_\perp, \widehat{V}_\perp$ of* $\text{range}(V)^\perp, \text{range}(\widehat{V})^\perp$*, we have*

$$\left|\left|\widehat{U}_\perp^\top U\right|\right|_2 = \left|\left|U_\perp^\top \widehat{U}\right|\right|_2 \leq \min\left\{\frac{\epsilon}{1-\epsilon},\ 2\epsilon\right\}$$
$$\left|\left|\widehat{V}_\perp^\top V\right|\right|_2 = \left|\left|V_\perp^\top \widehat{V}\right|\right|_2 \leq \min\left\{\frac{\epsilon}{1-\epsilon},\ 2\epsilon\right\}$$

Note that if $\epsilon < 1/2$ the bound $\epsilon/(1-\epsilon) < 2\epsilon < 1$ is tighter.

Proof. The statement follows from Theorem 3.4.8, 3.4.9, and 3.4.5. □

It is also possible to derive a version of Wedin that does not involve orthogonal complements. The proof illustrates a useful technique: given any orthonormal basis $U \in \mathbb{R}^{m\times k}$,

$$I_{m\times m} = UU^\top + U_\perp U_\perp^\top$$

This allows for a decomposition of any vector in $\mathbb{R}^m$ into $\text{range}(U)$ and $\text{range}(U_\perp)$.

Theorem 3.4.11 (Wedin)**.** *Let $A \in \mathbb{R}^{m\times n}$ with rank k and $\widehat{A} \in \mathbb{R}^{m\times n}$ with rank at least k. Let $U, \widehat{U} \in \mathbb{R}^{m\times k}$ denote the top k left singular vectors of $A, \widehat{A}$. If $\left|\left|\widehat{A} - A\right|\right|_2 \leq \epsilon\sigma_k(A)$ for some $\epsilon < 1/2$, then*

$$\left|\left|\widehat{U}^\top x\right|\right|_2 \geq \sqrt{1-\epsilon_0^2}\ ||x||_2 \qquad \forall x \in \text{range}(U) \tag{3.45}$$

where $\epsilon_0 := \epsilon/(1-\epsilon) < 1$.

Proof. Since we have $||y||_2 = ||Uy||_2 = \left|\left|\widehat{U}y\right|\right|_2$ for any $y \in \mathbb{R}^k$, we can write

$$||y||_2^2 = \left|\left|\widehat{U}\widehat{U}^\top Uy\right|\right|_2^2 + \left|\left|\widehat{U}_\perp\widehat{U}_\perp^\top Uy\right|\right|_2^2 = \left|\left|\widehat{U}^\top Uy\right|\right|_2^2 + \left|\left|\widehat{U}_\perp^\top Uy\right|\right|_2^2$$

By Corollary 3.4.10, we have $\left|\left|\widehat{U}_\perp^\top U\right|\right|_2^2 \leq \epsilon_0^2 < 1$, thus

$$\left|\left|\widehat{U}^\top Uy\right|\right|_2^2 \geq \left(1 - \left|\left|\widehat{U}_\perp^\top U\right|\right|_2^2\right)||y||_2^2 \geq \left(1-\epsilon_0^2\right)||y||_2^2 \qquad \forall y \in \mathbb{R}^k$$

Then the claim follows. □

3.4.3.1 Examples

We now show how Wedin's theorem can be used in practice with some examples. In these examples, we assume a matrix $A \in \mathbb{R}^{m \times n}$ with rank k and an empirical estimate $\widehat{A}$ with rank at least k. Let $U, \widehat{U} \in \mathbb{R}^{m \times k}$ denote the top k left singular vectors of $A, \widehat{A}$.

In order to apply Wedin's theorem, we must establish that the empirical estimate $\widehat{A}$ is sufficiently accurate, so that

$$\left|\left|\widehat{A} - A\right|\right|_2 \leq \epsilon \sigma_k(A) \qquad \epsilon < 1/2 \tag{3.46}$$

Note that the condition depends on the smallest positive singular value of A. Let $\epsilon_0 := \epsilon/(1-\epsilon) < 1$.

Example 3.4.1 (Empirical invertibility, Hsu *et al.* [2008])**.** *Let $O \in \mathbb{R}^{m \times k}$ be a matrix such that* range(O) = range(U)*. Note that $U^\top O \in \mathbb{R}^{k \times k}$ is invertible. We now show that $\widehat{U}^\top O$ is also invertible if* (3.46) *holds. Apply* (3.45) *with $x = Oz$ to obtain:*

$$\left|\left|\widehat{U}^\top O z\right|\right|_2 \geq \sqrt{1-\epsilon_0^2}\ ||Oz||_2 \qquad \forall z \in \mathbb{R}^k$$

Since $\sigma_i(M)$ is the maximum of $||Mz||_2$ over orthonormally constrained z, this implies

$$\sigma_i\left(\widehat{U}^\top O\right) \geq \sqrt{1-\epsilon_0^2}\ \sigma_i(O) \qquad \forall i \in [k]$$

In particular, $\sigma_k\left(\widehat{U}^\top O\right) > 0$ and thus $\widehat{U}^\top O$ is invertible.

Example 3.4.2 (Empirical separability, Chapter 5)**.** *Assume that $Q \in \mathbb{R}^{k \times k}$ is an orthogonal matrix with columns $q_i \in \mathbb{R}^k$. That is,*

$$q_i^\top q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Let $\widehat{Q} := \widehat{U}^\top U Q$ with columns $\hat{q}_i \in \mathbb{R}^k$. We can bound the separation between the columns $\hat{q}_i$ assuming that (3.46) *holds. By Corollary 3.4.10, we have $\left|\left|\widehat{U}_\perp^\top U q_i\right|\right|_2 \leq \epsilon_0$. Then since*

$$||q_i||^2 = \left|\left|\widehat{U}\widehat{U}^\top U q_i\right|\right|^2 + \left|\left|\widehat{U}_\perp \widehat{U}_\perp^\top U q_i\right|\right|^2 = ||\hat{q}_i||^2 + \left|\left|\widehat{U}_\perp^\top U q_i\right|\right|^2 = 1$$

we have

$$\hat{q}_i^\top \hat{q}_i = 1 - \left|\left|\widehat{U}_\perp^\top U q_i\right|\right|_2^2 \geq 1 - \epsilon_0^2$$

Also, if $i \neq j$,

$$\begin{aligned}
\hat{q}_i^\top \hat{q}_j &= q_i^\top U^\top \widehat{U}\widehat{U}^\top U q_j \\
&= q_i^\top U^\top \left(I_{m\times m} - \widehat{U}_\perp \widehat{U}_\perp^\top \right) U q_j \\
&= q_i^\top q_j - q_i U^\top \widehat{U}_\perp \widehat{U}_\perp^\top U q_j \\
&= -q_i U^\top \widehat{U}_\perp \widehat{U}_\perp^\top U q_j \\
&\leq \left|\left| \widehat{U}_\perp^\top U q_i \right|\right|_2 \left|\left| \widehat{U}_\perp^\top U q_j \right|\right|_2 \\
&\leq \epsilon_0^2
\end{aligned}$$

where the first inequality is the Cauchy-Schwarz inequality.

Chapter 4

Examples of Spectral Techniques

This chapter gives examples of spectral techniques in the literature to demonstrate the range of spectral applications.

4.1 The Moore–Penrose Pseudoinverse

The **Moore–Penrose pseudoinverse** (or just the pseudoinverse) of a matrix $A \in \mathbb{R}^{m\times n}$ is the unique matrix $A^+ \in \mathbb{R}^{n\times m}$ such that[1]

1. $AA^+ \in \mathbb{R}^{n\times n}$ is the orthogonal projection onto range(A), and
2. $A^+A \in \mathbb{R}^{m\times m}$ is the orthogonal projection onto row(A).

A simple construction of the pseudoinverse A^+ is given by an SVD of A.

Proposition 4.1.1. *Let $A \in \mathbb{R}^{m\times n}$ with $r := \operatorname{rank}(A) \leq \min\{m, n\}$. Let $A = U\Sigma V^\top$ denote a rank-r SVD of A. Then $A^+ := V\Sigma^{-1}U^\top \in \mathbb{R}^{m\times n}$ is a matrix such that $AA^+ \in \mathbb{R}^{n\times n}$*

[1]This is a simplified definition sufficient for the purposes of the thesis: see Section 6.7 of Friedberg *et al.* [2003] for a formal treatment. It is defined as the matrix corresponding to a (linear) function $L : \mathbb{R}^n \to \mathbb{R}^m$ such that

$$L(v) = \begin{cases} u & \text{if } \exists u \text{ such that } v = Au \text{ (i.e., } v \in \text{range}(A)) \\ 0 & \text{otherwise} \end{cases}$$

from which the properties in the main text follow.

is the orthogonal projection onto $\text{range}(A)$ *and* $A^+A \in \mathbb{R}^{m\times m}$ *is the orthogonal projection onto* $\text{row}(A)$.

Proof. The orthogonal projections onto $\text{range}(A)$ and $\text{row}(A)$ are respectively given by $UU^\top$ and $VV^\top$, and since $U^\top U = V^\top V = I_{r\times r}$,

$$AA^+ = U\Sigma V^\top V\Sigma^{-1}U^\top = UU^\top$$
$$A^+A = V\Sigma^{-1}U^\top U\Sigma V^\top = VV^\top$$

□

The pseudoinverse A^+ is the unique minimizer of $||AX - I_{m\times m}||_F$ over $X \in \mathbb{R}^{n\times m}$ (p. 257, Golub and Van Loan [2012]) and can be seen as a generalization of matrix inverse:

- If A has linearly independent columns (so $A^\top A$ is invertible),

$$AA^+ = I_{m\times m}$$
$$A^+ = (A^\top A)^{-1}A^\top$$

- If A has linearly independent rows (so $AA^\top$ is invertible),

$$A^+A = I_{n\times n}$$
$$A^+ = A^\top(AA^\top)^{-1}$$

- If A is square and has full rank, then $A^+ = A^{-1}$.

4.2 Low-Rank Matrix Approximation

A celebrated application of SVD is the low-rank matrix approximation problem:

Theorem 4.2.1 (Eckart and Young [1936], Mirsky [1960])**.** *Let* $A \in \mathbb{R}^{m\times n}$. *Let* $k \leq \min\{m, n\}$ *and consider*

$$Z^* = \underset{Z\in\mathbb{R}^{m\times n}:\ \text{rank}(Z)\leq k}{\arg\min} ||A - Z|| \tag{4.1}$$

where $||\cdot||$ *is an orthogonally invariant norm:* $||M|| = ||QMR||$ *for orthogonal* Q *and* R *(e.g., the Frobenius norm* $||\cdot||_F$*, the spectral norm* $||\cdot||_2$*). Then an optimal solution is given by a rank-*k *SVD of* A*,* $Z^* = U_k\Sigma_k V_k^\top$.

Proof. Let $A = U\Sigma V$ be an SVD of A. Then

$$||A - Z||^2 = \left|\left|\Sigma - \overline{Z}\right|\right|^2 = \sum_{i=1}^{r} \left(\sigma_i - \overline{Z}_{i,i}\right)^2 + \sum_{i \neq j} \overline{Z}_{i,j}^2$$

where $\overline{Z} := U^\top Z V \in \mathbb{R}^{m \times n}$ has rank k. This is minimized (uniquely if $\sigma_k > \sigma_{k+1}$) at $\sum_{i=k+1}^{r} \sigma_i^2$ by a rectangular diagonal matrix $\overline{Z}_{i,i} = \sigma_i$ for $1 \leq i \leq k$, which implies $Z = U_k \Sigma_k V_k$. □

It is illuminating to examine a closely related unconstrained problem:

$$\{b_i^*\}_{i=1}^m, \{c_i^*\}_{i=1}^n = \underset{\substack{b_1 \ldots b_m \in \mathbb{R}^k \\ c_1 \ldots c_n \in \mathbb{R}^k}}{\arg\min} \sum_{i,j} \left(A_{i,j} - b_i^\top c_j\right)^2 \tag{4.2}$$

which in matrix form can be written as

$$(B^*, C^*) = \underset{\substack{B \in \mathbb{R}^{k \times m} \\ C \in \mathbb{R}^{k \times n}}}{\arg\min} \left|\left|A - B^\top C\right|\right|_F \tag{4.3}$$

This is equivalent to (4.1) (with the Frobenius norm) since any matrix with rank at most k can be expressed as $B^\top C$ (e.g., by SVD) and $\operatorname{rank}(B^\top C) \leq k$. It has infinite level sets since $\left|\left|A - B^\top C\right|\right|_F = \left|\left|A - \overline{B}^\top \overline{C}\right|\right|_F$ for $\overline{B} = Q^\top B$ and $\overline{C} = Q^{-1} C$ where Q is any $k \times k$ invertible matrix. For convenience, we can fix the form $B = \sqrt{\widetilde{\Sigma}_k}\widetilde{U}_k^\top$ and $C = \sqrt{\widetilde{\Sigma}_k}\widetilde{V}_k^\top$ by a rank-k SVD of $B^\top C = \widetilde{U}_k \widetilde{\Sigma}_k \widetilde{V}_k^\top$. The stationary conditions of (4.3) are then

$$A\widetilde{V}_k = \widetilde{U}_k \widetilde{\Sigma}_k \qquad\qquad A^\top \widetilde{U}_k = \widetilde{V}_k \widetilde{\Sigma}_k$$

which imply that each stationary point is given by some k singular components of A. In particular, the global minima are given by components corresponding to the largest k singular values (Theorem 4.2.1). Surprisingly, all other stationary points are saddle points; a proof can be found on page 29 of Ho [2008]. Thus (4.2) is a (very special type of) non-convex objective for which SVD provides a global minimum.

A slight variant of (4.2) is the following:

$$\{b_i^*\}_{i=1}^m, \{c_i^*\}_{i=1}^n = \underset{\substack{b_1 \ldots b_m \in \mathbb{R}^k \\ c_1 \ldots c_n \in \mathbb{R}^k}}{\arg\min} \sum_{i,j} W_{i,j} \left(A_{i,j} - b_i^\top c_j\right)^2 \tag{4.4}$$

where $W \in \mathbb{R}^{n \times m}$ is a non-negative weight matrix. Unfortunately, there is no SVD-based closed-form solution to this problem [Srebro *et al.*, 2003]. Unlike the unweighted case, the

objective has local optima that are not saddle points and can be shown to be generally NP-hard [Gillis and Glineur, 2011]. Despite the intractability, (4.4) is successfully optimized by iterative methods (e.g., gradient descent) in numerous practical applications such as recommender systems [Koren *et al.*, 2009] and word embeddings [Pennington *et al.*, 2014].

4.3 Finding the Best-Fit Subspace

A very practical interpretation of SVD is that of projecting data points to the "closest" lower-dimensional subspace. Specifically, let $x^{(1)} \dots x^{(M)} \in \mathbb{R}^d$ be M data points in $\mathbb{R}^d$. Given $k \leq d$, we wish to find an orthonormal basis $V^* = [v_1^* \dots v_k^*] \in \mathbb{R}^{d \times k}$ of a k-dimensional subspace such that

$$V^* = \underset{V \in \mathbb{R}^{d \times k} \colon V^\top V = I_{k \times k}}{\arg\min} \sum_{i=1}^{M} \left|\left| x^{(i)} - VV^\top x^{(i)} \right|\right|_2 \tag{4.5}$$

The subspace $\text{span}\{v_1^* \dots v_k^*\}$ is called the **best-fit subspace**. Since $x^{(i)} - VV^\top x^{(i)}$ is orthogonal to $VV^\top x^{(i)}$, by the Pythagorean theorem

$$\left|\left| x^{(i)} - VV^\top x^{(i)} \right|\right|_2^2 = \left|\left| x^{(i)} \right|\right|_2^2 - \left|\left| VV^\top x^{(i)} \right|\right|_2^2 = \left|\left| x^{(i)} \right|\right|_2^2 - \left|\left| V^\top x^{(i)} \right|\right|_2^2$$

Let $X \in \mathbb{R}^{M \times d}$ be a data matrix whose i-th row is given by $x^{(i)}$. Since $\sum_{i=1}^{M} \left|\left| V^\top x^{(i)} \right|\right|_2^2 = \text{Tr}(V^\top X^\top XV) = ||XV||_F^2$, (4.5) is equivalent to

$$V^* = \underset{V \in \mathbb{R}^{d \times k} \colon V^\top V = I_{k \times k}}{\arg\max} ||XV||_F \tag{4.6}$$

An optimal solution is given by $V^* = V_k$ where $U_k \Sigma_k V_k^\top$ is a rank-k SVD of X. The projected data points are given by the rows of $\underline{X} \in \mathbb{R}^{M \times k}$ where

$$\underline{X} = XV_k = U_k \Sigma_k \tag{4.7}$$

4.4 Principal Component Analysis (PCA)

Principal component analysis (PCA) is a classical spectral technique for dimensionality reduction [Pearson, 1901]. A standard formulation of PCA is as follows [Jolliffe, 2002].

Given a random variable $X \in \mathbb{R}^d$, we wish to find $m \leq d$ vectors $a_1 \dots a_m \in \mathbb{R}^d$ such that for each $i = 1 \dots m$:

$$\begin{aligned} a_i = &\arg\max_{a \in \mathbb{R}^d} \quad \mathrm{Var}(a^\top X) \\ &\text{subject to} \quad ||a||_2 = 1, \text{ and} \\ &\qquad\qquad a^\top a_j = 0 \text{ for all } j < i \end{aligned} \tag{4.8}$$

That is, $a_1 \dots a_m$ are orthonormal vectors such that a_i is the direction of the i-th largest variance of X. We express the objective in terms of the covariance matrix:

$$C_X := \boldsymbol{E}\left[(X - \boldsymbol{E}[X])(X - \boldsymbol{E}[X])^\top\right]$$

as $\mathrm{Var}(a^\top X) = a^\top C_X a$. Since $C_X \succeq 0$, it has an eigendecomposition of the form $C_X = U \Lambda U^\top$ where $U = [u_1 \dots u_d]$ is orthonormal and $\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_d)$ with $\lambda_1 \geq \dots \geq \lambda_d \geq 0$. Then a solution

$$\begin{aligned} a_i = &\arg\max_{a \in \mathbb{R}^d} \quad a^\top C_X a \\ &\text{subject to} \quad ||a||_2 = 1, \text{ and} \\ &\qquad\qquad a^\top a_j = 0 \text{ for all } j < i \end{aligned} \tag{4.9}$$

is given by $a_i = u_i$ and the value of the maximized variance is the eigenvalue λ_i since $\mathrm{Var}(a_i^\top X) = a_i^\top C_X a_i = \lambda_i$.

4.4.1 Best-Fit Subspace Interpretation

Let $x^{(1)} \dots x^{(M)}$ be M samples of X with the sample mean $\hat{\mu} := \sum_{i=1}^M x^{(i)}/M$. The sample covariance matrix is:

$$\hat{C}_X = \frac{1}{M} \sum_{i=1}^{M} (x^{(i)} - \hat{\mu})(x^{(i)} - \hat{\mu})^\top$$

By pre-processing the data as $\bar{x}^{(i)} := (x^{(i)} - \hat{\mu})/\sqrt{M}$ and organizing it into a matrix $\bar{X} \in \mathbb{R}^{M \times d}$ where $\bar{X}_i = \bar{x}^{(i)}$, we can write:

$$\hat{C}_X = \bar{X}^\top \bar{X}$$

Let $\bar{X} = \hat{U}\hat{\Sigma}\hat{V}^\top$ be an SVD of $\bar{X}$ where $\hat{\Sigma} = \mathrm{diag}(\hat{\sigma}_1 \dots \hat{\sigma}_d)$ is a diagonal matrix of ordered singular values $\hat{\sigma}_1 \geq \dots \geq \hat{\sigma}_d \geq 0$ and $\hat{V} = [\hat{v}_1 \dots \hat{v}_d]$ is the orthonormal matrix of right singular vectors. Since $\hat{C}_X = \hat{V}\hat{\Sigma}^2\hat{V}$ and it is an eigendecomposition in the desired form, the i-th PCA direction is given by $\hat{a}_i = \hat{v}_i$ and the value of the maximized variance is $\hat{\sigma}_i^2$. We make a few observations on this result:

- There is no need to explicitly compute the sample covariance matrix $\hat{C}_X$ and its eigendecomposition. We can directly apply an SVD on the data matrix $\bar{X}$.
- Since $\hat{a_1} \dots \hat{a_m}$ are the right singular vectors of $\sqrt{M}\bar{X}$ corresponding to the largest m singular values, the orthogonal projection $\hat{\Pi} := [\hat{a_1} \dots \hat{a_m}][\hat{a_1} \dots \hat{a_m}]^\top$ minimizes

$$\sum_{i=1}^{M} \left|\left|(x^{(i)} - \hat{\mu}) - \hat{\Pi}(x^{(i)} - \hat{\mu})\right|\right|^2$$

  Hence PCA can be interpreted as finding the best-fit subspace of mean-centered data points.

4.5 Canonical Correlation Analysis (CCA)

Canonical correlation analysis (CCA) is a classical spectral technique for analyzing the correlation between two variables [Hotelling, 1936]. A standard formulation of CCA is as follows [Hardoon *et al.*, 2004]. Given a pair of random variables $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}^{d'}$, we wish to find $m \leq \min(d, d')$ pairs of vectors $(a_1, b_1) \dots (a_m, b_m) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ recursively such that for each $i = 1 \dots m$:

$$\begin{aligned} (a_i, b_i) = \underset{(a,b)\in\mathbb{R}^d\times\mathbb{R}^{d'}}{\arg\max} \quad & \mathrm{Cor}(a^\top X, b^\top Y) \\ \text{subject to} \quad & \mathrm{Cor}(a^\top X, a_j^\top X) = 0 \ \text{ for all } j < i \\ & \mathrm{Cor}(b^\top Y, b_j^\top Y) = 0 \ \text{ for all } j < i \end{aligned} \tag{4.10}$$

That is, (a_i, b_i) projects (X, Y) to 1-dimensional random variables $(a_i^\top X, b_i^\top Y)$ that are maximally correlated, but $a_i^\top X$ is uncorrelated to $a_j^\top X$ for all $j < i$ (respectively for Y). Note that the solution is not unique because the correlation coefficient $\mathrm{Cor}(Y, Z)$ is invariant

under separate linear transformations on $Y, Z \in \mathbb{R}$:

$$\mathrm{Cor}(\alpha Y + \gamma, \beta Z + \lambda) = \mathrm{Cor}(Y, Z)$$

for any constants $\alpha, \beta, \gamma, \lambda \in \mathbb{R}$ where α and β are nonzero.

We express the objective in terms of the cross-covariance and covariance matrices:

$$C_{XY} := \boldsymbol{E}\left[(X - \boldsymbol{E}[X])(Y - \boldsymbol{E}[Y])^\top\right] \qquad C_X := \boldsymbol{E}\left[(X - \boldsymbol{E}[X])(X - \boldsymbol{E}[X])^\top\right]$$
$$C_Y := \boldsymbol{E}\left[(Y - \boldsymbol{E}[Y])(Y - \boldsymbol{E}[Y])^\top\right]$$

Since $\mathrm{Cor}(a^\top X, b^\top Y) = a^\top C_{XY} b / \sqrt{(a^\top C_X a)(b^\top C_Y b)}$, we write:

$$\begin{aligned}(a_i, b_i) = \underset{(a,b)\in\mathbb{R}^d\times\mathbb{R}^{d'}}{\arg\max} \quad & a^\top C_{XY} b \\ \text{subject to} \quad & a^\top C_X a = b^\top C_Y b = 1, \text{ and} \\ & a^\top C_X a_j = b^\top C_Y b_j = 0 \text{ for all } j < i\end{aligned} \tag{4.11}$$

We now consider a change of basis $c = C_X^{1/2} a$ and $d = C_Y^{1/2} b$. Assuming that C_X and C_Y are non-singular, we plug in $a = C_X^{-1/2} c$ and $b = C_Y^{-1/2} d$ above to obtain the auxiliary problem:

$$\begin{aligned}(c_i, d_i) = \underset{(c,d)\in\mathbb{R}^d\times\mathbb{R}^{d'}}{\arg\max} \quad & c^\top C_X^{-1/2} C_{XY} C_Y^{-1/2} d \\ \text{subject to} \quad & c^\top c = d^\top d = 1, \text{ and} \\ & c^\top c_j = d^\top d_j = 0 \text{ for all } j < i\end{aligned} \tag{4.12}$$

whereupon the original solution is given by $a_i = C_X^{-1/2} c_i$ and $b_i = C_Y^{-1/2} d_i$.

A solution of (4.12) is given by $c_i = u_i$ and $d_i = v_i$ where u_i and v_i are the left and right singular vectors of

$$\Omega := C_X^{-1/2} C_{XY} C_Y^{-1/2} \in \mathbb{R}^{d\times d'}$$

corresponding to the i-th largest singular value σ_i. The singular value σ_i is the value of the maximized correlation since $\mathrm{Cor}(a_i^\top X, b_i^\top Y) = a_i^\top C_{XY} b_i = u_i^\top \Omega v_i = \sigma_i$, thus it is bounded as $0 \le \sigma_i \le 1$. Note that the output of CCA also satisfies $\mathrm{Cor}(a_i^\top X, b_j^\top Y) = 0$ for all $i \neq j$ (even though this is not a constraint).

Matrix form We often organize the CCA vectors in matrix form. Specifically, given $(a_1, b_1) \dots (a_m, b_m) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ in (4.10), we let $A := [a_1 \dots a_m] \in \mathbb{R}^{d \times m}$ and $B := [b_1 \dots b_m] \in \mathbb{R}^{d' \times m}$ so that the dimensions of $A^\top X, B^\top Y \in \mathbb{R}^m$ are maximally correlated (pairwise). By (4.11), these matrices can be viewed as a solution of the following constrained trace optimization:

$$\begin{aligned}(A, B) = \underset{\overline{A} \in \mathbb{R}^{d \times m},\, \overline{B} \in \mathbb{R}^{d' \times m}}{\arg\max} \quad & \operatorname{Tr}\left(\overline{A}^\top C_{XY} \overline{B}\right) \\ \text{subject to} \quad & \overline{A}^\top C_X \overline{A} = \overline{B}^\top C_Y \overline{B} = I_{m \times m}\end{aligned} \tag{4.13}$$

4.5.1 Least Squares Interpretation

CCA can be viewed as minimizing the squared error between the induced representations.

Proposition 4.5.1. *Let $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}^{d'}$ be random variables. Let $A := [a_1 \dots a_m] \in \mathbb{R}^{d \times m}$ and $B := [b_1 \dots b_m] \in \mathbb{R}^{d' \times m}$ where $(a_i, b_i) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ is the CCA vector pair for (X, Y) defined in* (4.10)*. Then*

$$\begin{aligned}(A, B) = \underset{\overline{A} \in \mathbb{R}^{d \times m},\, \overline{B} \in \mathbb{R}^{d' \times m}}{\arg\min} \quad & \boldsymbol{E}\left[\left|\left|\overline{A}^\top (X - \boldsymbol{E}[X]) - \overline{B}^\top (Y - \boldsymbol{E}[Y])\right|\right|_2\right] \\ \textit{subject to} \quad & \overline{A}^\top C_X \overline{A} = \overline{B}^\top C_Y \overline{B} = I_{m \times m}\end{aligned} \tag{4.14}$$

Proof. By expanding the objective, we have

$$\begin{aligned}\boldsymbol{E}\left[\left|\left|\overline{A}^\top (X - \boldsymbol{E}[X]) - \overline{B}^\top (Y - \boldsymbol{E}[Y])\right|\right|_2^2\right] = \operatorname{Tr}\left(\overline{A}^\top C_X \overline{A}\right) + \operatorname{Tr}\left(\overline{B}^\top C_Y \overline{B}\right) \\ - 2 \operatorname{Tr}\left(\overline{A}^\top C_{XY} \overline{B}\right)\end{aligned}$$

Since the first two terms are constant (m) under the constraints, the minimization in (4.14) is equivalent to the trace maximization in (4.13). □

This least squares interpretation provides a different way to understand CCA and is also often convenient for computational reasons. For instance, Golub and Zha [1995] show that an alternating least squares algorithm solves (4.14). Ma *et al.* [2015] derive a gradient descent algorithm to find a locally optimal solution to (4.14).

4.5.2 New Coordinate Interpretation

CCA can also be framed as inducing new coordinate systems for the input variables $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}^{d'}$, called the **CCA coordinate systems**, in which they have special covariance structures.

Proposition 4.5.2. *Let $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}^{d'}$ be random variables with invertible covariance matrices. Let $U\Sigma V^\top$ denote an SVD of $\Omega := C_X^{-1/2} C_{XY} C_Y^{-1/2}$ and let $A := C_X^{-1/2} U$ and $B := C_Y^{-1/2} V$. If $X_{CCA} := A^\top (X - \boldsymbol{E}[X])$ and $Y_{CCA} := B^\top (Y - \boldsymbol{E}[Y])$,*

- *The covariance matrix of X_{CCA} is $I_{d \times d}$.*
- *The covariance matrix of Y_{CCA} is $I_{d' \times d'}$.*
- *The cross-covariance matrix of X_{CCA} and Y_{CCA} is Σ.*

Proof. For the first claim,

$$\boldsymbol{E}[X_{\text{CCA}} X_{\text{CCA}}^\top] = A^\top \boldsymbol{E}[(X - \boldsymbol{E}[X])(X - \boldsymbol{E}[X])^\top] A = U^\top C_X^{-1/2} C_X C_X^{-1/2} U = U^\top U = I_{d \times d}$$

The second claim follows similarly. For the third claim,

$$\boldsymbol{E}[X_{\text{CCA}} Y_{\text{CCA}}^\top] = A^\top \boldsymbol{E}[(X - \boldsymbol{E}[X])(Y - \boldsymbol{E}[Y])^\top] B = U^\top \Omega V = \Sigma$$

□

That is, in the CCA coordinates, the dimensions $i = 1 \ldots \min\{d, d'\}$ of each variable are sorted (in descending order) by the strength of correlation with the corresponding dimensions of the other variable.

4.5.3 Dimensionality Reduction with CCA

A significant part of the recent advance in spectral methods is due to the pioneering theoretical work by Kakade and Foster [2007] and Foster *et al.* [2008] that provides insights into how CCA can be used for dimensionality reduction in certain scenarios. We give a simplified version of these results.

The theory is based on multi-view learning for linear regression. Let $X^{(1)}, X^{(2)} \in \mathbb{R}^d$ be random variables with invertible covariance matrices representing two distinct "views" of

another variable (to be specified below). For simplicity, we assume that the two views have the same dimension, but this can be easily relaxed.

CCA coordinate convention Without loss of generality, we assume that $X^{(1)}, X^{(2)}$ are already put in the coordinate systems induced by CCA between $X^{(1)}$ and $X^{(1)}$ (Proposition 4.5.2). Thus they have zero means, identity covariance matrices, and a diagonal cross-covariance matrix $\Sigma = \text{diag}(\sigma_1 \dots \sigma_d)$ where $\sigma_i := \text{Cor}\left(X_i^{(1)}, X_i^{(2)}\right)$ is the i-th maximized correlation. This convention significantly simplifies the notations below. In particular, note that for each $v \in \{1, 2\}$, the top $m \leq d$ most correlated dimensions of $X^{(v)}$ are simply its first m entries which we denote by

$$\underline{X}^{(v)} := (X_1^{(v)} \dots X_m^{(v)})$$

This choice of an m-dimensional representation of the original variable leads to desirable properties under certain assumptions about the relation between $X^{(1)}$ and $X^{(2)}$ with respect to the variable being predicted.

4.5.3.1 Assumption 1: Shared Latent Variable

The **shared latent variable assumption** is that there is some latent variable $H \in \mathbb{R}^m$ where $m \leq d$ such that $X^{(1)}, X^{(2)} \in \mathbb{R}^d$ are (i) conditionally independent given H, and (ii) linear in H in expectation as follows: there exist full-rank matrices $A^{(1)}, A^{(2)} \in \mathbb{R}^{d \times m}$ such that

$$\boldsymbol{E}\left[X^{(1)}|H\right] = A^{(1)}H \qquad\qquad \boldsymbol{E}\left[X^{(2)}|H\right] = A^{(2)}H \tag{4.15}$$

We assume that $\boldsymbol{E}\left[HH^\top\right] = I_{m \times m}$ without loss of generality, since we can always whiten H and the linearity assumption is preserved.

Theorem 4.5.1 (Theorem 3, Foster *et al.* [2008])**.** *We make the shared latent variable assumption defined above. Let $A^{(*|v)} \in \mathbb{R}^{d \times m}$ denote the best linear predictor of $H \in \mathbb{R}^m$ with $X^{(v)} \in \mathbb{R}^d$:*

$$A^{(*|v)} := \underset{A \in \mathbb{R}^{d \times m}}{\arg\min}\ \boldsymbol{E}\left[\left|\left|A^\top X^{(v)} - H\right|\right|_2\right] = \boldsymbol{E}\left[X^{(v)}H^\top\right]$$

Let $\underline{A}^{(v)} \in \mathbb{R}^{m\times m}$ denote the best linear predictor of $H \in \mathbb{R}^m$ with $\underline{X}^{(v)} \in \mathbb{R}^m$:

$$\underline{A}^{(v)} := \underset{A\in\mathbb{R}^{m\times m}}{\arg\min}\ \boldsymbol{E}\left[\left|\left|A^\top \underline{X}^{(v)} - H\right|\right|_2\right] = \boldsymbol{E}\left[\underline{X}^{(v)} H^\top\right]$$

Then the optimal predictor $\underline{A}^{(v)}$ based on the top m most correlated dimensions is precisely as good as the optimal predictor $A^{(*|v)}$ based on all dimensions:

$$\left(A^{(*|v)}\right)^\top X^{(v)} = \left(\underline{A}^{(v)}\right)^\top \underline{X}^{(v)}$$

Proof. With the conditional independence of $X^{(1)}, X^{(2)}$ and the linear relation (4.15),

$$\begin{aligned}
\Sigma = \boldsymbol{E}\left[X^{(1)}\left(X^{(2)}\right)^\top\right] &= \boldsymbol{E}\left[\boldsymbol{E}\left[X^{(1)}\left(X^{(2)}\right)^\top \middle| H\right]\right] \\
&= \boldsymbol{E}\left[\boldsymbol{E}\left[X^{(1)}|H\right]\boldsymbol{E}\left[X^{(2)}|H\right]^\top\right] \\
&= A^{(v)}\,\boldsymbol{E}\left[HH^\top\right]\left(A^{(v)}\right)^\top \\
&= A^{(v)}\left(A^{(v)}\right)^\top
\end{aligned}$$

Thus $\Sigma \in \mathbb{R}^{d\times d}$ has rank m, implying that $\sigma_{m+1} = \cdots = \sigma_d = 0$. Let $\underline{\Sigma} \in \mathbb{R}^{m\times m}$ denote the $(m \times m)$ upper left block of Σ. Next, observe that the best predictor $A^{(*|v)}$ of H based on $X^{(v)}$ is in fact $A^{(v)}$:

$$A^{(v)} = \underset{A\in\mathbb{R}^{d\times m}}{\arg\min}\ \boldsymbol{E}\left[\left|\left|AH - X^{(v)}\right|\right|_2^2\right] = \boldsymbol{E}\left[X^{(v)}H^\top\right] = A^{(*|v)}$$

Together, we have

$$\left(A^{(*|v)}\right)^\top X^{(v)} = \left(A^{(v)}\right)^\top X^{(v)} = \left(A^{(v)}\right)^+ \Sigma X^{(v)} = \left(\underline{A}^{(v)}\right)^+ \underline{\Sigma} X^{(v)} = \left(\underline{A}^{(v)}\right)^\top \underline{X}^{(v)}$$

where we used the fact that $\left(A^{(v)}\right)^\top = \left(A^{(v)}\right)^+ \Sigma$ and that $\underline{A}^{(v)} = \boldsymbol{E}\left[\underline{X}^{(v)}H^\top\right]$ is the first m rows of $A^{(v)} = \boldsymbol{E}\left[X^{(v)}H^\top\right]$. □

4.5.3.2 Assumption 2: Redundancy of the Views

Let $Y \in \mathbb{R}$ and D denote the joint distribution over $(X^{(1)}, X^{(2)}, Y)$ (expectations are with respect to D unless otherwise noted). The **redundancy assumption** is that each individual view $X^{(v)}$ is nearly as (linearly) predictive of the response variable Y as the union of

them $X := (X^{(1)}, X^{(2)})$. More precisely, if we denote the best possible predictor $\beta^* \in \mathbb{R}^{2d}$ of Y with both views $X \in \mathbb{R}^{2d}$ by

$$\beta^* := \underset{\beta \in \mathbb{R}^{2d}}{\arg\min}\ \boldsymbol{E}\left[(\beta \cdot X - Y)^2\right]$$

and denote the best possible predictor $\beta^{(v)} \in \mathbb{R}^d$ of Y with only view $X^{(v)} \in \mathbb{R}^d$ by

$$\beta^{(v)} := \underset{\beta \in \mathbb{R}^{d}}{\arg\min}\ \boldsymbol{E}\left[(\beta \cdot X^{(v)} - Y)^2\right] = \boldsymbol{E}\left[X^{(v)}Y\right] \qquad \forall v \in \{1,2\}$$

then the ϵ-redundancy assumption is that for some ϵ,

$$\boldsymbol{E}\left[\left(\beta^{(v)} \cdot X^{(v)} - Y\right)^2\right] - \boldsymbol{E}\left[(\beta^* \cdot X - Y)^2\right] \leq \epsilon \qquad \forall v \in \{1,2\} \tag{4.16}$$

Lemma 4.5.2 (Lemma 2, Kakade and Foster [2007])**.** *Under the ϵ-redundancy assumption, the optimal predictor $\beta^{(v)}$ of Y with view $X^{(v)} \in \mathbb{R}^d$ cannot have large weights corresponding to weakly correlated dimensions,*

$$\sum_{i=1}^{d} (1-\sigma_i)\left(\beta_i^{(v)}\right)^2 \leq 4\epsilon \tag{4.17}$$

for each view $v \in \{1,2\}$. Note that the bound is independent of d.

Proof. It follows from (4.16) that $\boldsymbol{E}\left[\left(\beta^{(1)} \cdot X^{(1)} - \beta^{(2)} \cdot X^{(2)}\right)^2\right] \leq 4\epsilon$ (Lemma 1, Kakade and Foster [2007]). Furthermore, since $X^{(v)}$ is in the CCA coordinate system,

$$\begin{aligned}
\boldsymbol{E}\left[\left(\beta^{(1)} \cdot X^{(1)} - \beta^{(2)} \cdot X^{(2)}\right)^2\right] &= \sum_{i=1}^{d} \left(\beta_i^{(1)}\right)^2 + \left(\beta_i^{(2)}\right)^2 - 2\sigma_i \beta_i^{(1)} \beta_i^{(2)} \\
&= \sum_{i=1}^{d} (1-\sigma_i)\left(\beta_i^{(1)}\right)^2 + (1-\sigma_i)\left(\beta_i^{(2)}\right)^2 + \sigma_i\left(\beta_i^{(1)} - \beta_i^{(2)}\right)^2 \\
&\geq \sum_{i=1}^{d} (1-\sigma_i)\left(\beta_i^{(v)}\right)^2 \qquad \forall v \in \{1,2\}
\end{aligned}$$

Together, the stated bound is implied. □

Lemma 4.5.2 motivates discarding weakly correlated dimensions. Let

$$m = \left|\left\{i \in [d] :\ \mathrm{Cor}\left(X_i^{(1)}, X_i^{(2)}\right) \geq 1 - \sqrt{\epsilon}\right\}\right|$$

be the number of $(1-\sqrt{\epsilon})$**-strongly correlated dimensions**. Define a thresholded estimator $\beta^{(v)}_{\text{threshold}} \in \mathbb{R}^d$ by

$$[\beta^{(v)}_{\text{threshold}}]_i := \begin{cases} \boldsymbol{E}[X_i^{(v)} Y] & \text{if Cor}\left(X_i^{(1)}, X_i^{(2)}\right) \geq 1-\sqrt{\epsilon} \\ 0 & \text{otherwise} \end{cases} \tag{4.18}$$

which can be thought of as a *biased* estimator of $\beta^{(v)}$. Note that $\beta^{(v)}_{\text{threshold}} \cdot X^{(v)} = \underline{\beta}^{(v)} \cdot \underline{X}^{(v)}$, where $\underline{\beta}^{(v)}$ denotes the optimal linear predictor of Y with $\underline{X}^{(v)} \in \mathbb{R}^m$:

$$\underline{\beta}^{(v)} := \operatorname*{arg\,min}_{\beta \in \mathbb{R}^d} \boldsymbol{E}\left[(\beta \cdot \underline{X}^{(v)} - Y)^2\right] = \boldsymbol{E}\left[\underline{X}^{(v)} Y\right] \qquad \forall v \in \{1,2\}$$

Sample estimates We assume a set of n samples of $(X^{(1)}, X^{(2)}, Y)$ drawn iid from the distribution D,

$$T := \{(x_1^{(1)}, x_1^{(2)}, y_1) \ldots (x_n^{(1)}, x_n^{(2)}, y_n)\}$$

We use the superscript $\wedge$ to denote empirical estimates. For instance, $\hat{\beta}^{(v)} \in \mathbb{R}^d$ is defined as $\hat{\beta}^{(v)} := \frac{1}{n}\sum_{l=1}^n x_l^{(v)} y_l$, and $\underline{\hat{\beta}}^{(v)} \in \mathbb{R}^m$ is defined as $\underline{\hat{\beta}}^{(v)} := \frac{1}{n}\sum_{l=1}^n [x_l^{(v)}]_i y_l$ for $i \in [m]$. Note that the sample estimates are with respect to a fixed T. We use $\boldsymbol{E}_T[\cdot]$ to denote the expected value with respect to T.

Theorem 4.5.3 (Theorem 2, Kakade and Foster [2007])**.** *We make the ϵ-redundancy assumption defined above. Assuming $\boldsymbol{E}[Y^2|X] \leq 1$, the empirical estimate of $\underline{\beta}^{(v)} \in \mathbb{R}^m$ incurs the regret (in expectation)*

$$\boldsymbol{E}_T\left[\text{regret}_T^*\left(\underline{\hat{\beta}}^{(v)}\right)\right] \leq \sqrt{\epsilon}(\sqrt{\epsilon}+4) + \frac{m}{n}$$

where regret_T^* *is relative to the best possible predictor $\beta^* \in \mathbb{R}^{2d}$ using both views $X \in \mathbb{R}^{2d}$:*

$$\text{regret}_T^*\left(\underline{\hat{\beta}}^{(v)}\right) := \boldsymbol{E}\left[\left(\underline{\hat{\beta}}^{(v)} \cdot \underline{X}^{(v)} - Y\right)^2\right] - \boldsymbol{E}\left[(\beta^* \cdot X - Y)^2\right]$$

A remarkable aspect of this result is that as the number of samples n increases, the empirical estimate of the biased estimator $\beta^{(v)}_{\text{threshold}}$ converges[2] to the optimal estimator β^*

[2]The suboptimality $\sqrt{\epsilon}(\sqrt{\epsilon}+4)$ is due to bias and (4.16).

with no dependence on the original dimension d; it only depends on the number of $(1-\sqrt{\epsilon})$-strongly correlated dimensions m. Thus if $m \ll d$, then we need much fewer samples to estimate the biased estimator $\beta^{(v)}_{\text{threshold}}$ than to estimate the unbiased estimators $\beta^{(v)}$ or β^* (in which case the regret depends on d) to achieve (nearly) optimal regret.

Proof of Theorem 4.5.3. By (4.16), it is sufficient to show that

$$\boldsymbol{E}_T\left[\text{regret}_T\left(\hat{\underline{\beta}}^{(v)}\right)\right] \leq 4\sqrt{\epsilon} + \frac{m}{n}$$

where regret_T is relative to the best possible predictor $\beta^{(v)} \in \mathbb{R}^d$ using view $X^{(v)} \in \mathbb{R}^d$:

$$\text{regret}_T\left(\hat{\underline{\beta}}^{(v)}\right) := \boldsymbol{E}\left[\left(\hat{\underline{\beta}}^{(v)} \cdot \underline{X}^{(v)} - Y\right)^2\right] - \boldsymbol{E}\left[\left(\beta^{(v)} \cdot X^{(v)} - Y\right)^2\right]$$

The regret takes a particularly simple form because of linearity and the choice of coordinates. Given a fixed set of samples T (so that $\hat{\underline{\beta}}^{(v)}$ is not random),

$$\begin{aligned}
\text{regret}_T\left(\hat{\underline{\beta}}^{(v)}\right) &:= \boldsymbol{E}\left[\left(\hat{\underline{\beta}}^{(v)} \cdot \underline{X}^{(v)} - Y\right)^2\right] - \boldsymbol{E}\left[\left(\beta^{(v)} \cdot X^{(v)} - Y\right)^2\right] \\
&= \hat{\underline{\beta}}^{(v)} \cdot \hat{\underline{\beta}}^{(v)} - \beta^{(v)} \cdot \beta^{(v)} - 2\hat{\underline{\beta}}^{(v)} \cdot \boldsymbol{E}[\underline{X}^{(v)}Y] + 2\beta^{(v)} \cdot \boldsymbol{E}[X^{(v)}Y] \\
&= \left|\left|\hat{\beta}^{(v)}_{\text{threshold}}\right|\right|_2^2 - 2\hat{\beta}^{(v)}_{\text{threshold}} \cdot \beta^{(v)} + \left|\left|\beta^{(v)}\right|\right|_2^2 \\
&= \left|\left|\hat{\beta}^{(v)}_{\text{threshold}} - \beta^{(v)}\right|\right|_2^2
\end{aligned}$$

This allows for a *bias-variance decomposition* of the expected regret:

$$\begin{aligned}
\boldsymbol{E}_T\left[\text{regret}_T\left(\hat{\underline{\beta}}^{(v)}\right)\right] &= \boldsymbol{E}_T\left[\left|\left|\hat{\beta}^{(v)}_{\text{threshold}} - \beta^{(v)}\right|\right|_2^2\right] \\
&= \left|\left|\beta^{(v)}_{\text{threshold}} - \beta^{(v)}\right|\right|_2^2 + \boldsymbol{E}_T\left[\left|\left|\hat{\beta}^{(v)}_{\text{threshold}} - \beta^{(v)}\right|\right|_2^2\right] \\
&= \left|\left|\beta^{(v)}_{\text{threshold}} - \beta^{(v)}\right|\right|_2^2 + \sum_{i=1}^{m} \text{Var}\left(\hat{\underline{\beta}}^{(v)}_i\right)
\end{aligned}$$

The first term corresponds to the bias of the estimator, and the second term is the amount of variance with respect to T.

To bound the variance term, note that:

$$\begin{aligned}
\text{Var}\left(\hat{\underline{\beta}}^{(v)}_i\right) &= \frac{1}{n}\text{Var}\left(\underline{X}^{(v)}_i Y\right) \leq \frac{1}{n}\boldsymbol{E}\left[\left(\underline{X}^{(v)}_i Y\right)^2\right] \\
&= \frac{1}{n}\boldsymbol{E}\left[\left(\underline{X}^{(v)}_i\right)^2 \boldsymbol{E}\left[Y^2|X\right]\right] \leq \frac{1}{n}\boldsymbol{E}\left[\left(X^{(v)}_i\right)^2\right] = \frac{1}{n}
\end{aligned}$$

where the second variance is with respect to D. We used the assumption that $\boldsymbol{E}[Y^2|X] \leq 1$. So the variance term is bounded by m/n.

To bound the bias term, it is crucial to exploit the multi-view assumption (4.16). For all $i > m$ we have $\sigma_i < 1 - \sqrt{\epsilon}$ and thus $1 \leq (1 - \sigma_i)/\sqrt{\epsilon}$, so

$$\left|\left|\beta^{(v)}_{\text{threshold}} - \beta^{(v)}\right|\right|_2^2 = \sum_{i>m} \left(\beta_i^{(v)}\right)^2 \leq \sum_{i=1}^{d} \left(\beta_i^{(v)}\right)^2 \leq \sum_{i=1}^{d} \frac{1-\sigma_i}{\sqrt{\epsilon}} \left(\beta_i^{(v)}\right)^2 \leq 4\sqrt{\epsilon}$$

where the last step is by (4.17) and makes the bias term independent of d.

□

Connection to semi-supervised learning The theory suggest a natural way to utilize unlabeled data with CCA to augment supervised training. In a semi-supervised scenario, we assume that the amount of labeled samples is limited: $(x_1^{(1)}, y_1) \ldots (x_n^{(1)}, y_n)$ samples of $(X^{(1)}, Y)$ for some small n. But if there is a second view $X^{(2)}$ as predictive of Y as $X^{(1)}$ (i.e., the redundancy assumption) for which it is easy to obtain a large amount of unlabeled samples of $(X^{(1)}, X^{(2)})$,

$$(x_1^{(1)}, x_1^{(2)}) \ldots (x_{n'}^{(1)}, x_{n'}^{(2)}) \qquad\qquad n' \gg n$$

then we can leverage these unlabeled samples to accurately estimate CCA projection vectors. These projection vectors are used to eliminate the dimensions of the labeled samples $x_1^{(1)} \ldots x_n^{(1)}$ that are not strongly correlated with the other view's. Theorem 4.5.3 implies that the supervised model trained on these low dimensional samples (corresponding to the thresholded estimator) converges to the optimal model at a faster rate.

4.6 Spectral Clustering

Spectral clustering refers to partitioning vertices in an undirected graph by matrix decomposition [Donath and Hoffman, 1973; Fiedler, 1973]. Here, we give one example framed as finding vertex representations suitable for the clustering problem [Shi and Malik, 2000]; this approach is closely relevant to our word clustering method in Chapter 5. For other examples of spectral clustering, see Von Luxburg [2007].

Given an undirected weighted graph described in Example 3.2.3, we wish to find a partition $\mathcal{P} = \{A_1 \dots A_m\}$ of vertices $[n]$ where $m \leq n$. One sensible formulation is to minimize the "flow" $W(A, \overline{A}) := \sum_{i \in A,\, j \in \overline{A}} w_{ij}$ between each cluster $A \in \mathcal{P}$ and its complement $\overline{A} := [n] \backslash A$ to encourage cluster independence, while normalizing by the "volume" $\mathrm{vol}(A) := \sum_{i \in A} d_i$ to discourage an imbalanced partition. This gives the following objective:

$$\mathcal{P}^* = \arg\min_{\mathcal{P}} \sum_{A \in \mathcal{P}} \frac{W(A, \overline{A})}{\mathrm{vol}(A)} \tag{4.19}$$

This problem is NP-hard [Wagner and Wagner, 1993], but there is a spectral method for solving a relaxed version of this problem.

In this method, vertex i is represented as the i-th row of a matrix $X_{\mathcal{P}} \in \mathbb{R}^{n \times m}$ where:

$$[X_{\mathcal{P}}]_{i,c} = \begin{cases} \frac{1}{\sqrt{\mathrm{vol}(A_c)}} & \text{if } i \in A_c \\ 0 & \text{otherwise} \end{cases} \tag{4.20}$$

with respect to a specific partition $\mathcal{P} = \{A_1 \dots A_m\}$ of $[n]$. Note that $X_{\mathcal{P}}^\top D X_{\mathcal{P}} = I_{m \times m}$ (for all $\mathcal{P}$) by design. We invoke the following fact:

$$\sum_{A \in \mathcal{P}} \frac{W(A, \overline{A})}{\mathrm{vol}(A)} = \mathrm{Tr}(X_{\mathcal{P}}^\top L X_{\mathcal{P}})$$

where L denotes the unnormalized graph Laplacian $L := W - D$. This holds by properties of L and the definition of $X_{\mathcal{P}}$; see Von Luxburg [2007] for a proof. Use this fact to rewrite the clustering objective as

$$X^* = \arg\min_{X_{\mathcal{P}} \in \mathbb{R}^{n \times m}} \mathrm{Tr}(X_{\mathcal{P}}^\top L X_{\mathcal{P}}) \tag{4.21}$$
$$X_{\mathcal{P}} \text{ has the form in (4.20) for some } \mathcal{P}$$

whereupon the optimal clusters can be recovered as: $i \in A_c$ iff $X^*_{i,c} > 0$. We obtain a relaxation of (4.21) by weakening the explicit form constraint as:

$$\widetilde{X} = \arg\min_{X \in \mathbb{R}^{n \times m}} \mathrm{Tr}(X^\top L X) \tag{4.22}$$
$$\text{subject to } X^\top D X = I_{m \times m}$$

Using a change of basis $U = D^{1/2}X$ and plugging in $X = D^{-1/2}U$ above, we can solve

$$\widetilde{U} = \underset{U \in \mathbb{R}^{n \times m}}{\arg\min} \ \mathrm{Tr}(U^\top D^{-1/2} L D^{-1/2} U) \quad \text{subject to } U^\top U = I_{m \times m} \tag{4.23}$$

and let $\widetilde{X} = D^{-1/2}\widetilde{U}$. It can be verified that the solution of (4.23) is given by the orthonormal eigenvectors of $D^{-1/2}LD^{-1/2}$ (called the normalized graph Laplacian) corresponding to the m smallest eigenvalues $0 \leq \lambda_1 \leq \ldots \leq \lambda_m$. More directly, the solution of (4.22) is given by the eigenvectors of $D^{-1}L$ corresponding to the same eigenvalues $\Lambda := \mathrm{diag}(\lambda_1, \ldots, \lambda_m)$ since

$$(D^{-1/2}LD^{-1/2})\widetilde{U} = \Lambda\widetilde{U} \qquad \Longleftrightarrow \qquad D^{-1}L\widetilde{X} = \Lambda\widetilde{X}$$

This gives the clustering algorithm of Shi and Malik [2000]:

1. Construct the normalized graph Laplacian $\overline{L} = D^{-1}L$.
2. (Rank-m eigendecomposition) Compute the eigenvectors of $\overline{L}$ corresponding to the smallest m eigenvalues as columns of matrix $\widetilde{X} \in \mathbb{R}^{n \times m}$.
3. Cluster the rows of $\widetilde{X}$ into m groups $A_1 \ldots A_m$ (e.g., with k-means).

In summary, the method approximates the idealized vertex representations X^* in (4.21) (which, if given, can be used to trivially recover the optimal clusters) with a surrogate representation $\widetilde{X}$ that is efficiently computable with an eigendecomposition of the normalized graph Laplacian. While this approximation can be arbitrarily suboptimal in the worst case [Guattery and Miller, 1998], it is effective in practice [Shi and Malik, 2000; Ng *et al.*, 2002].

4.7 Subspace Identification

Spectral methods have recently garnered much interest as a promising approach to learning latent-variable models. A pioneering work in this direction is the spectral algorithm of Hsu *et al.* [2008] for estimating distributions under HMMs. The Hsu *et al.* method is an amalgam of many ideas; see the paper for a detailed discussion. A crucial component of the method is the use of SVD to identify a low-dimensional subspace associated with the

model. We give a brief, informal review of their algorithm and its extension by Foster *et al.* [2012] from an angle of subspace identification.

Consider an HMM with m hidden states $h \in [m]$ and n observation states $x \in [n]$ where $m \ll n$. This HMM can be parametrized as a matrix-vector tuple (T, O, π) where

$$\begin{aligned} T \in \mathbb{R}^{m\times m}: &\qquad T_{h',h} = \text{transition probability from state } h \text{ to } h' \\ O \in \mathbb{R}^{n\times m}: &\qquad O_{x,h} = \text{emission probability from state } h \text{ to observation } x \\ \pi \in \mathbb{R}^{m}: &\qquad \pi_h = \text{prior probability of state } h \end{aligned}$$

It is well-known (and easily checkable) that with the following definition of "observable operators" (A, a_∞, a_1) [Ito *et al.*, 1992; Jaeger, 2000]

$$\begin{aligned} A(x) &:= T\,\mathrm{diag}(O_{x,1}\dots O_{x,m}) && \forall x \in [n] \\ a_\infty^\top &:= 1_m^\top \\ a_1 &:= \pi \end{aligned}$$

(1_m is a vector of ones in $\mathbb{R}^m$), the probability of any observation sequence $x_1 \dots x_N \in [n]$ under the HMM is given by a product of these operators:

$$p(x_1 \dots x_N) = a_\infty^\top A(x_N) \cdots A(x_1) a_1 \tag{4.24}$$

That is, (4.24) is the matrix form of the forward algorithm [Rabiner, 1989]. The approach pursued by Hsu *et al.* [2008] and Foster *et al.* [2012] is to instead estimate certain linear transformations of the operators:

$$B(x) := GA(x)G^+ \qquad \forall x \in [n] \tag{4.25}$$

$$b_\infty^\top := a_\infty^\top G^+ \tag{4.26}$$

$$b_1 := Ga_1 \tag{4.27}$$

where G is a matrix such that $G^+G = I_{m\times m}$. It is clear that the forward algorithm can be computed by (B, b_∞, b_1), since

$$\begin{aligned} p(x_1 \dots x_N) &= a_\infty^\top A(x_N) \cdots A(x_1) a_1 \\ &= a_\infty^\top G^+GA(x_N)G^+G \cdots G^+GA(x_1)G^+Ga_1 \\ &= b_\infty^\top B(x_N) \cdots B(x_1) b_1 \end{aligned}$$

Let $X_1, X_2 \in [n]$ be random variables corresponding to the first two observations under the HMM (where we assume the usual generative story). A central quantity considered by Hsu *et al.* [2008] is a matrix of bigram probabilities $P_{2,1} \in \mathbb{R}^{n \times n}$ defined as

$$[P_{2,1}]_{x',x} := P(X_1 = x, X_2 = x') \qquad \forall x, x' \in [n] \tag{4.28}$$

The matrix relates the past observation (X_1) to the future observation (X_2). It can be shown that this matrix can be expressed in terms of the HMM parameters as (Lemma 3, Hsu *et al.* [2008]):

$$P_{2,1} = OT \operatorname{diag}(\pi) O^\top \tag{4.29}$$

It follows that $\operatorname{rank}(P_{2,1}) = m$ if $O, T, \operatorname{diag}(\pi)$ have full-rank—even though the dimension of the matrix is $n \times n$.

Hsu *et al.* [2008] apply SVD on $P_{2,1}$ to identify the m-dimensional subspace spanned by the conditional emission distributions: $(O_{1,h}, \ldots, O_{n,h})$ for all $h \in [m]$. Specifically, if $P_{2,1} = U \Sigma V^\top$ is a rank-m SVD, then it can be shown that (Lemma 2, Hsu *et al.* [2008])

$$\operatorname{range}(U) = \operatorname{range}(O) \tag{4.30}$$

This projection matrix $U \in \mathbb{R}^{n \times m}$ is then used to reduce the dimension of observations from $\mathbb{R}^n$ to $\mathbb{R}^m$, whereupon the linearly transformed operators (4.25–4.27) are recovered by the method of moments. Importantly, the spectral dimensionality reduction leads to polynomial sample complexity (Theorem 6, Hsu *et al.* [2008]; Theorem 1, Foster *et al.* [2012]).

Note that the statement is about the *true* probabilities $P_{2,1}$ under the HMM. In order to establish finite sample complexity bounds, we must consider the empirical estimate $\widehat{P}_{2,1}$ of $P_{2,1}$ where each entry

$$[\widehat{P}_{2,1}]_{x',x} := \frac{1}{N} \sum_{i=1}^{N} \left[\left[X_1 = x, X_2 = x' \right] \right] \qquad \forall x, x' \in [n]$$

is estimated from a finite number of samples N, and examine how a rank-m SVD $\widehat{U} \widehat{\Sigma} \widehat{V}^\top$ of $\widehat{P}_{2,1}$ behaves with respect to a rank-m SVD $U \Sigma V^\top$ of $P_{2,1}$ as a function of N. Deriving such bounds can be quite involved (see Section 3.4) and is a major technical contribution of Hsu *et al.* [2008].

It should be emphasized that the subspace identification component can be disentangled from the method of moments. In particular, it can be verified that removing U in their definitions of $\vec{b}_1$, $\vec{b}_\infty$, and B_x in Hsu *et al.* [2008] still results in a consistent estimator of the distribution in (4.24).

4.8 Alternating Minimization Using SVD

Ando and Zhang [2005] propose learning a shared structure across multiple related classification tasks over a single domain. Specifically, they consider T binary classification tasks each of which has its own linear classifier $f_t : \mathbb{R}^d \to \mathbb{R}$ mapping a d-dimensional feature vector $x \in \mathbb{R}^d$ to a classification score

$$f_t(x) := (u_t + \Theta v_t)^\top x \tag{4.31}$$

Here, $u_t \in \mathbb{R}^d$ and $v_t \in \mathbb{R}^m$ are task-specific parameters but $\Theta \in \mathbb{R}^{d \times m}$ is a global parameter shared by all classifiers $f_1 \dots f_T$ (we assume $m \leq \min\{d, T\}$). In particular, if Θ is zero then each classifier is an independent linear function $u_t^\top x$. The predicted label is the sign of the classification score $\text{sign}(f_t(x)) \in \{\pm 1\}$.

The parameter sharing makes the estimation problem challenging, but Ando and Zhang [2005] develop an effective alternating loss minimization algorithm using a variational property of SVD. To illustrate their method, let $L : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}$ be a convex loss function for classification, for instance the hinge loss $L(p, y) = \max(0, 1 - py)$.[3] For each task $t \in [T]$, we are given n_t labeled samples $(x^{(1|t)}, y^{(1|t)}) \dots (x^{(n_t|t)}, y^{(n_t|t)}) \in \mathbb{R}^d \times \{\pm 1\}$. A training objective is given by the following empirical loss minimization:

$$\min_{\substack{u_t \in \mathbb{R}^d \ \forall t \in [T] \\ v_t \in \mathbb{R}^m \ \forall t \in [T] \\ \Theta \in \mathbb{R}^{d \times m}}} \sum_{t=1}^{T} \left(r(u_t, v_t) + \frac{1}{n_t} \sum_{i=1}^{n_t} L\left((u_t + \Theta v_t)^\top x^{(i|t)}, y^{(i|t)} \right) \right) + R(\Theta) \tag{4.32}$$

[3] Ando and Zhang [2005] use a quadratically smoothed hinge loss called modified Huber:

$$L(p, y) = \begin{cases} \max(0, 1 - py)^2 & \text{if } py \geq -1 \\ -4py & \text{otherwise} \end{cases}$$

where $r(u_t, v_t)$ and $R(\Theta)$ are appropriate regularizers for the parameters. In other words, we minimize the sum of average losses (averaging is necessary since the amount of labeled data can vary greatly for each task).

Ando and Zhang [2005] choose a particular version of (4.32) to accomodate the use of SVD, given by

$$\min_{\substack{u_t \in \mathbb{R}^d \ \forall t \in [T] \\ v_t \in \mathbb{R}^m \ \forall t \in [T] \\ \Theta \in \mathbb{R}^{d \times m} : \Theta^\top \Theta = I_{m \times m}}} \sum_{t=1}^{T} \left(\lambda_t \left|\left| u_t \right|\right|_2^2 + \frac{1}{n_t} \sum_{i=1}^{n_t} L\left((u_t + \Theta v_t)^\top x^{(i|t)}, y^{(i|t)} \right) \right) \tag{4.33}$$

Note that the parameters v_t are not regularized: $r(u_t, v_t) = \lambda_t \left|\left| u_t \right|\right|_2^2$ for some hyperparameter $\lambda_t \geq 0$. Also, Θ is constrained to be an orthonormal matrix and is thus implicitly regularized. With an orthonormal $\Theta \in \mathbb{R}^{d \times m}$, the problem can be interpreted as finding an m-dimensional subspace of $\mathbb{R}^d$ which is predictive of labels across all T tasks. If $x_\Theta := \Theta^\top x$ denotes the m-dimensional representation of $x \in \mathbb{R}^d$ projected in the subspace range(Θ), every f_t computes the classification score of x by using this representation:

$$f_t(x) = u_t^\top x + v_t^\top x_\Theta$$

The objective (4.33) can be re-written with the change of variable $w_t := u_t + \Theta v_t$:

$$\min_{\substack{w_t \in \mathbb{R}^d \ \forall t \in [T] \\ v_t \in \mathbb{R}^m \ \forall t \in [T] \\ \Theta \in \mathbb{R}^{d \times m} : \Theta^\top \Theta = I_{m \times m}}} \sum_{t=1}^{T} \left(\lambda_t \left|\left| w_t - \Theta v_t \right|\right|_2^2 + \frac{1}{n_t} \sum_{i=1}^{n_t} L\left(w_t^\top x^{(i|t)}, y^{(i|t)} \right) \right) \tag{4.34}$$

Clearly, the original solution can be recovered from the solution of this formulation by $u_t = w_t - \Theta v_t$. The intuition behind considering (4.34) instead of (4.33) is that this allows us to separate the parameters Θ and v_t from the loss function $L(\cdot, \cdot)$ if we fix w_t.

Theorem 4.8.1 (Ando and Zhang [2005])**.** *Assume the parameters $w_t \in \mathbb{R}^d$ are fixed in (4.34) for all $t \in [T]$. Define $A := [\sqrt{\lambda_1} w_1 \dots \sqrt{\lambda_T} w_T] \in \mathbb{R}^{d \times T}$, and let $U = [u_1 \dots u_m] \in \mathbb{R}^{d \times m}$ be the left singular vectors of A corresponding to the largest $m \leq \min\{d, T\}$ singular values. Then the optimal solution for the parameters $\Theta \in \mathbb{R}^{d \times m}$ (under the orthogonality constraint $\Theta^\top \Theta = I_{m \times m}$) and $v_t \in \mathbb{R}^m$ is given by $\Theta^* = U$ and $v_t^* = U^\top w_t$ for all $t \in [T]$.*

Proof. Since w_t's are fixed, the objective (4.34) becomes

$$\min_{\substack{v_t \in \mathbb{R}^m \ \forall t \in [T] \\ \Theta \in \mathbb{R}^{d \times m} \colon \Theta^\top \Theta = I_{m \times m}}} \sum_{t=1}^{T} \lambda_t \left|\left| w_t - \Theta v_t \right|\right|_2^2$$

Note that for any value of orthonormal Θ, the optimal solution for each v_t is given by regression $(\Theta^\top \Theta)^{-1} \Theta^\top w_t = \Theta^\top w_t$. Thus we can plug in $v_t = \Theta^\top w_t$ in the objective to remove dependence on all variables except for Θ,

$$\min_{\Theta \in \mathbb{R}^{d \times m} \colon \Theta^\top \Theta = I_{m \times m}} \sum_{t=1}^{T} \lambda_t \left|\left| w_t - \Theta \Theta^\top w_t \right|\right|_2^2$$

Since $\left|\left| w_t - \Theta \Theta^\top w_t \right|\right|_2^2 = ||w_t||_2^2 - w_t^\top \Theta \Theta^\top w_t$, the objective is equivalent to

$$\max_{\Theta \in \mathbb{R}^{d \times m} \colon \Theta^\top \Theta = I_{m \times m}} \left|\left| A^\top \Theta \right|\right|_F^2$$

Thus the columns of an optimal Θ^* are given by the left singular vectors of A corresponding to the largest m singular values (Corollary 3.3.4). This also gives the claim on v_t^*. □

The theorem yields an alternating minimization strategy for optimizing (4.34). That is, iterate the following two steps until convergence:

- Fix Θ and v_t's: optimize the convex objective (4.34) (convex in w_t).
- Fix w_t's: compute optimal values of Θ and v_t in (4.34) with SVD (Theorem 4.8.1).

Note, however, that in general this does not guarantee the global optimality of the output parameters w_t, v_t, and Θ.

4.9 Non-Negative Matrix Factorization

Non-negative matrix factorization (NMF) is the following problem: given a non-negative matrix $A \in \mathbb{R}^{n \times d}$ (i.e., $A_{i,j} \geq 0$ for all i and j), and also a rank value $m \leq \min\{n, d\}$, find non-negative matrices $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{m \times d}$ such that $A = BC$ (the existence of such B and C is often given by task-specific assumptions). If M_i denotes the i-th row of matrix M, it can be easily verified that

$$A_i = \sum_{j=1}^{m} B_{i,j} \times C_j \tag{4.35}$$

In other words, a row of A is a (non-negative) linear combination of the rows of C. Thus NMF can be seen as finding a set of "dictionary" rows $C_1 \dots C_m$ that can be non-negatively added to realize all n rows of A. NMF arises naturally in many applications.

Example 4.9.1 (Image analysis [Lee and Seung, 1999])**.** *Suppose that each row of $A \in \mathbb{R}^{n \times d}$ is a facial image represented as a vector of d non-negative pixel values. Let $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{m \times d}$ be non-negative matrices such that $A = BC$. Then each facial image $A_i = B_{i,1}C_1 + \cdots + B_{i,m}C_m$ is a non-negative linear combination of m "basis images" $C_1 \dots C_m$.*

Example 4.9.2 (Document analysis [Blei *et al.*, 2003; Arora *et al.*, 2012b])**.** *Suppose that each row of $A \in \mathbb{R}^{n \times d}$ is a document represented as the document's distribution over d word types (thus non-negative). Let $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{m \times d}$ be non-negative matrices such that $A = BC$ and additionally that each row of B sums to 1. Then the word distribution under the i-th document $A_i = B_{i,1}C_1 + \cdots + B_{i,m}C_m$ is a convex combination of the word distributions under m "topics" $C_1 \dots C_m$.*

Note that while NMF is matrix decomposition, it is somewhat divorced from the theory of eigendecompositon. NMF is often implicit in parameter estimation of probabilistic models; for instance, learning the parameters of latent Dirichlet allocation can be seen as an implicit NMF [Arora *et al.*, 2012b].

Donoho and Stodden [2003] provide an intuitive geometric interpretation of NMF which also leads to an understanding of when an NMF is unique. Since all values involved in the characterization of A's row

$$A_i = \sum_{j=1}^{m} B_{i,j} \times C_j$$

are non-negative, we have that

- A_i is a vector residing in the positive orthant of $\mathbb{R}^d$.
- $C_1 \dots C_m$ are vectors also in the positive orthant of $\mathbb{R}^d$ such that any A_i can be expressed as their combination (scaled by scalars $B_{i,1} \dots B_{i,m} \geq 0$).

Hence NMF can be viewed as finding a *conical hull* enclosing all $A_1 \dots A_n$.[4] If $A_1 \dots A_n$ do not lie on every axis, there are infinitely many conical hulls that enclose $A_1 \dots A_n$ and hence NMF does not have a unique solution. Using this intuition, Donoho and Stodden [2003] provide a separability condition for when an NMF is unique.

Vavasis [2009] shows that NMF is NP-hard in general, but Arora *et al.* [2012b; 2012a] develop a provable NMF algorithm by exploiting a natural separability condition. In particular, Arora *et al.* [2012a] derive a purely combinatorial method for extracting dictionary rows $C_1 \dots C_m$ (shown in Figure 7.1) and successfully apply it to learning topic models. In Chapter 7, we extend this framework to learning hidden Markov models.

4.10 Tensor Decomposition

(We borrow the tensor notation in previous work [Lim, 2006; Anandkumar *et al.*, 2014].)

A p-th order tensor T is a p-dimensional array with entries $T_{i_1 \dots i_p} \in \mathbb{R}$ (e.g., a matrix is a second-order tensor). For simplicity, we only consider $p \leq 3$. A tensor $T \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ defines a function that maps input matrices V_1, V_2, V_3, where $V_i \in \mathbb{R}^{n_i \times m_i}$, to an output tensor $T(V_1, V_2, V_3) \in \mathbb{R}^{m_1 \times m_2 \times m_3}$ as follows:

$$[T(V_1, V_2, V_3)]_{i,j,k} := \sum_{i',j',k'} T_{i',j',k'} [V_1]_{i',i} [V_2]_{j',j} [V_3]_{k',k} \tag{4.36}$$

This nonlinear function is called **multilinear** since it is linear in V_i if all input matrices are fixed except V_i. A tensor $T \in \mathbb{R}^{n \times n \times n}$ is called **supersymmetric** if its entries are invariant to a permutation on indices, that is, $[T]_{i,j,k} = [T]_{i,k,j} = \cdots$. The **rank** of a supersymmetric T is defined to be the smallest non-negative integer m such that $T = \sum_{i=1}^{m} v_i v_i^\top v_i^\top$ for some vectors $v_1 \dots v_m \in \mathbb{R}^n$. Given vectors $\{u, v, w\}$, the notation $uv^\top w^\top$ denotes a rank-1 tensor with entries $[uv^\top w^\top]_{i,j,k} = [u]_i [v]_j [w]_k$ (analogous to the matrix outer product).

The above terms are similarly defined for the first- and second-order tensors (i.e., vectors and matrices). Note that a supersymmetric second-order tensor $M \in \mathbb{R}^{n_1 \times n_2}$ reduces to the standard definition of a symmetric matrix; the rank of M reduces to the number of

[4] When each row of B is contrained to sum to 1 (as in Example 4.9.2), then NMF can be viewed as finding a *convex hull* enclosing all $A_1 \dots A_n$.

nonzero eigenvalues (Proposition 3.2.1); and the tensor product (4.36) reduces to the matrix product $M(V_1, V_2) = V_1^\top M V_2$. The notation (4.36) also accommodates bypassing certain input positions with identity matrices. For example, the matrix-vector product can be expressed as $M(I_{n_1 \times n_1}, v) = Mv \in \mathbb{R}^{n_1}$. For a supersymmetric tensor $T \in \mathbb{R}^{n \times n \times n}$, a unit **eigenvector** $v \in \mathbb{R}^n$ of T is a unit-length vector with a corresponding **eigenvalue** $\lambda \in \mathbb{R}$ such that

$$T(I_{n \times n}, v, v) = \lambda v \tag{4.37}$$

which is a direct analogue of the matrix counterpart (3.18).

Tensor decomposition is often useful (e.g., in the method of moments). Unfortunately, many of the tools developed in conventional linear algebra do not generalize to higher-order tensors. For instance, while a symmetric matrix always has an efficiently computable eigendecomposition (Theorem 3.2.6), it is not the case for a higher-order supersymmetric tensor T [Qi, 2005]. While the low-rank matrix approximation problem can be solved efficiently using SVD (Section 4.2), computing even a rank-1 approximation of T:

$$\min_{u,v,w} \left|\left| T - uv^\top w^\top \right|\right|_F \tag{4.38}$$

(where $||T||_F = \sqrt{\sum_{i,j,k} T_{i,j,k}^2}$) is NP-hard (Theorem 1.13, Hillar and Lim [2013]).

Anandkumar *et al.* [2014] show that the problem is much more manageable if tensors are *orthogonal* in addition to being supersymmetric. Specifically, they assume a supersymmetric and orthogonal tensor $T \in \mathbb{R}^{n \times n \times n}$ of rank-m, that is,

$$T = \sum_{i=1}^{m} \lambda_i v_i v_i^\top v_i^\top \tag{4.39}$$

where $v_1 \ldots v_m \in \mathbb{R}^n$ are orthonormal and $\lambda_1 \geq \ldots \geq \lambda_m > 0$. Since $T(I_{n \times n}, v_i, v_i) = \lambda_i v_i$, each (v_i, λ_i) is an eigenvector-eigenvalue pair. In this case, a random initial vector $v \in \mathbb{R}^n$ under the **tensor power iterations**:

$$v \mapsto \frac{T(I_{n \times n}, v, v)}{||T(I_{n \times n}, v, v)||_2} \tag{4.40}$$

converges to some v_i (Theorem 4.1, Anandkumar *et al.* [2014]). Thus the eigencomponents of T can be extracted through the power iteration method similar to the matrix case in

Figure 3.3. Note a subtle difference: the extracted eigencomponents may not be in a descending order of eigenvalues, since the iteration (4.40) converges to *some* eigenvector v_i, not necessarily v_1.

Another important contribution of Anandkumar *et al.* [2014] is a scheme to *orthogonalize* a rank-m supersymmetric tensor $T = \sum_{i=1}^{m} w_i u_i u_i^\top u_i^\top$ where $w_1 \geq \ldots \geq w_m > 0$ but $u_1 \ldots u_m$ are not necessarily orthogonal (but assumed to be linearly independent) with a corresponding rank-m symmetric matrix $M = \sum_{i=1}^{m} w_i u_i u_i^\top$. Let $W \in \mathbb{R}^{n \times m}$ be a whitening matrix for M, that is,

$$M(W, W) = \sum_{i=1}^{m} (\sqrt{w_i} W^\top u_i)(\sqrt{w_i} W^\top u_i)^\top = I_{m \times m}$$

For instance, one can set $W = V \Lambda^{-1/2}$ where $M = V \Lambda V^\top$ is a rank-m SVD of M. This implies that $\sqrt{w_1} W^\top u_1 \ldots \sqrt{w_m} W^\top u_m \in \mathbb{R}^m$ are orthonormal. Then the $(m \times m \times m)$ tensor

$$T(W, W, W) = \sum_{i=1}^{m} \frac{1}{\sqrt{w_i}} (\sqrt{w_i} W^\top u_i)(\sqrt{w_i} W^\top u_i)^\top (\sqrt{w_i} W^\top u_i)^\top$$

is orthogonal and is decomposable by the tensor power iteration method $T(W, W, W) = \sum_{i=1}^{m} \lambda_i v_i v_i^\top v_i^\top$. The original variables can be recovered as $w_i = 1/\lambda_i^2$ and $u_i = \lambda_i (W^\top)^+ v_i$.

In summary, the method of Anandkumar *et al.* [2014] can be used to recover linearly independent $u_1 \ldots u_m \in \mathbb{R}^n$ and positive scalars $w_1 \ldots w_m \in \mathbb{R}$ from supersymmetric second- and third-order tensors of rank m:

$$M = \sum_{i=1}^{m} w_i u_i u_i^\top \tag{4.41}$$

$$T = \sum_{i=1}^{m} w_i u_i u_i^\top u_i^\top \tag{4.42}$$

Anandkumar *et al.* [2014] show that this can be used as a *learning algorithm* for a variety of latent-variable models. For instance, consider learning a bag-of-words model with n word types and m topic types. The task is to estimate the model parameters

- $w_i \in \mathbb{R}$: probability of topic $i \in [m]$
- $u_i \in \mathbb{R}^n$: conditional distribution over n word types given topic $i \in [m]$

Then it is easily verifiable that the observable quantities $M \in \mathbb{R}^{n\times n}$ and $T \in \mathbb{R}^{n\times n\times n}$ where $M_{i,j}$ is the probability of words $i, j \in [n]$ occurring together in a document (not necessarily consecutively) and $T_{i,j,k}$ is the probability of words $i, j, k \in [n]$ occurring together in a document (not necessarily consecutively) have the form (4.41) and (4.42). Thus the parameters (w_i, u_i) can be estimated by tensor decomposition.

We mention that there is ongoing progress in tensor decomposition. For example, see Kuleshov *et al.* [2015] for a decomposition scheme applicable to a wider class of tensors.

Part II

Inducing Lexical Representations

Chapter 5

Word Clusters Under Class-Based Language Models

This chapter is adapted from joint work with Do-kyum Kim, Michael Collins, and Daniel Hsu entitled "A Spectral Algorithm for Learning Class-Based n-gram Models of Natural Language" [Stratos *et al.*, 2014].

The Brown clustering algorithm [Brown *et al.*, 1992] is widely used in NLP to derive lexical representations that are then used to improve performance on various NLP problems. The algorithm assumes an underlying model that is essentially an HMM, with the restriction that each word in the vocabulary is emitted from a single state. A greedy, bottom-up method is then used to find the clustering; this method does not have a guarantee of finding the correct underlying clustering. In this work, we describe a new algorithm for clustering under the Brown *et al.* model. The method relies on two steps: first, the use of canonical correlation analysis to derive a low-dimensional representation of words; second, a bottom-up hierarchical clustering over these representations. We show that given a sufficient number of training examples sampled from the Brown *et al.* model, the method is guaranteed to recover the correct clustering. Experiments show that the method recovers clusters of comparable quality to the algorithm of Brown *et al.* [1992], but is an order of magnitude more efficient.

In this chapter, $||v||$ denotes the Euclidean norm of a vector v ($||v|| = ||v||_2$) and $||M||$

denotes the spectral norm of a matrix M ($||M|| = ||M||_2$).

5.1 Introduction

There has recently been great interest in the natural language processing (NLP) community in methods that derive lexical representations from large quantities of unlabeled data [Brown *et al.*, 1992; Pereira *et al.*, 1993; Ando and Zhang, 2005; Liang, 2005; Turian *et al.*, 2010; Dhillon *et al.*, 2011b; Collobert *et al.*, 2011; Mikolov *et al.*, 2013b,c]. These representations can be used to improve accuracy on various NLP problems, or to give significant reductions in the number of training examples required for learning. The Brown clustering algorithm [Brown *et al.*, 1992] is one of the most widely used algorithms for this task. Brown clustering representations have been shown to be useful in a diverse set of problems including named-entity recognition [Miller *et al.*, 2004; Turian *et al.*, 2010], syntactic chunking [Turian *et al.*, 2010], parsing [Koo *et al.*, 2008], and language modeling [Kneser and Ney, 1993; Gao *et al.*, 2001].

The Brown clustering algorithm assumes a model that is essentially a hidden Markov model (HMM), with a restriction that each word in the vocabulary can only be emitted from a single state in the HMM (i.e, there is a deterministic mapping from words to underlying states). The algorithm uses a greedy, bottom-up method in deriving the clustering. This method is a heuristic, in that there is no guarantee of recovering the correct clustering. In practice, the algorithm is quite computationally expensive: for example in our experiments, the implementation of Liang [2005] takes over 22 hours to derive a clustering from a dataset with 205 million tokens and 300,000 distinct word types.

We introduce a new algorithm for clustering under the Brown *et al.* [1992] model (henceforth, the Brown model). Crucially, under an assumption that the data is generated from the Brown model, our algorithm is guaranteed to recover the correct clustering when given a sufficient number of training examples (see the theorems in Section 5.4). The algorithm draws on ideas from canonical correlation analysis (CCA) and agglomerative clustering, and has the following simple form:

1. Estimate a normalized covariance matrix from a corpus and use singular value de-

composition (SVD) to derive low-dimensional vector representations for word types (Figure 5.4).

2. Perform a bottom-up hierarchical clustering of these vectors (Figure 5.7).

In our experiments, we find that our clusters are comparable to the Brown clusters in improving the performance of a supervised learner, but our method is significantly faster. For example, both our clusters and Brown clusters improve the F1 score in named-entity recognition (NER) by 2-3 points, but the runtime of our method is around 10 times faster than the Brown algorithm (Table 5.3).

The chapter is structured as follows. In Section 5.2, we discuss related work. In Section 5.3, we define the Brown model. In Section 5.4, we present the main result and describe the algorithm. In Section 5.5, we report experimental results.

5.2 Background

5.2.1 The Brown Clustering Algorithm

The Brown clustering algorithm [Brown *et al.*, 1992] has been used in many NLP applications [Koo *et al.*, 2008; Miller *et al.*, 2004; Liang, 2005]. We briefly describe the algorithm below. For details, see Section A.1.

The input to the algorithm is a corpus of text with N tokens of n distinct word types. The algorithm initializes each word type as a distinct cluster, and repeatedly merges the pair of clusters that cause the smallest decrease in the likelihood of the corpus according to a discrete hidden Markov model (HMM). The observation parameters of this HMM are assumed to satisfy a certain disjointedness condition (Assumption 5.3.1). We will explicitly define the model in Section 5.3.

At the end of the algorithm, one obtains a hierarchy of word types which can be represented as a binary tree as in Figure 5.2. Within this tree, each word is uniquely identified by its path from the root, and this path can be compactly represented with a bit string. In order to obtain a clustering of the words, we select all nodes at a certain depth from the root of the hierarchy. For example, in Figure 5.2 we might select the four nodes at

Input: corpus with N tokens of n distinct word types $w^{(1)}, \ldots, w^{(n)}$ ordered by decreasing frequency; number of clusters m.

Output: hierarchical clustering of $w^{(1)}, \ldots, w^{(n)}$.

1. Initialize active clusters $\mathcal{C} = \{\{w^{(1)}\}, \ldots, \{w^{(m)}\}\}$.
2. For $i = m + 1$ to $n + m - 1$:
   (a) If $i \leq n$: set $\mathcal{C} = \mathcal{C} \cup \{\{w^{(i)}\}\}$.
   (b) Merge $c, c' \in \mathcal{C}$ that cause the smallest decrease in the likelihood of the corpus.

Figure 5.1: A standard implementation of the Brown clustering algorithm. See Figure A.1 for more details.

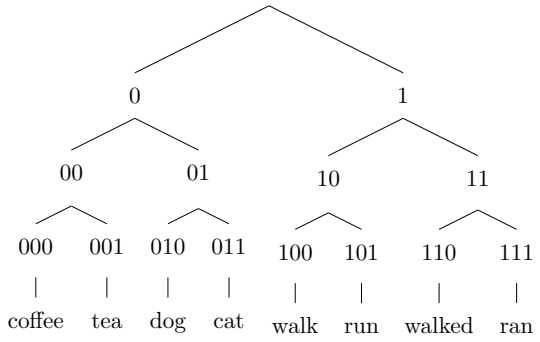


Figure 5.2: An example of a Brown word-cluster hierarchy. Each node in the tree is labeled with a bit string indicating the path from the root node to that node, where 0 indicates a left branch and 1 indicates a right branch.

depth 2 from the root, yielding the clusters {`coffee`, `tea`}, {`dog`, `cat`}, {`walk`, `run`}, and {`walked`, `ran`}. Note that the same clustering can be obtained by truncating each word's bit string to a 2-bit prefix. By using prefixes of various lengths, we can produce clusterings of different granularities.

A naive implementation of this algorithm has runtime $O(n^5)$. Brown *et al.* [1992] propose a technique to reduce the runtime to $O(n^3)$. Since this is still not acceptable for large values of n, a common trick used for practical implementation is to specify the number of

active clusters $m \ll n$, for example, $m = 1000$. A sketch of this implementation is shown in Figure 5.1. Using this technique, it is possible to achieve $O(N + nm^2)$ runtime. We note that our algorithm in Figure 5.7 has a similar form and asymptotic runtime, but is empirically much faster. We discuss this issue in Section 5.5.3.1.

In this work, we present a very different algorithm for deriving a word hierarchy based on the Brown model. In all our experiments, we compared our method against the highly optimized implementation of the Brown algorithm in Figure 5.1 by Liang (2005).

5.2.2 CCA and Agglomerative Clustering

Our algorithm in Figure 5.4 operates in a fashion similar to the mechanics of CCA. CCA is a statistical technique used to maximize the correlation between a pair of random variables [Hotelling, 1936]. A central operation in CCA to achieve this maximization is SVD; in this work, we also critically rely on SVD to recover the desired parameters.

Recently, it has been shown that one can use CCA-style algorithms, so-called spectral methods, to learn HMMs in polynomial sample/time complexity [Hsu *et al.*, 2012]. These methods will be important to our goal since the Brown model can be viewed as a special case of an HMM.

We briefly note that one can view our approach from the perspective of spectral clustering [Ng *et al.*, 2002]. A spectral clustering algorithm typically proceeds by constructing a graph Laplacian matrix from the data and performing a standard clustering algorithm (e.g., k-means) on reduced-dimensional points that correspond to the top eigenvalues of the Laplacian. We do not make use of a graph Laplacian, but we do make use of spectral methods for dimensionality reduction before clustering.

Agglomerative clustering refers to hierarchical grouping of n points using a bottom-up style algorithm [Ward Jr, 1963; Shanbehzadeh and Ogunbona, 1997]. It is commonly used for its simplicity, but a naive implementation requires $O(dn^3)$ time where d is the dimension of a point. Franti *et al.* [2000] presented a faster algorithm that requires $O(\gamma dn^2)$ time where γ is a data-dependent quantity which is typically much smaller than n. In our work, we use a variant of this last approach that has runtime $O(\gamma dmn)$ where $m \ll n$ is the number of active clusters we specify (Figure 5.7). We also remark that under our derivation, the

dimension d is always equal to m, thus we express the runtime simply as $O(\gamma nm^2)$.

5.3 Brown Model Definition

A Brown model is a 5-tuple (n, m, π, t, o) for integers n, m and functions π, t, o where

- $[n]$ is a set of states that represent word types.
- $[m]$ is a set of states that represent clusters.
- $\pi(c)$ is the probability of generating $c \in [m]$ in the first position of a sequence.
- $t(c'|c)$ is the probability of generating $c' \in [m]$ given $c \in [m]$.
- $o(x|c)$ is the probability of generating $x \in [n]$ given $c \in [m]$.

In addition, the model makes the following assumption on the parameters $o(x|c)$. This assumption comes from Brown *et al.* [1992] who require that the word clusters partition the vocabulary.

Assumption 5.3.1 (Brown *et al.* [1992] assumption)**.** *For each $x \in [n]$, there is a unique $C(x) \in [m]$ such that $o(x|C(x)) > 0$ and $o(x|c) = 0$ for all $c \neq C(x)$.*

In other words, the model is a discrete HMM with a many-to-one deterministic mapping $C : [n] \to [m]$ from word types to clusters. Under the model, a sequence of N tokens $(x_1, \ldots, x_N) \in [n]^N$ has probability

$$p(x_1, \ldots, x_N) = \pi(C(x_1)) \times \prod_{i=1}^{N} o(x_i|C(x_i)) \times \prod_{i=1}^{N-1} t(C(x_{i+1})|C(x_i))$$

An equivalent definition of a Brown model is given by organizing the parameters in matrix form. Under this definition, a Brown model has parameters (π, T, O) where $\pi \in \mathbb{R}^m$ is a vector and $T \in \mathbb{R}^{m \times m}, O \in \mathbb{R}^{n \times m}$ are matrices whose entries are set to:

- $\pi_c = \pi(c)$ for $c \in [m]$
- $T_{c',c} = t(c'|c)$ for $c, c' \in [m]$
- $O_{x,c} = o(x|c)$ for $c \in [m], x \in [n]$

We will assume throughout that T, O have rank m. The following is an equivalent reformulation of Assumption 5.3.1 and will be important to the derivation of our algorithm.

Assumption 5.3.2 (Brown *et al.* [1992] assumption)**.** *Each row of O has exactly one non-zero entry.*

5.4 Clustering Under the Brown Model

In this section, we develop a method for clustering words based on the Brown model. The resulting algorithm is a simple two-step procedure: an application of SVD followed by agglomerative hierarchical clustering in Euclidean space.

5.4.1 An Overview of the Approach

Suppose the parameter matrix O is known. Under Assumption 5.3.2, a simple way to recover the correct word clustering is as follows:

1. Compute $\bar{M} \in \mathbb{R}^{n \times m}$ whose rows are the rows of $\sqrt{O}$ normalized to have length 1.
2. Put words x, x' in the same cluster iff $\bar{M}_x = \bar{M}_{x'}$, where $\bar{M}_x$ is the x-th row of $\bar{M}$.

This works because Assumption 5.3.2 implies that the rows of $\sqrt{O}$ corresponding to words from the same cluster lie along the same coordinate-axis in $\mathbb{R}^m$. Row-normalization puts these rows precisely at the standard basis vectors. See Figure 5.3 for illustration.

In Section 5.4.2, we prove that the rows of $\sqrt{O}$ can be recovered, up to an orthogonal transformation $Q \in \mathbb{R}^{m \times m}$, just from unigram and bigram word probabilities (which can be estimated from observed sequences). It is clear that the correctness of the above procedure is unaffected by the orthogonal transformation. Let M denote the row-normalized form of $\sqrt{O}Q^\top$: then M still satisfies the property that $M_x = M_{x'}$ iff x, x' belong to the same cluster. We give an algorithm to estimate this M from a sequence of words in Figure 5.4.

5.4.2 Spectral Estimation of Observation Parameters

To derive a method for estimating the observation parameter $\sqrt{O}$ (up to an orthogonal transformation), we first define the following random variables to model a single random

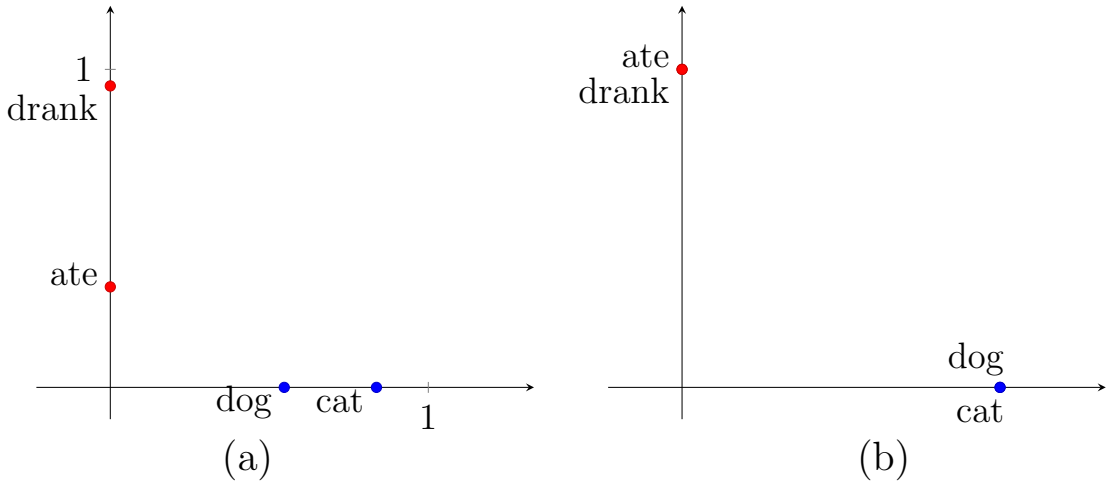


Figure 5.3: Illustration of our clustering scheme. (a) Original rows of $\sqrt{O}$. (b) After row-normalization.

sentence. Let $(X_1, \ldots, X_N) \in [n]^N$ be a random sequence of tokens drawn from the Brown model, along with the corresponding (hidden) cluster sequence $(C_1, \ldots, C_N) \in [m]^N$; independently, pick a position $I \in [N-1]$ uniformly at random. Let $B \in \mathbb{R}^{n \times n}$ be a matrix of bigram probabilities, $u, v \in \mathbb{R}^n$ vectors of unigram probabilities, and $\tilde{\pi} \in \mathbb{R}^m$ a vector of cluster probabilities:

$$\begin{aligned} B_{x,x'} &:= P(X_I = x, X_{I+1} = x') && \forall x, x' \in [n] \\ u_x &:= P(X_I = x) && \forall x \in [n] \\ v_x &:= P(X_{I+1} = x) && \forall x \in [n] \\ \tilde{\pi}_c &:= P(C_I = c) && \forall c \in [m]. \end{aligned}$$

We assume that $\mathrm{diag}(\tilde{\pi})$ has rank m; note that this assumption is weaker than requiring $\mathrm{diag}(\pi)$ to have rank m. We will consider a matrix $\Omega \in \mathbb{R}^{n \times n}$ defined as

$$\Omega := \mathrm{diag}(u)^{-1/2} B \, \mathrm{diag}(v)^{-1/2} \tag{5.1}$$

Theorem 5.4.1. *Let $U \in \mathbb{R}^{n \times m}$ be the matrix of m left singular vectors of Ω corresponding to nonzero singular values. Then there exists an orthogonal matrix $Q \in \mathbb{R}^{m \times m}$ such that $U = \sqrt{O} Q^\top$.*

To prove Theorem 5.4.1, we need to examine the structure of the matrix Ω. The following matrices $A, \tilde{A} \in \mathbb{R}^{n \times m}$ will be important for this purpose:

$$A = \operatorname{diag}(O\tilde{\pi})^{-1/2} O \operatorname{diag}(\tilde{\pi})^{1/2}$$
$$\tilde{A} = \operatorname{diag}(OT\tilde{\pi})^{-1/2} OT \operatorname{diag}(\tilde{\pi})^{1/2}$$

The first lemma shows that Ω can be decomposed into A and $\tilde{A}^\top$.

Lemma 5.4.2. $\Omega = A\tilde{A}^\top$.

Proof. It can be algebraically verified from the definition of B, u, v that $B = O \operatorname{diag}(\tilde{\pi})(OT)^\top$, $u = O\tilde{\pi}$, and $v = OT\tilde{\pi}$. Plugging in these expressions in Eq. (5.1), we have

$$\Omega = \operatorname{diag}(O\tilde{\pi})^{-1/2} O \operatorname{diag}(\tilde{\pi})^{1/2}$$
$$\left(\operatorname{diag}(OT\tilde{\pi})^{-1/2} OT \operatorname{diag}(\tilde{\pi})^{1/2} \right)^\top = A\tilde{A}^\top. \qquad \square$$

The second lemma shows that A is in fact the desired matrix. The proof of this lemma crucially depends on the disjoint-cluster assumption of the Brown model.

Lemma 5.4.3. *$A = \sqrt{O}$ and $A^\top A = I_{m \times m}$.*

Proof. By Assumption 5.3.2, the x-th entry of $O\tilde{\pi}$ has value $O_{x,C(x)} \times \tilde{\pi}_{C(x)}$, and the $(x, C(x))$-th entry of $O \operatorname{diag}(\tilde{\pi})^{1/2}$ has value $O_{x,C(x)} \times \sqrt{\tilde{\pi}_{C(x)}}$. Thus the $(x, C(x))$-th entry of A is

$$A_{x,C(x)} = \frac{O_{x,C(x)} \sqrt{\tilde{\pi}_{C(x)}}}{\sqrt{O_{x,C(x)} \tilde{\pi}_{C(x)}}} = \sqrt{O_{x,C(x)}}$$

The columns of A have disjoint supports since A has the same sparsity pattern as O. Furthermore, the l_2 (Euclidean) norm of any column of A is the l_1 norm of the corresponding column of O. This implies $A^\top A = I_{m \times m}$ $\square$

Now we give a proof of the main theorem.

Proof of Theorem 5.4.1. The orthogonal projection matrix onto range(Ω) is given by $UU^\top$ and also by $\Omega(\Omega^\top \Omega)^+ \Omega^\top$. Hence from Lemma 5.4.2 and 5.4.3, we have

$$\begin{aligned} UU^\top &= \Omega(\Omega^\top \Omega)^+ \Omega^\top \\ &= (A\tilde{A}^\top)(\tilde{A}A^\top A\tilde{A}^\top)^+ (A\tilde{A}^\top)^\top \\ &= (A\tilde{A}^\top)(\tilde{A}\tilde{A}^\top)^+ (A\tilde{A}^\top)^\top = A\Pi A^\top \end{aligned}$$

Input: sequence of $N \geq 2$ words $(x_1, \ldots, x_N) \in [n]^N$; number of clusters m; smoothing parameter κ.

Output: matrix $\hat{M} \in \mathbb{R}^{n \times m}$ defining $f : x \mapsto \hat{M}_x \ \forall x \in [n]$.

1. Compute $\hat{B} \in \mathbb{R}^{n \times n}$, $\hat{u} \in \mathbb{R}^n$, and $\hat{v} \in \mathbb{R}^n$ where

$$\hat{B}_{x,x'} := \frac{1}{N-1} \sum_{i=1}^{N-1} [[x_i = x, x_{i+1} = x']] \qquad \forall x, x' \in [n]$$

$$\hat{u}_x := \frac{1}{N-1} \sum_{i=1}^{N-1} [[x_i = x]] + \frac{\kappa}{N-1} \qquad \forall x \in [n]$$

$$\hat{v}_x := \frac{1}{N-1} \sum_{i=1}^{N-1} [[x_{i+1} = x]] + \frac{\kappa}{N-1} \qquad \forall x \in [n]$$

2. Compute rank-m SVD of the sparse matrix

$$\hat{\Omega} := \text{diag}(\hat{u})^{-1/2} \ \hat{B} \ \text{diag}(\hat{v})^{-1/2}.$$

   Let $\hat{U} \in \mathbb{R}^{n \times m}$ be a matrix of m left singular vectors of $\hat{\Omega}$ corresponding to the m largest singular values.

3. Let $\hat{M}$ be the result of normalizing every row of $\hat{U}$ to have length 1.

Figure 5.4: Estimation of M from samples.

where $\Pi = \tilde{A}(\tilde{A}^\top \tilde{A})^+ \tilde{A}^\top$ is the orthogonal projection matrix onto range$(\tilde{A})$. But since $\tilde{A}$ has rank m, range$(\tilde{A}) = \mathbb{R}^m$ and thus $\Pi = I_{m \times m}$. Then we have $UU^\top = AA^\top$ where both U and A have orthogonal columns (Lemma 5.4.3). This implies that there is an orthogonal matrix $Q \in \mathbb{R}^{m \times m}$ such that $U = AQ^\top$. □

5.4.3 Estimation from Samples

In Figure 5.4, we give an algorithm for computing an estimate $\hat{M}$ of M from a sample of words $(x_1, \ldots, x_N) \in [n]^N$ (where M is described in Section 5.4.1). The algorithm estimates unigram and bigram word probabilities u, v, B to form a plug-in estimate $\hat{\Omega}$ of Ω (defined

in Eq. (5.1)), computes a low-rank SVD of a sparse matrix, and normalizes the rows of the resulting left singular vector matrix.

The following theorem implies the consistency of our algorithm, assuming the consistency of $\hat{\Omega}$.

Theorem 5.4.4. *Let* $\varepsilon := \left|\left|\hat{\Omega} - \Omega\right|\right| / \sigma_m(\Omega)$. *If* $\varepsilon \leq 0.07 \min_{x \in [n]} \sqrt{o(x|C(x))}$, *then the word embedding* $f(x) := \hat{M}_x$ *(where* $\hat{M}_x$ *is the* x*-th row of* $\hat{M}$*) satisfies the following property: for all* $x, x', x'' \in [n]$,

$$C(x) = C(x') \neq C(x'') \quad \Longrightarrow \quad \left|\left|f(x) - f(x')\right|\right| < \left|\left|f(x) - f(x'')\right|\right|$$

(i.e., the embedding of any word x *is closer to that of other words* x' *from the same cluster than it is to that of any word* x'' *from a different cluster).*

The property established by Theorem 5.4.4 (proved in Section A.3) allows many distance-based clustering algorithms (e.g., single-linkage, average-linkage) to recover the correct clustering:

Corollary 5.4.5. *The ground-truth clustering* $C : [n] \to [m]$ *in the Brown model is some pruning (of size* m*) of a tree over the vocabulary produced by single-linkage or average-linkage hierarchical clustering applied on the word embeddings* $f(x)$ *in Theorem 5.4.4.*

Proof. By Theorem 5.4.4, the Euclidiean distance for word embeddings $f(x)$ is a symmetric similarity function satisfying the strict separation property (Property 1, Balcan *et al.*, 2008) with respect to C, thus C is some pruning of a tree produced by single-linkage clustering (Theorem 2, Balcan *et al.*, 2008). Since it satisfies the strict separation property, it also satisfies the strong stability property (Property 2, Balcan *et al.*, 2008), thus C is also some pruning of a tree produced by average-linkage clustering (Theorem 8, Balcan *et al.*, 2008). □

Moreover, it is possible to establish finite sample complexity bounds for the estimation error of $\hat{\Omega}$. For simplicity, instead of estimating B, u, and v from a single long sentence, we estimate them (via maximum likelihood) using N i.i.d. sentences, each of length 2. Let $\kappa_m(\Omega) := \sigma_1(\Omega)/\sigma_m(\Omega)$ be the rank-m condition number of Ω. Let $u_{\min} := \min_x u_x$ and

$v_{\min} := \min_x v_x$. Define

$$n_1 := \max_{x \in [n]} \sum_{x' \in [n]} \frac{P(C_2 = C(x') | C_1 = C(x))}{P(C_2 = C(x'))} \qquad n_2 := \max_{x' \in [n]} \sum_{x \in [n]} \frac{P(C_2 = C(x') | C_1 = C(x))}{P(C_2 = C(x'))}$$

Theorem 5.4.6. *There is an absolute constant $c'' > 0$ such that for any $\varepsilon \in (0,1)$, if*

$$N \geq c'' \cdot \frac{\kappa_m(\Omega)^2 \log(n/\delta) \max\{n_1, n_2, 1/u_{\min}, 1/v_{\min}\}}{\varepsilon^2}$$

then with probability at least $1 - \delta$, $\left|\left|\hat{\Omega} - \Omega\right|\right| / \sigma_m(\Omega) \leq \varepsilon$.

Theorem 5.4.4 together with Theorem 5.4.6 (proved in Section A.4) imply that given enough samples from the Brown model, the word embeddings induced by our algorithm in Figure 5.4 can be used to infer the underlying clusters of word types.

In practice, it is important to regularize the estimates $\hat{u}$ and $\hat{v}$ using a smoothing parameter $\kappa \geq 0$. This can be viewed as adding pseudocounts to alleviate the noise from infrequent words, and has a significant effect on the resulting representations. The practical importance of smoothing is also seen in previous methods using CCA [Hardoon *et al.*, 2004].

Another practical consideration is the use of richer context. So far, the context used for the token X_I is just the next token X_{I+1}; hence, the spectral estimation is based just on unigram and bigram probabilities. However, it is straightforward to generalize the technique to use other context—details are in Section A.2. For instance, if we use the previous and next tokens (X_{I-1}, X_{I+1}) as context, then we form $\hat{\Omega} \in \mathbb{R}^{n \times 2n}$ from $\hat{B} \in \mathbb{R}^{n \times 2n}$, $\hat{u} \in \mathbb{R}^n$, $\hat{v} \in \mathbb{R}^{2n}$; however, we still extract $\hat{M} \in \mathbb{R}^{n \times m}$ from $\hat{\Omega}$ in the same way to form the word embedding.

5.4.4 Agglomerative Clustering

As established in Theorem 5.4.4, the word embedding obtained by mapping words to their corresponding rows of $\hat{M}$ permits distance-based clustering algorithms to recover the correct clustering. However, with small sample sizes and model approximation errors, the property from Theorem 5.4.4 may not hold exactly. Therefore, we propose to compute a hierarchical clustering of the word embeddings, with the goal of finding the correct clustering (or at least a good clustering) as some pruning of the resulting tree. Simple agglomerative clustering algorithms can provably recover the correct clusters when idealized properties (such as that

from Theorem 5.4.4) hold [Balcan *et al.*, 2008], and can also be seen to be optimizing a sensible objective regardless [Dasgupta and Long, 2005]. These algorithms also yield a hierarchy of word types—just as the original Brown clustering algorithm.

We use a form of average-linkage agglomerative clustering called Ward's algorithm [Ward Jr, 1963], which is particularly suited for hierarchical clustering in Euclidean spaces. In this algorithm, the cost of merging clusters c and c' is defined as

$$d(c, c') = \frac{|c||c'|}{|c| + |c'|} ||\mu_c - \mu_{c'}||^2 \tag{5.2}$$

where $|c|$ refers to the number of elements in cluster c and $\mu_c = |c|^{-1} \sum_{u \in c} u$ is the mean of cluster c. The algorithm starts with every point (word) in its own cluster, and repeatedly merges the two clusters with cheapest merge cost.

Figure 5.7 sketches a variant of Ward's algorithm that only considers merges among (at most) $m + 1$ clusters at a time. The initial $m + 1$ (singleton) clusters correspond to the $m + 1$ most frequent words (according to $\hat{u}$); after a merge, the next most frequent word (if one exists) is used to initialize a new singleton cluster. This heuristic is also adopted by the original Brown algorithm, and is known to be very effective.

Using an implementation trick from Franti *et al.* [2000], the runtime of the algorithm is $O(\gamma n m^2)$, where γ is a data-dependent constant often much smaller than m, as opposed to $O(nm^3)$ in a naive implementation in which we search for the closest pair among $O(m^2)$ pairs at every merge.

The basic idea of Franti *et al.* [2000] is the following. For each cluster, we keep an estimation of the lower bound on the distance to the nearest cluster. We also track if this lower bound is tight; in the beginning, every bound is tight. When searching for the nearest pair, we simply look for a cluster with the smallest lower bound among m clusters instead of $O(m^2)$ cluster pairs. If the cluster has a tight lower bound, we merge it with its nearest cluster. Otherwise, we tighten its bound and again look for a cluster with the smallest bound. Thus γ is the effective number of searches at each iteration. At merge, the bound of a cluster whose nearest cluster is either of the two merged clusters becomes loose. We report empirical values of γ in our experimental study (see Table 5.3).

5.5 Experiments

To evaluate the effectiveness of our approach, we used the clusters from our algorithm as additional features in supervised models for NER. We then compared the improvement in performance and also the time required to derive the clusters against those of the Brown clustering algorithm. Additionally, we examined the mutual information (MI) of the derived clusters on the training corpus:

$$\sum_{c,c'} \frac{\text{count}(c,c')}{N} \log \frac{\text{count}(c,c')N}{\text{count}(c)\text{count}(c')} \tag{5.3}$$

where N is the number of tokens in the corpus, $\text{count}(c)$ is the number of times cluster c appears, and $\text{count}(c,c')$ is the number of times clusters c,c' appear consecutively. Note that this is the quantity the Brown algorithm directly maximizes [Brown *et al.*, 1992].

5.5.1 Experimental Settings

For NER experiments, we used the scripts provided by Turian *et al.* [2010]. We used the greedy perceptron for NER experiments [Ratinov and Roth, 2009] using the standard features as our baseline models. We used the CoNLL 2003 dataset for NER with the standard train/dev/test split.

For the choice of unlabeled text data, we used the Reuters-RCV1 corpus which contains 205 million tokens with 1.6 million distinct word types. To keep the size of the vocabulary manageable and also to reduce noise from infrequent words, we used only a selected number of the most frequent word types and replaced all other types in the corpus with a special token. For the size of the vocabulary, we used 50,000 and 300,000.

Our algorithm can be broken down into two stages: the SVD stage (Figure 5.4) and the clustering stage (Figure 5.7). In the SVD stage, we need to choose the number of clusters m and the smoothing parameter κ. As mentioned, we can easily define Ω to incorporate information beyond one word to the right. We experimented with the following configurations for context:

1. R1 ($\Omega \in \mathbb{R}^{n \times n}$): 1 word to the right. This is the version presented in Figure 5.4.

2. LR1 ($\Omega \in \mathbb{R}^{n \times 2n}$): 1 word to the left/right.

	vocab	context	dev	test
Baseline	—	—	90.03	84.39
Spectral	50k	LR1	92	86.72
($\kappa = 200$)	300k	LR2	92.31	87.76
Brown	50k	—	92	88.56
	300k		92.68	88.76

Table 5.1: Performance gains in NER.

	vocab size	context	MI
Spectral	50k	LR2	1.48
($\kappa = 5000$)	300k	LR2	1.54
Brown	50k	—	1.52
	300k	—	1.6

Table 5.2: Mutual information computed as in Eq. (5.3) on the RCV1 corpus.

3. LR2 ($\Omega \in \mathbb{R}^{n \times 4n}$): 2 words to the left/right.

5.5.2 Comparison to the Brown Algorithm: Quality

There are multiple ways to evaluate the quality of clusters. We considered the improvement in the F1 score in NER from using the clusters as additional features. We also examined the MI on the training corpus. For all experiments in this section, we used 1,000 clusters for both the spectral algorithm (i.e., $m = 1000$) and the Brown algorithm.

5.5.2.1 NER

In NER, there is significant improvement in the F1 score from using the clusters as additional features (Table 5.1). The dev F1 score is improved from 90.03 to 92 with either spectral or Brown clusters using 50k vocabulary size; it is improved to 92.31 with the spectral clusters and to 92.68 with the Brown clusters using 300k vocabulary size. The spectral clusters are a little behind the Brown clusters in the test set results. However, we remark that the

well-known discrepancy between the dev set and the test set in the CoNLL 2003 dataset makes a conclusive interpretation difficult. For example, Turian *et al.* [2010] report that the F1 score using the embeddings of Collobert and Weston [2008] is higher than the F1 score using the Brown clusters on the dev set (92.46 vs 92.32) but lower on the test set (87.96 vs 88.52).

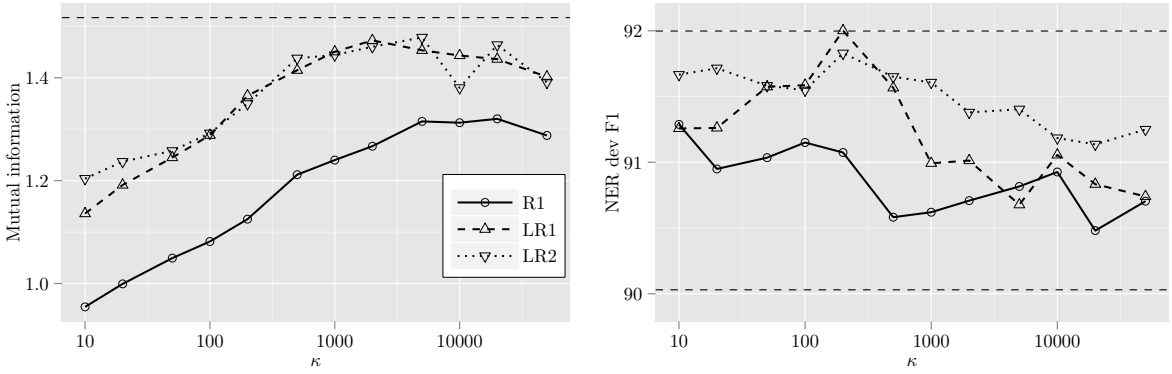
5.5.2.2 MI

Table 5.2 shows the MI computed as in Eq. (5.3) on the RCV1 corpus. The Brown algorithm optimizes the MI directly and generally achieves higher MI scores than the spectral algorithm. However, the spectral algorithm also achieves a surprisingly respectable level of MI scores even though the MI is not its objective. That is, the Brown algorithm specifically merges clusters in order to maximize the MI score in Eq. (5.3). In contrast, the spectral algorithm first recovers the model parameters using SVD and perform hierarchical clustering according to the parameter estimates, without any explicit concern for the MI score.

5.5.3 Comparison to the Brown Algorithm: Speed

To see the runtime difference between our algorithm and the Brown algorithm, we measured how long it takes to extract clusters from the RCV1 corpus for various numbers of clusters. In all the reported runtimes, we exclude the time to read and write data. We report results with 200, 400, 600, 800, and 1,000 clusters. All timing experiments were done on a machine with dual-socket, 8-core, 2.6GHz Intel Xeon E5-2670 (Sandy Bridge). The implementations for both algorithms were written in C++. The spectral algorithm also made use of Matlab for matrix calculations such as the SVD calculation.

Table 5.3 shows the runtimes required to extract these clusters as well as the F1 scores on the NER dev set obtained with these clusters. The spectral algorithm is considerably faster than the Brown algorithm while providing comparable improvement in the F1 scores. The runtime difference becomes more prominent as the number of clusters increases. Moreover, the spectral algorithm scales much better with larger vocabulary size. With 1,000 clusters and 300k vocabulary size, the Brown algorithm took over 22 hours whereas the spectral algorithm took 2 hours, 4 minutes, and 15 seconds—less than 10% of the time the Brown



(a) (b)

Figure 5.5: Effect of the choice of κ and context on (a) MI and (b) NER dev F1 score. We used 1,000 clusters on RCV1 with vocabulary size 50k. In (a), the horizontal line is the MI achieved by Brown clusters. In (b), the top horizontal line is the F1 score achieved with Brown clusters and the bottom horizontal line is the baseline F1 score achieved without using clusters.

algorithm takes.

We also note that for the Brown algorithm, the improvement varies significantly depending on how many clusters are used; it is 0.76 with 200 clusters but 1.97 with 1,000 clusters. For the spectral algorithm, this seems to be less the case; the improvement is 1.5 with 200 clusters and 1.97 with 1,000 clusters.

5.5.3.1 Discussion on Runtimes

The final asymptotic runtime is $O(N + \gamma nm^2)$ for the spectral algorithm and $O(N + nm^2)$ for the Brown algorithm, where N is the size of the corpus, n is the number of distinct word types, m is the number of clusters, and γ is a data-dependent constant. Thus it may be puzzling why the spectral algorithm is significantly faster in practice. We explicitly discuss the issue in this section.

The spectral algorithm proceeds in two stages. First, it constructs a scaled covariance matrix in $O(N)$ time and performs a rank-m SVD of this matrix. Table 5.3 shows that SVD scales well with the value of m and the size of the corpus.

Second, the algorithm performs hierarchical clustering in $O(\gamma nm^2)$ time. This stage

$$
\begin{aligned}
&\texttt{computeL2usingOld}(s,t,u,v,w) = \texttt{L2}[v][w] \\
&- \texttt{q2}[v][s] - \texttt{q2}[s][v] - \texttt{q2}[w][s] - \texttt{q2}[s][w] \\
&- \texttt{q2}[v][t] - \texttt{q2}[t][v] - \texttt{q2}[w][t] - \texttt{q2}[t][w] \\
&+ (\texttt{p2}[v][s] + \texttt{p2}[w][s]) * \log((\texttt{p2}[v][s] + \texttt{p2}[w][s])/((\texttt{p1}[v] + \texttt{p1}[w]) * \texttt{p1}[s])) \\
&+ (\texttt{p2}[s][v] + \texttt{p2}[s][w]) * \log((\texttt{p2}[s][v] + \texttt{p2}[s][w])/((\texttt{p1}[v] + \texttt{p1}[w]) * \texttt{p1}[s])) \\
&+ (\texttt{p2}[v][t] + \texttt{p2}[w][t]) * \log((\texttt{p2}[v][t] + \texttt{p2}[w][t])/((\texttt{p1}[v] + \texttt{p1}[w]) * \texttt{p1}[t])) \\
&+ (\texttt{p2}[t][v] + \texttt{p2}[t][w]) * \log((\texttt{p2}[t][v] + \texttt{p2}[t][w])/((\texttt{p1}[v] + \texttt{p1}[w]) * \texttt{p1}[t])) \\
&+ \texttt{q2}[v][u] + \texttt{q2}[u][v] + \texttt{q2}[w][u] + \texttt{q2}[u][w] \\
&- (\texttt{p2}[v][u] + \texttt{p2}[w][u]) * \log((\texttt{p2}[v][u] + \texttt{p2}[w][u])/((\texttt{p1}[v] + \texttt{p1}[w]) * \texttt{p1}[u])) \\
&- (\texttt{p2}[u][v] + \texttt{p2}[u][w]) * \log((\texttt{p2}[u][v] + \texttt{p2}[u][w])/((\texttt{p1}[v] + \texttt{p1}[w]) * \texttt{p1}[u]))
\end{aligned}
$$

Figure 5.6: A $O(1)$ function that is called $O(nm^2)$ times in Liang's implementation of the Brown algorithm, accounting for over 40% of the runtime. Similar functions account for the vast majority of the runtime. The values in the arrays $\texttt{L2}, \texttt{q2}, \texttt{p2}, \texttt{p1}$ are precomputed. $\texttt{p2}[v][w] = p(v,w)$, i.e, the probability of cluster v being followed by cluster w; $\texttt{p1}[v] = p(v)$ is the probability of cluster v; $\texttt{q2}[v][w] = p(v,w)\log((p(v)p(w))^{-1}p(v,w))$ is the contribution of the mutual information between clusters v and w. The function recomputes $\texttt{L2}[v][w]$, which is the loss in log-likelihood if clusters v and w are merged. The function updates $\texttt{L2}$ after clusters s and t have been merged to form a new cluster u. There are many operations involved in this calculation: 6 divisions, 12 multiplications, 36 additions (26 additions and 10 subtractions), and 6 log operations.

consists of $O(\gamma nm)$ calls to an $O(m)$ time function that computes Eq. (5.2), that is,

$$d(c,c') = \frac{|c||c'|}{|c|+|c'|}\,||\mu_c - \mu_{c'}||^2$$

This function is quite simple: it calculates a scaled distance between two vectors in $\mathbb{R}^m$.

Moreover, it avails itself readily to existing optimization techniques such as vectorization.[1] Finally, we found that the empirical value of γ was typically small: it ranged from 3.35 to 13.77 in our experiments reported in Table 5.3 (higher m required higher γ).

In contrast, while the main body of the Brown algorithm requires $O(N+nm^2)$ time, the constant factors are high due to fairly complex book-keeping that is required. For example, the function in Figure 5.6 (obtained from Liang's implementation) is invoked $O(nm^2)$ times in total: specifically, whenever two clusters s and t are merged to form a new cluster u (this happens $O(n)$ times), the function is called $O(m^2)$ times, for all pairs of clusters v, w such that v and w are not equal to s, t, or u. The function recomputes the loss in likelihood if clusters v and w are merged, after s and t are merged to form u. It requires a relatively large number of arithmetic operations, leading to high constant factors. Calls to this function alone take over 40% of the runtime for the Brown algorithm; similar functions account for the vast majority of the algorithm's runtime. It is not clear that this overhead can be reduced.

5.5.4 Effect of the Choice of κ and Context

Figure 5.5 shows the MI and the F1 score on the NER dev set for various choices of κ and context. For NER, around 100-200 for the value of κ gives good performance. For the MI, the value of κ needs to be much larger.

LR1 and LR2 perform much better than R1 but are very similar to each other across the results, suggesting that words in the immediate vicinity are necessary and nearly sufficient for these tasks.

5.6 Conclusion

In this work, we have presented a new and faster alternative to the Brown clustering algorithm. Our algorithm has a provable guarantee of recovering the underlying model parameters. This approach first uses SVD to consistently estimate low-dimensional representations

[1] Many linear algebra libraries automatically support vectorization. For instance, the `Eigen` library in our implementation enables vectorization by default, which gave a 2-3 time speedup in our experiments.

of word types that reveal their originating clusters by exploiting the implicit disjoint-cluster assumption of the Brown model. Then agglomerative clustering is performed over these representations to build a hierarchy of word types. The resulting clusters give a competitive level of improvement in performance in NER as the clusters from the Brown algorithm, but the spectral algorithm is significantly faster.

There are several areas for the future work. One can try to speed up the algorithm even more via a top-down rather than bottom-up approach for hierarchical clustering, for example recursively running the 2-means algorithm. Experiments with the clusters in tasks other than NER (e.g., dependency parsing), as well as larger-scale experiments, can help further verify the quality of the clusters and highlight the difference between the spectral algorithm and the Brown algorithm.

Input: vectors $\mu^{(1)}, \ldots, \mu^{(n)} \in \mathbb{R}^m$ corresponding to word types $[n]$ ordered in decreasing frequency.

Output: hierarchical clustering of the input vectors.

Tightening: Given a set of clusters $\mathcal{C}$, the subroutine tighten(c) for $c \in \mathcal{C}$ consists of the following three steps:

$$\text{nearest}(c) := \arg\min_{c' \in \mathcal{C}: c' \neq c} d(c, c')$$

$$\text{lowerbound}(c) := \min_{c' \in \mathcal{C}: c' \neq c} d(c, c')$$

$$\text{tight}(c) := \texttt{True}$$

Main body:

1. Initialize active clusters $\mathcal{C} = \{\{\mu^{(1)}\}, \ldots, \{\mu^{(m)}\}\}$ and call tighten(c) for all $c \in \mathcal{C}$.
2. For $i = m + 1$ to $n + m - 1$:
   (a) If $i \leq n$: let $c := \{\mu^{(i)}\}$, call tighten(c), and let $\mathcal{C} := \mathcal{C} \cup \{c\}$.
   (b) Let $c^* := \arg\min_{c \in \mathcal{C}} \text{lowerbound}(c)$.
   (c) While tight(c^*) is `False`,
      i. Call tighten(c^*).
      ii. Let $c^* := \arg\min_{c \in \mathcal{C}} \text{lowerbound}(c)$.
   (d) Merge c^* and nearest(c^*) in $\mathcal{C}$.
   (e) For each $c \in \mathcal{C}$: if nearest(c) $\in \{c^*, \text{nearest}(c^*)\}$, set tight($c$) := `False`.

Figure 5.7: Variant of Ward's algorithm from Section 5.4.4.

m	vocab	Spectral runtime				Brown runtime	Ratio (%)	Spectral F1	Brown F1
		γ	SVD	cluster	total				
200	50k	3.35	4m24s	13s	4m37s	10m37s	43.48	91.53	90.79
400		5.17	6m39s	1m8s	7m47s	37m16s	20.89	91.73	91.21
600		9.80	5m29s	3m1s	8m30s	1h33m55s	9.05	91.68	91.79
800		12.64	9m26s	6m59s	16m25s	2h20m40s	11.67	91.81	91.83
1000		12.68	11m10s	10m25s	21m35s	3h37m	9.95	92.00	92.00
1000	300k	13.77	59m38s	1h4m37s	2h4m15s	22h19m37s	9.28	92.31	92.68

Table 5.3: Speed and performance comparison with the Brown algorithm for different numbers of clusters and vocabulary sizes. In all the reported runtimes, we exclude the time to read and write data. We report the F1 scores on the NER dev set; for the spectral algorithm, we report the best scores.

Chapter 6

Word Embeddings from Decompositions of Count Matrices

This chapter is adapted from joint work with Michael Collins and Daniel Hsu entitled "Model-Based Word Embeddings from Decompositions of Count Matrices" [Stratos *et al.*, 2015].

This work develops a new statistical understanding of word embeddings induced from transformed count data. Using the class of hidden Markov models (HMMs) underlying Brown clustering as a generative model, we demonstrate how canonical correlation analysis (CCA) and certain count transformations permit efficient and effective recovery of model parameters with lexical semantics. We further show in experiments that these techniques empirically outperform existing spectral methods on word similarity and analogy tasks, and are also competitive with other popular methods such as WORD2VEC and GLOVE.[1]

In this chapter, we use $M^{\langle a \rangle}$ to denote the element-wise power of matrix M by a scalar a, that is, $[\sqrt{M}^{\langle a \rangle}]_{i,j} = M_{i,j}^a$. This is distinguished from the usual matrix power M^a (defined only for a square M).

[1]We consistently denote the method of Mikolov *et al.* [2013b] by "WORD2VEC" and the method of Pennington *et al.* [2014] by "GLOVE" in this chapter.

6.1 Introduction

The recent spike of interest in dense, low-dimensional lexical representations (i.e., word embeddings) is largely due to their ability to capture subtle syntactic and semantic patterns that are useful in a variety of natural language tasks. A successful method for deriving such embeddings is the negative sampling training of the skip-gram model suggested by Mikolov *et al.* [2013b] and implemented in the popular software WORD2VEC. The form of its training objective was motivated by efficiency considerations, but has subsequently been interpreted by Levy and Goldberg [2014b] as seeking a *low-rank factorization* of a matrix whose entries are word-context co-occurrence counts, scaled and transformed in a certain way. This observation sheds new light on WORD2VEC, yet also raises several new questions about word embeddings based on decomposing count data. What is the right matrix to decompose? Are there rigorous justifications for the choice of matrix and count transformations?

In this paper, we answer some of these questions by investigating the decomposition specified by CCA Hotelling [1936], a powerful technique for inducing generic representations whose computation is efficiently and exactly reduced to that of a matrix singular value decomposition (SVD). We build on the work of Stratos *et al.* [2014] which uses CCA for learning the class of HMMs underlying Brown clustering. We show that certain count transformations enhance the accuracy of the estimation method and significantly improve the empirical performance of word representations derived from these model parameters (Table 6.1).

In addition to providing a rigorous justification for CCA-based word embeddings, we also supply a general template that encompasses a range of spectral methods (algorithms employing SVD) for inducing word embeddings in the literature, including the method of Levy and Goldberg [2014b]. In experiments, we demonstrate that CCA combined with the square-root transformation achieves the best result among spectral methods and performs competitively with other popular methods such as WORD2VEC and GLOVE on word similarity and analogy tasks. We additionally demonstrate that CCA embeddings provide the most competitive improvement when used as features in named-entity recognition (NER).

6.2 Background in CCA

In this section, we review the variational characterization of CCA. This provides a flexible framework for a wide variety of tasks. CCA seeks to maximize a statistical quantity known as the Pearson correlation coefficient between random variables $L, R \in \mathbb{R}$:

$$\mathrm{Cor}(L, R) := \frac{\boldsymbol{E}[LR] - \boldsymbol{E}[L]\,\boldsymbol{E}[R]}{\sqrt{\boldsymbol{E}[L^2] - \boldsymbol{E}[L]^2}\sqrt{\boldsymbol{E}[R^2] - \boldsymbol{E}[R]^2}}$$

This is a value in $[-1, 1]$ indicating the degree of linear dependence between L and R.

6.2.1 CCA Objective

Let $X \in \mathbb{R}^n$ and $Y \in \mathbb{R}^{n'}$ be two random vectors. Without loss of generality, we will assume that X and Y have zero mean.[2] Let $m \leq \min(n, n')$. CCA can be cast as finding a set of projection vectors (called canonical directions) $a_1 \ldots a_m \in \mathbb{R}^n$ and $b_1 \ldots b_m \in \mathbb{R}^{n'}$ such that for $i = 1 \ldots m$:

$$\begin{aligned}(a_i, b_i) = \underset{a \in \mathbb{R}^n,\, b \in \mathbb{R}^{n'}}{\arg\max} \quad & \mathrm{Cor}(a^\top X, b^\top Y) \\ & \mathrm{Cor}(a^\top X, a_j^\top X) = 0 \qquad \forall j < i \\ & \mathrm{Cor}(b^\top Y, b_j^\top Y) = 0 \qquad \forall j < i\end{aligned} \tag{6.1}$$

That is, at each i we simultaneously optimize vectors a, b so that the projected random variables $a^\top X, b^\top Y \in \mathbb{R}$ are maximally correlated, subject to the constraint that the projections are uncorrelated to all previous projections.

Let $A := [a_1 \ldots a_m]$ and $B := [b_1 \ldots b_m]$. Then we can think of the joint projections

$$\underline{X} = A^\top X \qquad \underline{Y} = B^\top Y \tag{6.2}$$

as new m-dimensional representations of the original variables that are transformed to be as correlated as possible with each other. Furthermore, often $m \ll \min(n, n')$, leading to a dramatic reduction in dimensionality.

[2] This can be always achieved through data preprocessing ("centering").

6.2.2 Exact Solution via SVD

Eq. (6.1) is non-convex due to the terms a and b that interact with each other, so it cannot be solved exactly using a standard optimization technique. However, a method based on SVD provides an efficient and exact solution. For a proof and more detailed discussions, see Hardoon *et al.* [2004].

Lemma 6.2.1 (Hotelling [1936])**.** *Assume X and Y have zero mean. The solution (A, B) to (6.1) is given by $A = \boldsymbol{E}[XX^\top]^{-1/2}U$ and $B = \boldsymbol{E}[YY^\top]^{-1/2}V$ where the i-th column of $U \in \mathbb{R}^{n\times m}$ ($V \in \mathbb{R}^{n'\times m}$) is the left (right) singular vector of*

$$\Omega := \boldsymbol{E}[XX^\top]^{-1/2}\,\boldsymbol{E}[XY^\top]\,\boldsymbol{E}[YY^\top]^{-1/2} \tag{6.3}$$

corresponding to the i-th largest singular value σ_i. Furthermore, $\sigma_i = Cor(a_i^\top X, b_i^\top Y)$.

6.2.3 Using CCA for Word Representations

As presented in Section 6.2.1, CCA is a general framework that operates on a pair of random variables. Adapting CCA specifically to inducing word representations results in a simple recipe for calculating (6.3).

A natural approach is to set X to represent a word and Y to represent the relevant "context" information about a word. We can use CCA to project X and Y to a low-dimensional space in which they are maximally correlated: see Eq. (6.2). The projected X can be considered as a new word representation.

Denote the set of distinct word types by $[n]$. We set $X, Y \in \mathbb{R}^n$ to be one-hot encodings of words and their associated context words. We define a context word to be a word occurring within ρ positions to the left and right (excluding the current word). For example, with $\rho = 1$, the following snippet of text where the current word is "souls":

```
Whatever our souls are made of
```

will generate two samples of $X \times Y$: a pair of indicator vectors for "souls" and "our", and a pair of indicator vectors for "souls" and "are".

CCA requires performing SVD on the following matrix $\Omega \in \mathbb{R}^{n \times n}$:

$$\begin{aligned}\Omega =&(\boldsymbol{E}[XX^\top] - \boldsymbol{E}[X]\,\boldsymbol{E}[X]^\top)^{-1/2} \\ &(\boldsymbol{E}[XY^\top] - \boldsymbol{E}[X]\,\boldsymbol{E}[Y]^\top) \\ &(\boldsymbol{E}[YY^\top] - \boldsymbol{E}[Y]\,\boldsymbol{E}[Y]^\top)^{-1/2}\end{aligned}$$

At a quick glance, this expression looks daunting: we need to perform matrix inversion and multiplication on potentially large dense matrices. However, Ω is easily computable with the following observations:

Observation 1. We can ignore the centering operation when the sample size is large [Dhillon *et al.*, 2011a]. To see why, let $\{(x^{(i)}, y^{(i)})\}_{i=1}^N$ be N samples of X and Y. Consider the sample estimate of the term $\boldsymbol{E}[XY^\top] - \boldsymbol{E}[X]\,\boldsymbol{E}[Y]^\top$:

$$\frac{1}{N}\sum_{i=1}^{N} x^{(i)}(y^{(i)})^\top - \frac{1}{N^2}\left(\sum_{i=1}^{N} x^{(i)}\right)\left(\sum_{i=1}^{N} y^{(i)}\right)^\top$$

The first term dominates the expression when N is large. This is indeed the setting in this task where the number of samples (word-context pairs in a corpus) easily tends to billions.

Observation 2. The (uncentered) covariance matrices $\boldsymbol{E}[XX^\top]$ and $\boldsymbol{E}[YY^\top]$ are diagonal. This follows from our definition of the word and context variables as one-hot encodings since $\boldsymbol{E}[X_w X_{w'}] = 0$ for $w \neq w'$ and $\boldsymbol{E}[Y_c Y_{c'}] = 0$ for $c \neq c'$.

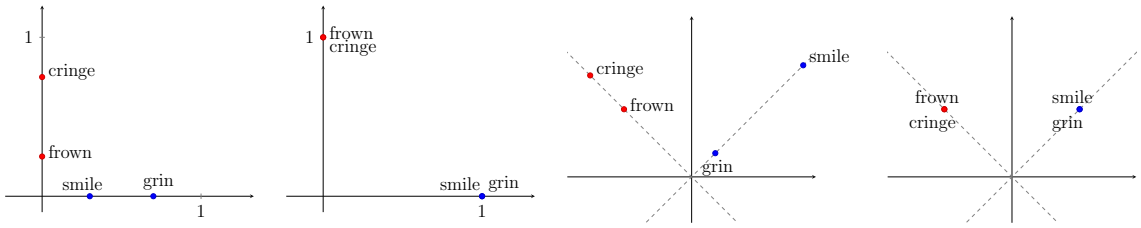
With these observations and the binary definition of (X, Y), each entry in Ω now has a simple closed-form solution:

$$\Omega_{w,c} = \frac{P(X_w = 1, Y_c = 1)}{\sqrt{P(X_w = 1)P(Y_c = 1)}} \tag{6.4}$$

which can be readily estimated from a corpus.

6.3 Using CCA for Parameter Estimation

In a less well-known interpretation of Eq. (6.4), CCA is seen as a parameter estimation algorithm for a language model [Stratos *et al.*, 2014]. This model is a restricted class



(a) (b)

Figure 6.1: Visualization of the representational scheme under a Brown model with 2 hidden states. (a) Normalizing the original rows of O. (b) Normalizing the scaled and rotated rows of O.

of HMMs introduced by Brown *et al.* [1992], henceforth called the Brown model. In this section, we extend the result of Stratos *et al.* [2014] and show that its correctness is preserved under certain element-wise data transformations.

6.3.1 Clustering under a Brown Model

A Brown model is a 5-tuple (n, m, π, t, o) for $n, m \in \mathbb{N}$ and functions π, t, o where

- $[n]$ is a set of word types.
- $[m]$ is a set of hidden states.
- $\pi(h)$ is the probability of generating $h \in [m]$ in the first position of a sequence.
- $t(h'|h)$ is the probability of generating $h' \in [m]$ given $h \in [m]$.
- $o(w|h)$ is the probability of generating $w \in [n]$ given $h \in [m]$.

Importantly, the model makes the following additional assumption:

Assumption 6.3.1 (Brown assumption)**.** *For each word type $w \in [n]$, there is a unique hidden state $\mathcal{H}(w) \in [m]$ such that $o(w|\mathcal{H}(w)) > 0$ and $o(w|h) = 0$ for all $h \neq \mathcal{H}(w)$.*

In other words, this model is an HMM in which observation states are partitioned by hidden states. Thus a sequence of N words $w_1 \dots w_N \in [n]^N$ has probability $\pi(\mathcal{H}(w_1)) \times \prod_{i=1}^{N} o(w_i|\mathcal{H}(w_i)) \times \prod_{i=1}^{N-1} t(\mathcal{H}(w_{i+1})|\mathcal{H}(w_i))$.

An equivalent definition of a Brown model is given by organizing the parameters in matrix form. Under this definition, a Brown model has parameters (π, T, O) where $\pi \in \mathbb{R}^m$ is a vector and $T \in \mathbb{R}^{m \times m}, O \in \mathbb{R}^{n \times m}$ are matrices whose entries are set to:

$$\begin{aligned} \pi_h &= \pi(h) && h \in [m] \\ T_{h',h} &= t(h'|h) && h, h' \in [m] \\ O_{w,h} &= o(w|h) && h \in [m], w \in [n] \end{aligned}$$

Our main interest is in obtaining some representations of word types that allow us to identify their associated hidden states under the model. For this purpose, representing a word by the corresponding row of O is sufficient. To see this, note that each row of O must have a single nonzero entry by Assumption 6.3.1. Let $v(w) \in \mathbb{R}^m$ be the w-th row of O normalized to have unit 2-norm: then $v(w) = v(w')$ iff $\mathcal{H}(w) = \mathcal{H}(w')$. See Figure 6.1(a) for illustration.

A crucial aspect of this representational scheme is that its correctness is *invariant* to scaling and rotation. In particular, clustering the normalized rows of $\mathrm{diag}(s)O^{\langle a \rangle} \mathrm{diag}(s_2)Q^\top$ where $O^{\langle a \rangle}$ is any element-wise power of O with any $a \neq 0$, $Q \in \mathbb{R}^{m \times m}$ is any orthogonal transformation, and $s_1 \in \mathbb{R}^n$ and $s_2 \in \mathbb{R}^m$ are any positive vectors yields the correct clusters under the model. See Figure 6.1(b) for illustration.

6.3.2 Spectral Estimation

Thus we would like to estimate O and use its rows for representing word types. But the likelihood function under the Brown model is non-convex, making an MLE estimation of the model parameters difficult. However, the hard-clustering assumption (Assumption 6.3.1) allows for a simple spectral method for consistent parameter estimation of O.

To state the theorem, we define an additional quantity. Let ρ be the number of left/right context words to consider in CCA. Let $(H_1, \ldots, H_N) \in [m]^N$ be a random sequence of hidden states drawn from the Brown model where $N \geq 2\rho + 1$. Independently, pick a position $I \in [\rho + 1, N - \rho]$ uniformly at random. Define $\tilde{\pi} \in \mathbb{R}^m$ where $\tilde{\pi}_h := P(H_I = h)$ for each $h \in [m]$.

Theorem 6.3.1. *Assume* $\tilde{\pi} > 0$ *and* $\operatorname{rank}(O) = \operatorname{rank}(T) = m$. *Assume that a Brown model* (π, T, O) *generates a sequence of words. Let* $X, Y \in \mathbb{R}^n$ *be one-hot encodings of words and their associated context words. Let* $U \in \mathbb{R}^{n \times m}$ *be the matrix of* m *left singular vectors of* $\Omega^{\langle a \rangle} \in \mathbb{R}^{n \times n}$ *corresponding to nonzero singular values where* Ω *is defined in Eq. (6.4) and* $a \neq 0$*:*

$$\Omega^{\langle a \rangle}_{w,c} = \frac{P(X_w = 1, Y_c = 1)^a}{\sqrt{P(X_w = 1)^a P(Y_c = 1)^a}}$$

Then there exists an orthogonal matrix $Q \in \mathbb{R}^{m \times m}$ *and a positive* $s \in \mathbb{R}^m$ *such that* $U = O^{\langle a/2 \rangle} \operatorname{diag}(s) Q^\top$.

This theorem states that the CCA projection of words in Section 6.2.3 is the rows of O up to scaling and rotation even if we raise each element of Ω in Eq. (6.4) to an arbitrary (nonzero) power. The proof is a variant of the proof in Stratos *et al.* [2014] and is given in Appendix B.

6.3.3 Choice of Data Transformation

Given a corpus, the sample estimate of $\Omega^{\langle a \rangle}$ is given by:

$$\hat{\Omega}^{\langle a \rangle}_{w,c} = \frac{\#(w,c)^a}{\sqrt{\#(w)^a \#(c)^a}} \tag{6.5}$$

where $\#(w,c)$ denotes the co-occurrence count of word w and context c in the corpus, $\#(w) := \sum_c \#(w,c)$, and $\#(c) := \sum_w \#(w,c)$. What choice of a is beneficial and why? We use $a = 1/2$ for the following reason: it stabilizes the variance of the term and thereby gives a more statistically stable solution.

6.3.3.1 Variance Stabilization for Word Counts

The square-root transformation is a *variance-stabilizing transformation* for Poisson random variables [Bartlett, 1936; Anscombe, 1948]. In particular, the square-root of a Poisson variable has variance close to $1/4$, independent of its mean.

Lemma 6.3.2 (Bartlett [1936])**.** *Let* X *be a random variable with distribution Poisson*$(n \times p)$ *for any* $p \in (0,1)$ *and positive integer* n. *Define* $Y := \sqrt{X}$. *Then the variance of* Y *approaches* $1/4$ *as* $n \to \infty$.

This transformation is relevant for word counts because they can be naturally modeled as Poisson variables. Indeed, if word counts in a corpus of length N are drawn from a multinomial distribution over $[n]$ with N observations, then these counts have the same distribution as n independent Poisson variables (whose rate parameters are related to the multinomial probabilities), conditioned on their sum equaling N [Steel, 1953]. Empirically, the peaky concentration of a Poisson distribution is well-suited for modeling word occurrences.

6.3.3.2 Variance-Weighted Squared-Error Minimization

At the heart of CCA is computing the SVD of the $\Omega^{\langle a \rangle}$ matrix: this can be interpreted as solving the following (non-convex) squared-error minimization problem:

$$\min_{u_w, v_c \in \mathbb{R}^m} \sum_{w,c} \left(\Omega^{\langle a \rangle}_{w,c} - u_w^\top v_c \right)^2$$

But we note that minimizing *unweighted* squared-error objectives is generally suboptimal when the target values are heteroscedastic. For instance, in linear regression, it is well-known that a *weighted least squares* estimator dominates ordinary least squares in terms of statistical efficiency [Aitken, 1936; Lehmann and Casella, 1998]. For our setting, the analogous weighted least squares optimization is:

$$\min_{u_w, v_c \in \mathbb{R}^m} \sum_{w,c} \frac{1}{\mathrm{Var}\left(\Omega^{\langle a \rangle}_{w,c} \right)} \left(\Omega^{\langle a \rangle}_{w,c} - u_w^\top v_c \right)^2 \tag{6.6}$$

where $\mathrm{Var}(X) := \boldsymbol{E}[X^2] - \boldsymbol{E}[X]^2$. This optimization is, unfortunately, generally intractable [Srebro *et al.*, 2003]. The square-root transformation, nevertheless, obviates the variance-based weighting since the target values have approximately the same variance of 1/4.

6.4 A Template for Spectral Methods

Figure 6.2 gives a generic template that encompasses a range of spectral methods for deriving word embeddings. All of them operate on co-occurrence counts $\#(w, c)$ and share the low-rank SVD step, but they can differ in the data transformation method (t) and the definition of the matrix of scaled counts for SVD (s).

We introduce two additional parameters $\alpha, \beta \leq 1$ to account for the following details. Mikolov *et al.* [2013b] proposed smoothing the empirical context distribution as $\hat{p}_\alpha(c) := \#(c)^\alpha / \sum_c \#(c)^\alpha$ and found $\alpha = 0.75$ to work well in practice. We also found that setting $\alpha = 0.75$ gave a small but consistent improvement over setting $\alpha = 1$. Note that the choice of α only affects methods that make use of the context distribution ($s \in \{\text{ppmi}, \text{cca}\}$).

The parameter β controls the role of singular values in word embeddings. This is always 0 for CCA as it does not require singular values. But for other methods, one can consider setting $\beta > 0$ since the best-fit subspace for the rows of Ω is given by $U\Sigma$. For example, Deerwester *et al.* [1990] use $\beta = 1$ and Levy and Goldberg [2014b] use $\beta = 0.5$. However, it has been found by many (including ourselves) that setting $\beta = 1$ yields substantially worse representations than setting $\beta \in \{0, 0.5\}$ [Levy *et al.*, 2015].

Different combinations of these aspects reproduce various spectral embeddings explored in the literature. We enumerate some meaningful combinations:

No scaling $\left[t \in \{—, \log, \text{sqrt}\},\ s = —\right]$. This is a commonly considered setting (e.g., in Pennington *et al.* [2014]) where no scaling is applied to the co-occurrence counts. It is however typically accompanied with some kind of data transformation.

Positive point-wise mutual information (PPMI) $\left[t = —,\ s = \text{ppmi}\right]$. Mutual information is a popular metric in many natural language tasks [Brown *et al.*, 1992; Pantel and Lin, 2002]. In this setting, each term in the matrix for SVD is set as the point-wise mutual information between word w and context c:

$$\log \frac{\hat{p}(w,c)}{\hat{p}(w)\hat{p}_\alpha(c)} = \log \frac{\#(w,c)\sum_c \#(c)^\alpha}{\#(w)\#(c)^\alpha}$$

Typically negative values are thresholded to 0 to keep Ω sparse. Levy and Goldberg [2014b] observed that the negative sampling objective of the skip-gram model of Mikolov *et al.* [2013b] is implicitly factorizing a shifted version of this matrix.[3]

Regression $\left[t \in \{—, \text{sqrt}\},\ s = \text{reg}\right]$. Another novelty of our work is considering a low-rank approximation of a linear regressor that predicts the context from words. Denoting

[3]This is not equivalent to applying SVD on this matrix, however, since the loss function is different.

SPECTRAL-TEMPLATE

Input: word-context co-occurrence counts $\#(w, c)$, dimension m, transformation method t, scaling method s, context smoothing exponent $\alpha \leq 1$, singular value exponent $\beta \leq 1$

Output: vector $v(w) \in \mathbb{R}^m$ for each word $w \in [n]$

Definitions: $\#(w) := \sum_c \#(w, c)$, $\#(c) := \sum_w \#(w, c)$, $N(\alpha) := \sum_c \#(c)^\alpha$

1. Transform all $\#(w, c)$, $\#(w)$, and $\#(c)$:

$$\#(\cdot) \leftarrow \begin{cases} \#(\cdot) & \text{if } t = \text{—} \\ \log(1 + \#(\cdot)) & \text{if } t = \log \\ \#(\cdot)^{2/3} & \text{if } t = \text{two-thirds} \\ \sqrt{\#(\cdot)} & \text{if } t = \text{sqrt} \end{cases}$$

2. Scale statistics to construct a matrix $\Omega \in \mathbb{R}^{n \times n}$:

$$\Omega_{w,c} \leftarrow \begin{cases} \#(w, c) & \text{if } s = \text{—} \\ \frac{\#(w,c)}{\#(w)} & \text{if } s = \text{reg} \\ \max\left(\log \frac{\#(w,c)N(\alpha)}{\#(w)\#(c)^\alpha}, 0\right) & \text{if } s = \text{ppmi} \\ \frac{\#(w,c)}{\sqrt{\#(w)\#(c)^\alpha}} \sqrt{\frac{N(\alpha)}{N(1)}} & \text{if } s = \text{cca} \end{cases}$$

3. Perform rank-m SVD on $\Omega \approx U\Sigma V^\top$ where $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_m)$ is a diagonal matrix of ordered singular values $\sigma_1 \geq \cdots \geq \sigma_m \geq 0$.

4. Define $v(w) \in \mathbb{R}^m$ to be the w-th row of $U\Sigma^\beta$ normalized to have unit 2-norm.

Figure 6.2: A template for spectral word embedding methods.

the word sample matrix by $\mathcal{X} \in \mathbb{R}^{N \times n}$ and the context sample matrix by $\mathcal{Y} \in \mathbb{R}^{N \times n}$, we seek $U^* = \arg\min_{U \in \mathbb{R}^{n \times n}} ||\mathcal{Y} - \mathcal{X}U||^2$ whose closed-from solution is given by:

$$U^* = (\mathcal{X}^\top \mathcal{X})^{-1} \mathcal{X}^\top \mathcal{Y} \tag{6.7}$$

Thus we aim to compute a low-rank approximation of U^* with SVD. This is inspired by other predictive models in the representation learning literature [Ando and Zhang, 2005; Mikolov *et al.*, 2013a]. We consider applying the square-root transformation for the same variance stabilizing effect discussed in Section 6.3.3.

CCA $\big[t \in \{\text{—}, \text{two-thirds}, \text{sqrt}\},\ s = \text{cca}\big]$. This is the focus of our work. As shown

in Theorem 6.3.1, we can take the element-wise power transformation on counts (such as the power of $1, 2/3, 1/2$ in this template) while preserving the representational meaning of word embeddings under the Brown model interpretation. If there is no data transformation ($t = —$), then we recover the original spectral algorithm of Stratos *et al.* [2014].

6.5 Related Work

We make a few remarks on related works not already discussed earlier. Dhillon *et al.* [2011a, 2012] propose novel modifications of CCA (LR-MVL and two-step CCA) to derive word embeddings, but do not establish any explicit connection to learning HMM parameters or justify the square-root transformation. Pennington *et al.* [2014] propose a weighted factorization of log-transformed co-occurrence counts, which is generally an intractable problem [Srebro *et al.*, 2003]. In contrast, our method requires only efficiently computable matrix decompositions. Finally, word embeddings have also been used as features to improve performance in a variety of supervised tasks such as sequence labeling [Dhillon *et al.*, 2011a; Collobert *et al.*, 2011] and dependency parsing [Lei *et al.*, 2014; Chen and Manning, 2014]. Here, we focus on understanding word embeddings in the context of a generative word class model, as well as in empirical tasks that directly evaluate the word embeddings themselves.

6.6 Experiments

6.6.1 Word Similarity and Analogy

We first consider word similarity and analogy tasks for evaluating the quality of word embeddings. Word similarity measures the Spearman's correlation coefficient between the human scores and the embeddings' cosine similarities for word pairs. Word analogy measures the accuracy on syntactic and semantic analogy questions. We refer to Levy and Goldberg [2014a] for a detailed description of these tasks. We use the multiplicative technique of Levy and Goldberg [2014a] for answering analogy questions.

For the choice of corpus, we use a preprocessed English Wikipedia dump (`http://dumps.wikimedia.org/`). The corpus contains around 1.4 billion words. We only preserve

Transform (t)	AVG-SIM	SYN	MIXED
—	0.572	39.68	57.64
log	0.675	55.61	69.26
two-thirds	0.650	60.52	74.00
sqrt	**0.690**	**65.14**	**77.70**

Table 6.1: Performance of CCA (1000 dimensions) on the development portion of data with different data transformation methods ($\alpha = 0.75$, $\beta = 0$).

word types that appear more than 100 times and replace all others with a special symbol, resulting in a vocabulary of size around 188k. We define context words to be 5 words to the left/right for all considered methods.

We use three word similarity datasets each containing 353, 3000, and 2034 word pairs.[4] We report the average similarity score across these datasets under the label AVG-SIM. We use two word analogy datasets that we call SYN (8000 syntactic analogy questions) and MIXED (19544 syntactic and semantic analogy questions).[5]

We implemented the template in Figure 6.2 in C++.[6] We compared against the public implementation of WORD2VEC by Mikolov *et al.* [2013b] and GLOVE by Pennington *et al.* [2014]. These external implementations have numerous hyperparameters that are not part of the core algorithm, such as random subsampling in WORD2VEC and the word-context averaging in GLOVE. We refer to Levy *et al.* [2015] for a discussion of the effect of these features. In our experiments, we enable all these features with the recommended default settings.

We reserve a half of each dataset (by category) as a held-out portion for development and use the other half for final evaluation.

[4] WordSim-353: `http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/`; MEN: `http://clic.cimec.unitn.it/~elia.bruni/MEN.html`; Stanford Rare Word: `http://www-nlp.stanford.edu/~lmthang/morphoNLM/`.

[5] `http://research.microsoft.com/en-us/um/people/gzweig/Pubs/myz_naacl13_test_set.tgz`; `http://www.fit.vutbr.cz/~imikolov/rnnlm/word-test.v1.txt`

[6] The code is available at `https://github.com/karlstratos/singular`.

Configuration		500 dimensions			1000 dimensions		
Transform (t)	Scale (s)	AVG-SIM	SYN	MIXED	AVG-SIM	SYN	MIXED
—	—	0.514	31.58	28.39	0.522	29.84	32.15
sqrt	—	0.656	60.77	65.84	0.646	57.46	64.97
log	—	0.669	59.28	66.86	0.672	55.66	68.62
—	reg	0.530	29.61	36.90	0.562	32.78	37.65
sqrt	reg	0.625	63.97	67.30	0.638	**65.98**	70.04
—	ppmi	0.638	41.62	58.80	0.665	47.11	65.34
sqrt	cca	**0.678**	**66.40**	**74.73**	**0.690**	65.14	**77.70**

Table 6.2: Performance of various spectral methods on the development portion of data.

6.6.1.1 Effect of Data Transformation for CCA

We first look at the effect of different data transformations on the performance of CCA. Table 6.1 shows the result on the development portion with 1000-dimensional embeddings. We see that without any transformation, the performance can be quite bad—especially in word analogy. But there is a marked improvement upon transforming the data. Moreover, the square-root transformation gives the best result, improving the accuracy on the two analogy datasets by 25.46% and 20.06% in absolute magnitude. This aligns with the discussion in Section 6.3.3.

6.6.1.2 Comparison Among Different Spectral Embeddings

Next, we look at the performance of various combinations in the template in Figure 6.2. We smooth the context distribution with $\alpha = 0.75$ for PPMI and CCA. We use $\beta = 0.5$ for PPMI (which has a minor improvement over $\beta = 0$) and $\beta = 0$ for all other methods. We generally find that using $\beta = 0$ is critical to obtaining good performance for $s \in \{—, \text{reg}\}$.

Table 6.2 shows the result on the development portion for both 500 and 1000 dimensions. Even without any scaling, SVD performs reasonably well with the square-root and log transformations. The regression scaling performs very poorly without data transformation, but once the square-root transformation is applied it performs quite well (especially in

Method		500 dimensions			1000 dimensions		
		AVG-SIM	SYN	MIXED	AVG-SIM	SYN	MIXED
Spectral	LOG	0.652	59.52	67.27	0.635	56.53	68.67
	REG	0.602	65.51	67.88	0.609	66.47	70.48
	PPMI	0.628	43.81	58.38	0.637	48.99	63.82
	CCA	**0.655**	68.38	74.17	**0.650**	66.08	76.38
Others	GLOVE	0.576	68.30	78.08	0.586	67.40	78.73
	CBOW	0.597	75.79	73.60	0.509	70.97	60.12
	SKIP	0.642	**81.08**	**78.73**	0.641	**79.98**	**83.35**

Table 6.3: Performance of different word embedding methods on the test portion of data. See the main text for the configuration details of spectral methods.

analogy questions). The PPMI scaling achieves good performance in word similarity but not in word analogy. The CCA scaling, combined with the square-root transformation, gives the best overall performance. In particular, it performs better than all other methods in mixed analogy questions by a significant margin.

6.6.1.3 Comparison with Other Embedding Methods

We compare spectral embedding methods against WORD2VEC and GLOVE on the test portion. We use the following combinations based on their performance on the development portion:

- LOG: log transform, — scaling
- REG: sqrt transform, reg scaling
- PPMI: — transform, ppmi scaling
- CCA: sqrt transform, cca scaling

For WORD2VEC, there are two model options: continuous bag-of-words (CBOW) and skip-gram (SKIP). Table 6.3 shows the result for both 500 and 1000 dimensions.

In word similarity, spectral methods generally excel, with CCA consistently performing the best. SKIP is the only external package that performs comparably, with GLOVE and CBOW falling behind. In word analogy, REG and CCA are significantly better than other spectral methods. They are also competitive to GLOVE and CBOW, but SKIP does perform the best among all compared methods on (especially syntactic) analogy questions.

6.6.2 As Features in a Supervised Task

Finally, we use word embeddings as features in NER and compare the subsequent improvements between various embedding methods. The experimental setting is identical to that of Stratos *et al.* [2014]. We use the Reuters RCV1 corpus which contains 205 million words. With frequency thresholding, we end up with a vocabulary of size around 301k. We derive LOG, REG, PPMI, and CCA embeddings as described in Section 6.6.1.3, and GLOVE, CBOW, and SKIP embeddings again with the recommended default settings. The number of left/right contexts is 2 for all methods. For comparison, we also derived 1000 Brown clusters (BROWN) on the same vocabulary and used the resulting bit strings as features Brown *et al.* [1992].

Table 6.4 shows the result for both 30 and 50 dimensions. In general, using any of these lexical features provides substantial improvements over the baseline.[7] In particular, the 30-dimensional CCA embeddings improve the F1 score by 2.84 on the development portion and by 4.88 on the test portion. All spectral methods perform competitively with external packages, with CCA and SKIP consistently delivering the biggest improvements on the development portion.

6.7 Conclusion

In this work, we revisited SVD-based methods for inducing word embeddings. We examined a framework provided by CCA and showed that the resulting word embeddings can be viewed as cluster-revealing parameters of a certain model and that this result is robust to

[7]We mention that the well-known dev/test discrepancy in the CoNLL 2003 dataset makes the results on the test portion less reliable.

Features	30 dimensions		50 dimensions	
	Dev	Test	Dev	Test
—	90.04	84.40	90.04	84.40
BROWN	92.49	88.75	92.49	88.75
LOG	92.27	88.87	92.91	**89.67**
REG	92.51	88.08	92.73	88.88
PPMI	92.25	89.27	92.53	89.37
CCA	**92.88**	**89.28**	92.94	89.01
GLOVE	91.49	87.16	91.58	86.80
CBOW	92.44	88.34	92.83	89.21
SKIP	92.63	88.78	**93.11**	89.32

Table 6.4: NER F1 scores when word embeddings are added as real-valued features to the baseline (—). For comparison, we also derive 1000 Brown clusters (BROWN) on the same vocabulary and use the resulting bit strings as features Brown *et al.* [1992].

data transformation. Our proposed method gives the best result among spectral methods and is competitive to other popular word embedding techniques.

This work suggests many directions for future work. Past spectral methods that involved CCA without data transformation (e.g., Cohen *et al.* [2013]) may be revisited with the square-root transformation. Using CCA to induce representations other than word embeddings is another important future work. It would also be interesting to formally investigate the theoretical merits and algorithmic possibility of solving the variance-weighted objective in Eq. (6.6). Even though the objective is hard to optimize in the worst case, it may be tractable under natural conditions.

Part III

Estimating Latent-Variable Models

Chapter 7

Spectral Learning of Anchor Hidden Markov Models

This chapter is adapted from joint work with Michael Collins and Daniel Hsu entitled "Unsupervised Part-Of-Speech Tagging with Anchor Hidden Markov Models" [Stratos *et al.*, 2016].

We tackle unsupervised part-of-speech (POS) tagging by learning hidden Markov models (HMMs) that are particularly well-suited for the problem. These HMMs, which we call *anchor HMMs*, assume that each tag is associated with at least one word that can have no other tag, which is a relatively benign condition for POS tagging (e.g., "the" is a word that appears only under the determiner tag). We exploit this assumption and extend the non-negative matrix factorization framework of Arora *et al.* [2012a] to design a consistent estimator for anchor HMMs. In experiments, our algorithm is competitive with strong baselines such as the clustering method of Brown *et al.* [1992] and the log-linear model of Berg-Kirkpatrick *et al.* [2010]. Furthermore, it produces an interpretable model in which hidden states are automatically lexicalized by words.

In this chapter, we use M_i denotes the i-th row of a matrix M (used as a column vector). Also, Δ^{m-1} denotes the $(m-1)$-dimensional probability simplex, that is, $\Delta^{m-1} := \{v \in \mathbb{R}^m : v \geq 0, \sum_i v_i = 1\}$.

7.1 Introduction

Part-of-speech (POS) tagging without supervision is a quintessential problem in unsupervised learning for natural language processing (NLP). A major application of this task is reducing annotation cost: for instance, it can be used to produce rough syntactic annotations for a new language that has no labeled data, which can be subsequently refined by human annotators.

Hidden Markov models (HMMs) are a natural choice of model and have been a workhorse for this problem. Early works estimated vanilla HMMs with standard unsupervised learning methods such as the expectation-maximization (EM) algorithm, but it quickly became clear that they performed very poorly in inducing POS tags [Merialdo, 1994]. Later works improved upon vanilla HMMs by incorporating specific structures that are well-suited for the task, such as a sparse prior [Johnson, 2007] or a hard-clustering assumption [Brown *et al.*, 1992].

In this work, we tackle unsupervised POS tagging with HMMs whose structure is deliberately suitable for POS tagging. These HMMs impose an assumption that each hidden state is associated with an observation state ("anchor word") that can appear under no other state. For this reason, we denote this class of restricted HMMs by *anchor HMMs*. Such an assumption is relatively benign for POS tagging; it is reasonable to assume that each POS tag has at least one word that occurs only under that tag. For example, in English, "the" is an anchor word for the determiner tag; "laughed" is an anchor word for the verb tag.

We build on the non-negative matrix factorization (NMF) framework of Arora *et al.* [2012a] to derive a consistent estimator for anchor HMMs. We make several new contributions in the process. First, to our knowledge, there is no previous work directly building on this framework to address unsupervised sequence labeling. Second, we generalize the NMF-based learning algorithm to obtain extensions that are important for empirical performance (Table 7.2). Third, we perform extensive experiments on unsupervised POS tagging and report competitive results against strong baselines such as the clustering method of Brown *et al.* [1992] and the log-linear model of Berg-Kirkpatrick *et al.* [2010].

One characteristic of the approach is the immediate interpretability of inferred hidden

states. Because each hidden state is associated with an observation, we can examine the set of such anchor observations to qualitatively evaluate the learned model. In our experiments on POS tagging, we find that anchor observations correspond to possible POS tags across different languages (Table 7.7). This property can be useful when we wish to develop a tagger for a new language that has no labeled data; we can label only the anchor words to achieve a complete labeling of the data.

This chapter is structured as follows. In Section 7.2, we define the model family of anchor HMMs. In Section 7.3, we derive a matrix decomposition algorithm for estimating the parameters of an anchor HMM. In Section 7.4, we present our experiments on unsupervised POS tagging. In Section 7.5, we discuss related works.

7.2 The Anchor Hidden Markov Model

Definition 7.2.1. *An **anchor HMM (A-HMM)** is a 6-tuple $(n, m, \pi, t, o, \mathcal{A})$ for positive integers n, m and functions $\pi, t, o, \mathcal{A}$ where*

- *$[n]$ is a set of observation states.*
- *$[m]$ is a set of hidden states.*
- *$\pi(h)$ is the probability of generating $h \in [m]$ in the first position of a sequence.*
- *$t(h'|h)$ is the probability of generating $h' \in [m]$ given $h \in [m]$.*
- *$o(x|h)$ is the probability of generating $x \in [n]$ given $h \in [m]$.*
- *$\mathcal{A}(h) := \{x \in [n] : \ o(x|h) > 0 \wedge o(x|h') = 0 \ \forall h' \neq h\}$ is non-empty for each $h \in [m]$.*

In other words, an A-HMM is an HMM in which each hidden state h is associated with at least one "anchor" observation state that can be generated by, and only by, h. Note that the anchor condition implies $n \geq m$.

An equivalent definition of an A-HMM is given by organizing the parameters in matrix form. Under this definition, an A-HMM has parameters (π, T, O) where $\pi \in \mathbb{R}^m$ is a vector and $T \in \mathbb{R}^{m \times m}, O \in \mathbb{R}^{n \times m}$ are matrices whose entries are set to:

- $\pi_h = \pi(h)$ for $h \in [m]$
- $T_{h',h} = t(h'|h)$ for $h, h' \in [m]$
- $O_{x,h} = o(x|h)$ for $h \in [m], x \in [n]$

The anchor condition implies that $\text{rank}(O) = m$. To see this, consider the rows $O_{a_1} \dots O_{a_m}$ where $a_h \in \mathcal{A}(h)$. Since each O_{a_h} has a single non-zero entry at the h-th index, the columns of O are linearly independent. We assume $\text{rank}(T) = m$.

An important special case of A-HMM introduced by Brown *et al.* [1992] is defined below. Under this A-HMM, every observation state is an anchor of some hidden state.

Definition 7.2.2. *A **Brown model** is an A-HMM in which $\mathcal{A}(1) \dots \mathcal{A}(m)$ partition $[n]$.*

7.3 Parameter Estimation for A-HMMs

We now derive an algorithm for learning A-HMMs. The algorithm reduces the learning problem to an instance of NMF from which the model parameters can be computed in closed-form.

7.3.1 NMF

We give a brief review of the NMF method of Arora *et al.* [2012a]. (Exact) NMF is the following problem: given an $n \times d$ matrix $A = BC$ where $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{m \times d}$ have non-negativity constraints, recover B and C. This problem is NP-hard in general [Vavasis, 2009], but Arora *et al.* [2012a] provide an exact and efficient method when A has the following special structure:

Condition 7.3.1. *A matrix $A \in \mathbb{R}^{n \times d}$ satisfies this condition if $A = BC$ for $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{m \times d}$ where*

1. *For each $x \in [n]$, $B_x \in \Delta^{m-1}$. That is, each row of B defines a probability distribution over $[m]$.*
2. *For each $h \in [m]$, there is some $a_h \in [n]$ such that $B_{a_h,h} = 1$ and $B_{a_h,h'} = 0$ for all $h' \neq h$.*

Anchor-NMF

Input: $A \in \mathbb{R}^{n \times d}$ satisfying Condition 7.3.1 with $A = BC$ for some $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{m \times d}$, value m

- For $h = 1 \dots m$, find a vertex a_h as

$$U \leftarrow \text{Gram-Schmidt}(\{A_{a_l}\}_{l=1}^{h-1})$$
$$a_h \leftarrow \arg\max_{x \in [n]} \left|\left| A_x - UU^\top A_x \right|\right|_2$$

  where Gram-Schmidt$(\{A_{a_l}\}_{l=1}^{h-1})$ is the Gram-Schmidt process that orthonormalizes $\{A_{a_l}\}_{l=1}^{h-1}$.

- For $x = 1 \dots n$, recover the x-th row of B as

$$\overline{B}_x \leftarrow \arg\min_{u \in \Delta^{m-1}} \left|\left| A_x - \sum_{h=1}^{m} u_h A_{a_h} \right|\right|_2 \tag{7.1}$$

- Set $\overline{C} = [A_{a_1} \dots A_{a_m}]^\top$.

Output: $\overline{B}$ and $\overline{C}$ such that $\overline{B}_h^\top = B_{\rho(h)}^\top$ and $\overline{C}_h = C_{\rho(h)}$ for some permutation $\rho : [m] \to [m]$

Figure 7.1: Non-negative matrix factorization algorithm of Arora *et al.* (2012).

3. $\text{rank}(C) = m$.

Since $\text{rank}(B) = \text{rank}(C) = m$ (by property 2 and 3), the matrix A must have rank m. Note that by property 1, each row of A is given by a convex combination of the rows of C: for $x \in [n]$,

$$A_x = \sum_{h=1}^{m} B_{x,h} \times C_h$$

Furthermore, by property 2 each $h \in [m]$ has an associated row $a_h \in [n]$ such that $A_{a_h} = C_{a_h}$. These properties can be exploited to recover B and C.

A concrete algorithm for factorizing a matrix satisfying Condition 7.3.1 is given in

Figure 7.1 [Arora *et al.*, 2012a]. It first identifies $a_1 \dots a_m$ (up to some permutation) by greedily locating the row of A furthest away from the subspace spanned by the vertices selected so far. Then it recovers each B_x as the convex coefficients required to combine $A_{a_1} \dots A_{a_m}$ to yield A_x. The latter computation (7.1) can be achieved with any constrained optimization method; we use the Frank-Wolfe algorithm [Frank and Wolfe, 1956]. See Arora *et al.* [2012a] for a proof of the correctness of this algorithm.

7.3.2 Random Variables

To derive our algorithm, we make use of certain random variables under the A-HMM. Let $(X_1, \dots, X_N) \in [n]^N$ be a random sequence of $N \geq 2$ observations drawn from the model, along with the corresponding hidden state sequence $(H_1, \dots, H_N) \in [m]^N$; independently, pick a position $I \in [N-1]$ uniformly at random. Let $Y_I \in \mathbb{R}^d$ be a d-dimensional vector which is conditionally independent of X_I given H_I, that is, $P(Y_I|H_I, X_I) = P(Y_I|H_I)$. We will provide how to define such a variable in Section 7.3.4.1: Y_I will be a function of $(X_1, \dots, X_N)$ serving as a "context" representation of X_I revealing the hidden state H_I.

Define unigram probabilities $u^\infty, u^1 \in \mathbb{R}^n$ and bigram probabilities $\mathcal{B} \in \mathbb{R}^{n \times n}$ where

$$\begin{aligned} u^\infty_x &:= P(X_I = x) && \forall x \in [n] \\ u^1_x &:= P(X_I = x | I = 1) && \forall x \in [n] \\ \mathcal{B}_{x,x'} &:= P(X_I = x, X_{I+1} = x') && \forall x, x' \in [n] \end{aligned}$$

Additionally, define $\bar{\pi} \in \mathbb{R}^m$ where

$$\bar{\pi}_h = P(H_I = h) \qquad \forall h \in [m] \tag{7.2}$$

We assume $\bar{\pi}_h > 0$ for all $h \in [m]$.

7.3.3 Derivation of a Learning Algorithm

The following proposition provides a way to use the NMF algorithm in Figure 7.1 to recover the emission parameters O up to scaling.

Proposition 7.3.1. *Let $X_I \in [n]$ and $Y_I \in \mathbb{R}^d$ be respectively an observation and a context vector drawn from the random process described in Section 7.3.2. Define a matrix $\Omega \in \mathbb{R}^{n \times d}$*

with rows

$$\Omega_x = \boldsymbol{E}[Y_I|X_I = x] \qquad \forall x \in [n] \tag{7.3}$$

If $\text{rank}(\Omega) = m$*, then* Ω *satisfies Condition 7.3.1:*

$$\Omega = \widetilde{O}\Theta$$

where $\widetilde{O}_{x,h} = P(H_I = h|X_I = x)$ *and* $\Theta_h = \boldsymbol{E}[Y_I|H_I = h]$.

Proof.

$$\begin{aligned}\boldsymbol{E}[Y_I|X_I = x] &= \sum_{h=1}^{m} P(H_I = h|X_I = x) \times \boldsymbol{E}[Y_I|H_I = h, X_I = x] \\ &= \sum_{h=1}^{m} P(H_I = h|X_I = x) \times \boldsymbol{E}[Y_I|H_I = h]\end{aligned}$$

The last equality follows by the conditional independence of Y_I. This shows $\Omega = \widetilde{O}\Theta$. By the anchor assumption of the A-HMM, each $h \in [m]$ has at least one $x \in \mathcal{A}(h)$ such that $P(H_I = h|X_I = x) = 1$, thus Ω satisfies Condition 7.3.1. □

A useful interpretation of Ω in Proposition 7.3.1 is that its rows $\Omega_1 \dots \Omega_n$ are d-dimensional vector representations of observation states forming a convex hull in $\mathbb{R}^d$. This convex hull has m vertices $\Omega_{a_1} \dots \Omega_{a_m}$ corresponding to anchors $a_h \in \mathcal{A}(h)$ which can be convexly combined to realize all $\Omega_1 \dots \Omega_n$.

Given $\widetilde{O}$, we can recover the A-HMM parameters as follows. First, we recover the stationary state distribution $\bar{\pi}$ as:

$$\bar{\pi}_h = \sum_{x \in [n]} P(H_I = h|X_I = x) \times P(X_I = x) = \sum_{x \in [n]} \widetilde{O}_{x,h} \times u_x^{\infty}$$

The emission parameters O are given by Bayes' theorem:

$$O_{x,h} = \frac{P(H_I = h|X_I = x) \times P(X_I = x)}{\sum_{x \in [n]} P(H_I = h|X_I = x) \times P(X_I = x)} = \frac{\widetilde{O}_{x,h} \times u_x^{\infty}}{\bar{\pi}_h}$$

Using the fact that the emission probabilities are position-independent, we see that the initial state distribution π satisfies $u^1 = O\pi$:

$$u_x^1 = \sum_{h \in [m]} P(X_I = x|H_I = h, I = 1) \times P(H_I = h|I = 1) = \sum_{h \in [m]} O_{x,h} \times \pi_h$$

Learn-Anchor-HMM

Input: Ω in Proposition 7.3.1, number of hidden states m, bigram probabilities $\mathcal{B}$, unigram probabilities u^∞, u^1

- Compute $(\widetilde{O}, \Theta) \leftarrow$ **Anchor-NMF**(Ω, m).
- Recover the parameters:

$$\bar{\pi} \leftarrow \widetilde{O}^\top u^\infty \tag{7.4}$$

$$O \leftarrow \operatorname{diag}(\bar{\pi})^{-1} \operatorname{diag}(u^\infty)\widetilde{O} \tag{7.5}$$

$$\pi = O^+ u^1 \tag{7.6}$$

$$T \leftarrow (\operatorname{diag}(\bar{\pi})^{-1} O^+ \mathcal{B}(O^\top)^+)^\top \tag{7.7}$$

Output: A-HMM parameters (π, T, O)

Figure 7.2: NMF-based learning algorithm for A-HMMs. The algorithm **Anchor-NMF** is given in Figure 7.1.

Thus π can be recovered as $\pi = O^+ u^1$. Finally, it can be algebraically verified that $\mathcal{B} = O \operatorname{diag}(\bar{\pi}) T^\top O^\top$ [Hsu *et al.*, 2012]. Since all the involved matrices have rank m, we can directly solve for T as

$$T = (\operatorname{diag}(\bar{\pi})^{-1} O^+ \mathcal{B}(O^\top)^+)^\top$$

Figure 7.2 shows the complete algorithm. As input, it receives a matrix Ω satisfying Proposition 7.3.1, the number of hidden states, and the probabilities of observed unigrams and bigrams. It first decomposes Ω using the NMF algorithm in Figure 7.1. Then it computes the A-HMM parameters whose solution is given analytically.

The following theorem guarantees the consistency of the algorithm.

Theorem 7.3.1. *Let (π, T, O) be an A-HMM such that* $\operatorname{rank}(T) = m$ *and $\bar{\pi}$ defined in* (7.2) *has strictly positive entries $\bar{\pi}_h > 0$. Given random variables Ω satisfying Proposition 7.3.1 and $\mathcal{B}, u^\infty, u^1$ under this model, the algorithm* ***Learn-Anchor-HMM*** *in Figure 7.2 outputs (π, T, O) up to a permutation on hidden states.*

Proof. By Proposition 7.3.1, Ω satisfies Condition 7.3.1 with $\Omega = \widetilde{O}\Theta$, thus $\widetilde{O}$ can be recovered up to a permutation on columns with the algorithm **Anchor-NMF**. The consistency of the recovered parameters follows from the correctness of (7.4–7.7) under the rank conditions. □

7.3.3.1 Constrained Optimization for π and T

Note that (7.6) and (7.7) require computing the pseudoinverse of the estimated O, which can be expensive and vulnerable to sampling errors in practice. To make our parameter estimation more robust, we can explicitly impose probability constraints. We recover π by solving:

$$\pi = \underset{\pi' \in \Delta^{m-1}}{\arg\min} \left|\left| u^1 - O\pi' \right|\right|_2 \tag{7.8}$$

which can again be done with algorithms such as Frank-Wolfe. We recover T by maximizing the log likelihood of observation bigrams

$$\sum_{x,x'} \mathcal{B}_{x,x'} \log \left(\sum_{h,h' \in [m]} \bar{\pi}_h O_{x,h} T_{h',h} O_{x',h'} \right) \tag{7.9}$$

subject to the constraint $(T^\top)_h \in \Delta^{m-1}$. Since (7.9) is concave in T with other parameters O and $\bar{\pi}$ fixed, we can use EM to find the global optimum.

7.3.4 Construction of the Convex Hull Ω

In this section, we provide several ways to construct a convex hull Ω satisfying Proposition 7.3.1.

7.3.4.1 Choice of the Context Y_I

In order to satisfy Proposition 7.3.1, we need to define the context variable $Y_I \in \mathbb{R}^d$ with two properties:

- $P(Y_I|H_I, X_I) = P(Y_I|H_I)$
- The matrix Ω with rows

$$\Omega_x = \boldsymbol{E}[Y_I|X_I = x] \qquad \forall x \in [n]$$

has rank m.

A simple construction [Arora *et al.*, 2012a] is given by defining $Y_I \in \mathbb{R}^n$ to be an indicator vector for the next observation:

$$[Y_I]_{x'} = \begin{cases} 1 & \text{if } X_{I+1} = x' \\ 0 & \text{otherwise} \end{cases} \tag{7.10}$$

The first condition is satisfied since X_{I+1} does not depend on X_I given H_I. For the second condition, observe that $\Omega_{x,x'} = P(X_{I+1} = x' | X_I = x)$, or in matrix form

$$\Omega = \operatorname{diag}(u^\infty)^{-1} \mathcal{B} \tag{7.11}$$

Under the rank conditions in Theorem 7.3.1, (7.11) has rank m.

More generally, we can let Y_I be an observation (encoded as an indicator vector as in (7.10)) randomly drawn from a window of $L \in \mathbb{N}$ nearby observations. We can either only use the identity of the chosen observation (in which case $Y_I \in \mathbb{R}^n$) or additionally indicate the relative position in the window (in which case $Y_I \in \mathbb{R}^{nL}$). It is straightforward to verify that the above two conditions are satisfied under these definitions. Clearly, (7.11) is a special case with $L = 1$.

7.3.4.2 Reducing the Dimension of Ω_x

With the definition of Ω in the previous section, the dimension of Ω_x is $d = O(n)$ which can be difficult to work with when $n \gg m$. Proposition 7.3.1 allows us to reduce the dimension as long as the final matrix retains the form in (7.3) and has rank m. In particular, we can multiply Ω by any rank-m projection matrix $\Pi \in \mathbb{R}^{d \times m}$ on the right side: if Ω satisfies the properties in Proposition 7.3.1, then so does $\Omega\Pi$ with m-dimensional rows

$$(\Omega\Pi)_x = \boldsymbol{E}[Y_I \Pi | X_I = x]$$

Since $\operatorname{rank}(\Omega) = m$, a natural choice of Π is the projection onto the best-fit m-dimensional subspace of the row space of Ω.

We mention that previous works on the NMF-learning framework have employed various projection methods, but they do not examine relative merits of their choices. For

instance, Arora *et al.* [2012a] simply use random projection, which is convenient for theoretical analysis. Cohen and Collins [2014] use a projection based on canonical correlation analysis (CCA) without further exploration. In contrast, we give a full comparison of valid construction methods and find that the choice of Ω is crucial in practice.

7.3.4.3 Construction of Ω for the Brown Model

We can formulate an alternative way to construct a valid Ω when the model is further restricted to be a Brown model. Since every observation is an anchor, $O_x \in \mathbb{R}^m$ has a single nonzero entry for every x. Thus the rows defined by $\Omega_x = O_x / ||O_x||$ (an indicator vector for the unique hidden state of x) form a trivial convex hull in which every point is a vertex. This corresponds to choosing an oracle context $Y_I \in \mathbb{R}^m$ where

$$[Y_I]_h = \begin{cases} 1 & \text{if } H_I = h \\ 0 & \text{otherwise} \end{cases}$$

It is possible to recover the Brown model parameters O up to element-wise scaling and rotation of rows using the algorithm of Stratos *et al.* [2015]. More specifically, let $f(O) \in \mathbb{R}^{n \times m}$ denote the output of their algorithm. Then they show that for some vector $s \in \mathbb{R}^m$ with strictly positive entries and an orthogonal matrix $Q \in \mathbb{R}^{m \times m}$:

$$f(O) = O^{\langle 1/4 \rangle} \operatorname{diag}(s) Q^\top$$

where $O^{\langle 1/4 \rangle}$ is an element-wise exponentiation of O by $1/4$. Since the rows of $f(O)$ are simply some scaling and rotation of the rows of O, using $\Omega_x = f(O)_x / ||f(O)_x||$ yields a valid Ω.

While we need to impose an additional assumption (the Brown model restriction) in order to justify this choice of Ω, we find in our experiments that it performs better than other alternatives. We speculate that this is because a Brown model is rather appropriate for the POS tagging task; many words are indeed unambiguous with respect to POS tags (Table 7.4). Also, the general effectiveness of $f(O)$ for representational purposes has been demostrated in previous works [Stratos *et al.*, 2014, 2015]. By restricting the A-HMM to be a Brown model, we can piggyback on the proven effectiveness of $f(O)$.

Input: bigram probabilities $\mathcal{B}$, unigram probabilities u^∞, number of hidden states m, construction method τ

Scaled Matrices: ($\sqrt{\cdot}$ is element-wise)

$$\overline{\mathcal{B}} := \operatorname{diag}(u^\infty)^{-1/2}\,\mathcal{B}\operatorname{diag}(u^\infty)^{-1/2}$$
$$\widetilde{\mathcal{B}} := \operatorname{diag}\left(\sqrt{u^\infty}\right)^{-1/2}\sqrt{\mathcal{B}}\operatorname{diag}\left(\sqrt{u^\infty}\right)^{-1/2}$$

Singular Vectors: $U(M)$ ($V(M)$) is an $n \times m$ matrix of the left (right) singular vectors of M corresponding to the largest m singular values

- If $\tau \neq$ **brown**: set

$$\Omega \leftarrow \operatorname{diag}(u^\infty)^{-1}\,\mathcal{B}\Pi$$

  where the projection matrix $\Pi \in \mathbb{R}^{n\times m}$ is given by

$$\begin{aligned} \Pi_{i,j} &\sim \mathcal{N}(0, 1/m) && \text{if } \tau = \textbf{random} \\ \Pi &= V(\operatorname{diag}(u^\infty)^{-1}\,\mathcal{B}) && \text{if } \tau = \textbf{best-fit} \\ \Pi &= \operatorname{diag}(u^\infty)^{-1/2}\,V(\overline{\mathcal{B}}) && \text{if } \tau = \textbf{cca} \end{aligned}$$

- If $\tau =$ **brown**: compute the transformed emission matrix as $f(O) = U(\widetilde{\mathcal{B}})$ and set

$$\Omega \leftarrow \operatorname{diag}(v)^{-1} f(O)$$

  where $v_x := ||f(O)_x||_2$ is the length of the x-th row of $f(O)$.

Output: $\Omega \in \mathbb{R}^{n\times m}$ in Proposition 7.3.1

Figure 7.3: Algorithm for constructing a valid Ω with different construction methods. For simplicity, we only show the bigram construction (context size $L = 1$), but an extension for larger context ($L > 1$) is straightforward.

Figure 7.3 shows an algorithm for constructing Ω with these different construction methods. For simplicity, we only show the bigram construction (context size $L = 1$), but an ex-

tension for larger context ($L > 1$) is straightforward as discussed earlier. The construction methods **random** (random projection), **best-fit** (projection to the best-fit subspace), and **cca** (CCA projection) all compute (7.11) and differ only in how the dimension is reduced. The construction method **brown** computes the transformed Brown parameters $f(O)$ as the left singular vectors of a scaled covariance matrix and then normalizes its rows. We direct the reader to Theorem 6.3.1 for a derivation of this calculation.

7.3.4.4 Ω with Feature Augmentation

The x-th row of Ω is a d-dimensional vector representation of x lying in a convex set with m vertices. This suggests a natural way to incorporate domain-specific features: we can add additional dimensions that provide information about hidden states from the surface form of x.

For instance, consider the the POS tagging task. In the simple construction (7.11), the representation of word x is defined in terms of neighboring words x':

$$[\Omega_x]_{x'} = \boldsymbol{E}\left[\, [[X_{I+1} = x']] \,|X_I = x\right]$$

We can augment this vector with s additional dimensions indicating the spelling features of x. For instance, the $(n+1)$-th dimension may be defined as:

$$[\Omega_x]_{n+1} = \boldsymbol{E}\left[\, [[x \text{ ends in “ing”}]] \,|X_I = x\right]$$

This value will be generally large for verbs and small for non-verbs, nudging verbs closer together and away from non-verbs. The modified $(n+s)$-dimensional representation is followed by the usual dimension reduction. Note that the spelling features are a deterministic function of a word, and we are implicitly assuming that they are independent of the word given its tag. While this is of course not true in practice, we find that these features can significantly boost the tagging performance.

7.4 Experiments

We evaluate our A-HMM learning algorithm on the task of unsupervised POS tagging. The goal of this task is to induce the correct sequence of POS tags (hidden states) given a

sequence of words (observation states). The anchor condition corresponds to assuming that each POS tag has at least one word that occurs only under that tag.

7.4.1 Background on Unsupervised POS Tagging

Unsupervised POS tagging has long been an active area of research [Smith and Eisner, 2005a; Johnson, 2007; Toutanova and Johnson, 2007; Haghighi and Klein, 2006; Berg-Kirkpatrick *et al.*, 2010], but results on this task are complicated by varying assumptions and unclear evaluation metrics [Christodoulopoulos *et al.*, 2010]. Rather than addressing multiple alternatives for evaluating unsupervised POS tagging, we focus on a simple and widely used metric: many-to-one accuracy (i.e., we map each hidden state to the most frequently coinciding POS tag in the labeled data and compute the resulting accuracy).

7.4.1.1 Better Model v.s. Better Learning

Vanilla HMMs are notorious for their mediocre performance on this task, and it is well known that they perform poorly largely because of *model misspecification*, not because of suboptimal parameter estimation (e.g., because EM gets stuck in local optima). More generally, a large body of work points to the inappropriateness of simple generative models for unsupervised induction of linguistic structure [Merialdo, 1994; Smith and Eisner, 2005b; Liang and Klein, 2008].

Consequently, many works focus on using more expressive models such as log-linear models [Smith and Eisner, 2005a; Berg-Kirkpatrick *et al.*, 2010] and Markov random fields (MRF) [Haghighi and Klein, 2006]. These models are shown to deliver good performance even though learning is approximate. Thus one may question the value of having a consistent estimator for A-HMMs and Brown models in this work: if the model is wrong, what is the point of learning it accurately?

However, there is also ample evidence that HMMs are competitive for unsupervised POS induction when they incorporate domain-specific structures. Johnson [2007] is able to outperform the sophisticated MRF model of Haghighi and Klein [2006] on one-to-one accuracy by using a sparse prior in HMM estimation. The clustering method of Brown *et al.* [1992] which is based on optimizing the likelihood under the Brown model (a special

	de	en	es	fr	id	it	ja	ko	pt-br	sv
# tokens	$293k$	$1047k$	$424k$	$397k$	$122k$	$168k$	$92k$	$70k$	$298k$	$96k$
# types	$52k$	$46k$	$50k$	$45k$	$22k$	$22k$	$57k$	$36k$	$34k$	$16k$
ratio	5.6	22.6	8.4	8.9	5.5	7.5	1.6	1.9	8.8	5.9

Table 7.1: Numbers of word tokens and types across 10 languages in the universal treebank dataset (version 2.0).

case of HMM) remains a baseline difficult to outperform [Christodoulopoulos *et al.*, 2010].

We add to this evidence by demonstrating the effectiveness of A-HMMs on this task. We also check the anchor assumption on data and show that the A-HMM model structure is in fact appropriate for the problem (Table 7.4).

7.4.2 Experimental Setting

We use the universal treebank dataset (version 2.0) which contains sentences annotated with 12 POS tag types for 10 languages [McDonald *et al.*, 2013]. Table 7.1 shows word statistics across languages ("de" for German, "en" for English, "es" for Spanish, "fr" for French, "id" for Indonesian, "it" for Italian, "ja" for Japanese, "ko" for Korean, "pt-br" for Portuguese-Brazilian, and "sv" for Swedish). Note that the amount of data differs across different languages. Also, the ratio between the number of word tokens and the number of word types is very different. In particular, the ratio is very small for Japanese and Korean due to segmentation issues. While we would like to have similar qualities for all languages, we can isolate the issue of data by comparing different models on the same dataset.

All models are trained with 12 hidden states. We use the English portion to experiment with different hyperparameter configurations. At test time, we fix a configuration (based on the English portion) and apply it across all languages.

The list of compared methods is given below:

BW The Baum-Welch algorithm, an EM algorithm for HMMs [Baum and Petrie, 1966].

CLUSTER A parameter estimation scheme for HMMs based on Brown clustering [Brown *et al.*, 1992]. We run the Brown clustering algorithm[1] to obtain 12 word clusters $C_1 \dots C_{12}$. Then we set the emission parameters $o(x|h)$, transition parameters $t(h'|h)$, and prior $\pi(h)$ to be the maximum-likelihood estimates under the fixed clusters.

ANCHOR Our algorithm **Learn-Anchor-HMM** in Figure 7.2 but with the constrained optimization (7.8) and (7.9) for estimating π and T.[2]

ANCHOR-FEAT Same as ANCHOR but employs the feature augmentation scheme described in Section 7.3.4.4.

LOG-LINEAR The unsupervised log-linear model described in Berg-Kirkpatrick *et al.* [2010]. Instead of emission parameters $o(x|h)$, the model maintains a miniature log-linear model with a weight vector w and a feature function ϕ. The probability of a word x given tag h is computed as

$$p(x|h) = \frac{\exp(w^\top \phi(x,h))}{\sum_{x \in [n]} \exp(w^\top \phi(x,h))}$$

The model can be trained by maximizing the likelihood of observed sequences. We use L-BFGS to directly optimize this objective.[3] This approach obtains the current state-of-the-art accuracy on fine-grained (45 tags) English WSJ dataset.

We use maximum marginal decoding for HMM predictions: that is, at each position, we predict the most likely tag given the entire sentence.

7.4.3 Practical Issues with the Anchor Algorithm

In our experiments, we find that **Anchor-NMF** (Figure 7.1) tends to propose extremely rare words as anchors. A simple fix is to search for anchors only among relatively frequent words. We find that any reasonable frequency threshold works well; we use the 300 most

[1] We use the implementation of Liang [2005].

[2] The code is available at `https://github.com/karlstratos/anchor`.

[3] We use the implementation of Berg-Kirkpatrick *et al.* [2010] (personal communication).

Choice of Ω	Accuracy
Random	48.2
Best-Fit	53.4
CCA	57.0
Brown	**66.1**

Table 7.2: Many-to-one accuracy on the English data with different choices of the convex hull Ω (Figure 7.3). These results do not use spelling features.

frequent words. Note that this is not a problem if these 300 words include anchor words corresponding to all the 12 tags.

We must define the context for constructing Ω. We use the previous and next words (i.e., context size $L = 2$) marked with relative positions. Thus Ω has $2n$ columns before dimension reduction. Table 7.2 shows the performance on the English portion with different construction methods for Ω. The Brown construction ($\tau =$ **brown** in Figure 7.3) clearly performs the best: essentially, the anchor algorithm is used to extract the HMM parameters from the CCA-based word embeddings of Stratos *et al.* [2015].

We also explore feature augmentation discussed in Section 7.3.4.4. For comparison, we employ the same word features used by Berg-Kirkpatrick *et al.* [2010]:

- Indicators for whether a word is capitalized, contains a hyphen, or contains a digit
- Suffixes of length 1, 2, and 3

We weigh the l_2 norm of these extra dimensions in relation to the original dimensions: we find a small weight (e.g., 0.1 of the norm of the original dimensions) works well. We also find that these features can sometimes significantly improve the performance. For instance, the accuracy on the English portion can be improved from 66.1% to 71.4% with feature augmentation.

Another natural experiment is to refine the HMM parameters obtained from the anchor algorithm (or Brown clusters) with a few iterations of the Baum-Welch algorithm. In our experiments, however, it did not significantly improve the tagging performance, so we omit this result.

Model	de	en	es	fr	id	it	ja	ko	pt-br	sv
BW	(4.8) 45.5	(3.4) 59.8	(2.2) 60.6	(3.6) 60.1	(3.1) 49.6	(2.6) 51.5	(2.1) 59.5	(0.6) 51.7	(3.7) 59.5	(3.0) 42.4
CLUSTER	60.0	62.9	67.4	66.4	59.3	**66.1**	60.3	47.5	**67.4**	**61.9**
ANCHOR	61.1	66.1	69.0	68.2	63.7	60.4	65.3	53.8	64.9	51.1
ANCHOR-FEAT	63.4	**71.4**	**74.3**	**71.9**	**67.3**	60.2	69.4	**61.8**	65.8	61.0
LOG-LINEAR	(1.8) **67.5**	(3.5) 62.4	(3.1) 67.1	(4.5) 62.1	(3.9) 61.3	(2.9) 52.9	(2.9) **78.2**	(3.6) 60.5	(2.2) 63.2	(2.5) 56.7

Table 7.3: Many-to-one accuracy on each language using 12 universal tags. The first four models are HMMs estimated with the Baum-Welch algorithm (BW), the clustering algorithm of Brown *et al.* [1992], the anchor algorithm without (ANCHOR) and with (ANCHOR-FEAT) feature augmentation. LOG-LINEAR is the model of Berg-Kirkpatrick *et al.* [2010] trained with the direct-gradient method using L-BFGS. For BW and LOG-LINEAR, we report the mean and the standard deviation (in parentheses) of 10 random restarts run for 1,000 iterations.

7.4.4 Tagging Accuracy

Table 7.3 shows the many-to-one accuracy on all languages in the dataset. For the Baum-Welch algorithm and the unsupervised log-linear models, we report the mean and the standard deviation (in parentheses) of 10 random restarts run for 1,000 iterations.

Both ANCHOR and ANCHOR-FEAT compete favorably. On 5 out of 10 languages, ANCHOR-FEAT achieves the highest accuracy, often closely followed by ANCHOR. The Brown clustering estimation is also competitive and has the highest accuracy on 3 languages. Not surprisingly, vanilla HMMs trained with BW perform the worst (see Section 7.4.1.1 for a discussion).

LOG-LINEAR is a robust baseline and performs the best on the remaining 2 languages. It performs especially strongly on Japanese and Korean datasets in which poorly segmentated strings such as “1950年11月5日には” (on November 5, 1950) and “40.3%로” (by 40.3%) abound. In these datasets, it is crucial to make effective use of morphological features.

7.4.5 Qualitative Analysis

7.4.5.1 A-HMM Parameters

An A-HMM can be easily interpreted since each hidden state is marked with an anchor observation. Table 7.7 shows the 12 anchors found in each language. Note that these anchor words generally have a wide coverage of possible POS tags.

We also experimented with using true anchor words (obtained from labeled data), but they did not improve performance over automatically induced anchors. Since anchor discovery is inherently tied to parameter estimation, it is better to obtain anchors in a data-driven manner. In particular, certain POS tags (e.g., `X`) appear quite infrequently, and the model is worse off by being forced to allocate a hidden state for such a tag.

Table 7.8 shows words with highest emission probabilities $o(x|h)$ under each anchor. We observe that an anchor is representative of a certain group of words. For instance, the state "loss" represents noun-like words, "1" represents numbers, "on" represents preposition-like words, "one" represents determiner-like words, and "closed" represents verb-like words. The conditional distribution is peaked for anchors that represent function tags (e.g., determiners, punctuation) and flat for anchors that represent content tags (e.g., nouns). Occasionally, an anchor assigns high probabilities to words that do not seem to belong to the corresponding POS tag. But this is to be expected since $o(x|h) \propto P(X_I = x)$ is generally larger for frequent words.

7.4.5.2 Model Assumptions on Data

Table 7.4 checks the assumptions in A-HMMs and Brown models on the universal treebank dataset. The anchor assumption is indeed satisfied with 12 universal tags: in every language, each tag has at least one word uniquely associated with the tag. The Brown assumption (each word has exactly one possible tag) is of course not satisfied, since some words are genuinely ambiguous with respect to their POS tags. However, the percentage of unambiguous words is very high (well over 90%). This analysis supports that the model assumptions made by A-HMMs and Brown models are appropriate for POS tagging.

Table 7.5 reports the log likelihood (normalized by the number of words) on the En-

glish portion of different estimation methods for HMMs. BW and CLUSTER obtain higher likelihood than the anchor algorithm, but this is expected given that both EM and Brown clustering directly optimize likelihood. In contrast, the anchor algorithm is based on the method of moments and does not (at least directly) optimize likelihood. Note that high likelihood does not imply high accuracy under HMMs.

7.5 Related Work

7.5.1 Latent-Variable Models

There has recently been great progress in estimation of models with latent variables. Despite the NP-hardness in general cases [Terwijn, 2002; Arora *et al.*, 2012b], many algorithms with strong theoretical guarantees have emerged under natural assumptions. For example, for HMMs with full-rank conditions, Hsu *et al.* [2012] derive a consistent estimator of the marginal distribution of observed sequences. With similar non-degeneracy conditions, Anandkumar *et al.* [2014] propose an exact tensor decomposition method for learning a wide class of latent variable models. For topic models with a certain parameter structure, Arora *et al.* [2012a] derive a provably correct learning algorithm.

The anchor-based framework has been originally formulated for learning topic models [Arora *et al.*, 2012a]. It has been subsequently adopted to learn other models such as latent-variable probabilistic context-free grammars [Cohen and Collins, 2014]. In our work, we have extended this framework to address unsupervised sequence labeling.

Zhou *et al.* [2014] also extend Arora *et al.* [2012a]'s framework to learn various models including HMMs, but they address a more general problem. Consequently, their algorithm draws from Anandkumar *et al.* [2012c] and is substantially different from ours.

7.5.2 Unsupervised POS Tagging

Unsupervised POS tagging is a classic problem in unsupervised learning that has been tackled with various approaches. Johnson [2007] observes that EM performs poorly in this task because it induces flat distributions; this is not the case with our algorithm as seen in the peaky distributions in Table 7.8. Haghighi and Klein [2006] assume a set of prototypical

	de	en	es	fr	id	it	ja	ko	pt-br	sv
% anchored tags	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
% unambig words	96.6	91.5	94.0	94.2	94.8	94.8	99.5	98.4	94.8	97.4

.	VERB	PRON	ADP	NOUN	ADV	CONJ	DET	NUM	ADJ	X	PRT
,	said	it	from	Mr.	n't	or	which	billion	new	bono	na
53928	6339	5147	4856	4436	3582	2748	2458	1935	1542	8	5

Table 7.4: Verifying model assumptions on the universal treebank. The anchor assumption is satisfied in every language. The Brown assumption (each word has exactly one possible tag) is violated but not by a large margin. The lower table shows the most frequent anchor word and its count under each tag on the English portion.

Model	Normalized LL	Acc
BW	-6.45	59.8
CLUSTER	-6.71	62.9
ANCHOR	-7.06	66.1
ANCHOR-FEAT	-7.05	71.4

Table 7.5: Log likelihood normalized by the number of words on English (along with accuracy). For BW, we report the mean of 10 random restarts run for 1,000 iterations.

words for each tag and report high accuracy. In contrast, our algorithm automatically finds such prototypes in a subroutine.

Berg-Kirkpatrick *et al.* [2010] achieve the state-of-the-art result in unsupervised fine-grained POS tagging (mid-70%). As described in Section 7.4.2, their model is an HMM in which probabilties are given by log-linear models. Table 7.6 provides a point of reference comparing our work with Berg-Kirkpatrick *et al.* [2010] in their setting: models are trained and tested on the entire 45-tag WSJ dataset. Their model outperforms our approach in this setting: with fine-grained tags, spelling features become more important, for instance to distinguish "played" (VBD) from "play" (VBZ). Nonetheless, we have shown that our approach is competitive when universal tags are used (Table 7.3).

Many past works on POS induction predate the introduction of the universal tagset by

Models	Accuracy
BW	62.6 (1.1)
CLUSTER	65.6
ANCHOR	67.2
ANCHOR-FEAT	67.7
LOG-LINEAR	74.9 (1.5)

Table 7.6: Many-to-one accuracy on the English data with 45 original tags. We use the same setting as in Table 7.3. For BW and LOG-LINEAR, we report the mean and the standard deviation (in parentheses) of 10 random restarts run for 1,000 iterations.

Petrov *et al.* [2011] and thus report results with fine-grained tags. More recent works adopt the universal tagset but they leverage additional resources. For instance, Das and Petrov [2011] and Täckström *et al.* [2013] use parallel data to project POS tags from a supervised source language. Li *et al.* [2012] use tag dictionaries built from Wiktionary. Thus their results are not directly comparable to ours.[4]

7.6 Conclusion

We have presented an exact estimation method for learning anchor HMMs from unlabeled data. There are several directions for future work. An important direction is to extend the method to a richer family of models such as log-linear models or neural networks. Another direction is to further generalize the method to handle a wider class of HMMs by relaxing the anchor condition (Condition 7.3.1). This will require a significant extension of the NMF algorithm in Figure 7.1.

[4]Das and Petrov [2011] conduct unsupervised experiments using the model of Berg-Kirkpatrick *et al.* [2010], but their dataset and evaluation method differ from ours.

de	en	es	fr	id	it	ja	ko	pt-br	sv
empfehlen	loss	y	avait	bulan	radar	お世話に	완전	E	och
wie	1	hizo	commune	tetapi	però	ないと	중에	de	bör
;	on	-	Le	wilayah	sulle	ことにより	경우	partida	grund
Sein	one	especie	de	-	-	されている。	줄	fazer	mellan
Berlin	closed	Además	président	Bagaimana	Stati	ものを	같아요	meses	i
und	are	el	qui	,	Lo	,	많은	os	sociala
,	take	países	(	sama	legge	した	,	:	.
-	,	la	à	.	al	それは	볼	diretor	bli
der	vice	España	États	dan	far-	、	자신의	2010	den
im	to	en	Unis	Utara	di	幸福の	받고	,	,
des	York	de	Cette	pada	la	ことが	맛있는	uma	tid
Region	Japan	municipio	quelques	yang	art.	通常の	위한	O	Detta

Table 7.7: Anchor words found in each language (model ANCHOR-FEAT).

loss	year (.02)	market (.01)	share (.01)	company (.01)	stock (.01)	quarter (.01)	shares (.01)	price (.01)
1	1 (.03)	10 (.02)	30 (.02)	15 (.02)	8 (.02)	2 (.01)	20 (.01)	50 (.01)
on	of (.14)	in (.12)	. (.08)	for (.06)	on (.04)	by (.04)	from (.04)	and (.03)
one	the (.23)	a (.12)	“ (.03)	an (.03)	$ (.02)	its (.02)	that (.02)	this (.02)
closed	said (.05)	’s (.02)	is (.02)	says (.02)	was (.01)	has (.01)	had (.01)	expected (.01)
are	and (.08)	is (.08)	are (.05)	was (.04)	’s (.04)	“ (.04)	has (.03)	of (.03)
take	be (.04)	% (.02)	have (.02)	million (.02)	But (.02)	do (.01)	The (.01)	make (.01)
,	, (.53)	. (.25)	and (.05)	” (.04)	% (.01)	million (.01)	– (.01)	that (.01)
vice	’s (.03)	The (.02)	“ (.02)	New (.01)	and (.01)	new (.01)	first (.01)	chief (.01)
to	to (.39)	. (.11)	a (.06)	will (.04)	$ (.03)	n’t (.03)	would (.02)	% (.02)
York	the (.15)	a (.05)	The (.04)	of (.04)	’s (.04)	million (.01)	% (.01)	its (.01)
Japan	Mr. (.03)	it (.02)	” (.02)	$ (.02)	he (.02)	that (.02)	which (.01)	company (.01)

Table 7.8: Most likely words under each anchor word (English model ANCHOR-FEAT). Emission probabilities $o(x|h)$ are given in parentheses.

Chapter 8

Spectral Learning of Refinement Hidden Markov Models

This chapter is adapted from joint work with Alexander Rush, Shay Cohen, and Michael Collins entitled "Spectral learning of refinement HMMs" [Stratos *et al.*, 2013].

We derive a spectral algorithm for learning the parameters of a refinement HMM. This method is simple, efficient, and can be applied to a wide range of supervised sequence labeling tasks. Like other spectral methods, it avoids the problem of local optima and provides a consistent estimate of the parameters. Our experiments on a phoneme recognition task show that when equipped with informative feature functions, it performs significantly better than a supervised HMM and competitively with EM.

In this chapter, $C \in \mathbb{R}^{m \times m \times m}$ denotes a third-order tensor, that is, a set of m^3 values $C_{i,j,k}$ for $i, j, k \in [m]$. $C(v)$ denotes the $m \times m$ matrix with $[C(v)]_{i,j} = \sum_{k \in [m]} C_{i,j,k} v_k$. Finally, $C = xy^\top z^\top$ where $x, y, z \in \mathbb{R}^m$ denotes an $m \times m \times m$ tensor with $[C]_{i,j,k} = x_i y_j z_k$.

8.1 Introduction

Consider the task of supervised sequence labeling. We are given a training set where the j'th training example consists of a sequence of observations $x_1^{(j)} ... x_N^{(j)}$ paired with a sequence of labels $a_1^{(j)} ... a_N^{(j)}$ and asked to predict the correct labels on a test set of observations. A common approach is to learn a joint distribution over sequences $p(a_1 \dots a_N, x_1 \dots x_N)$ as a

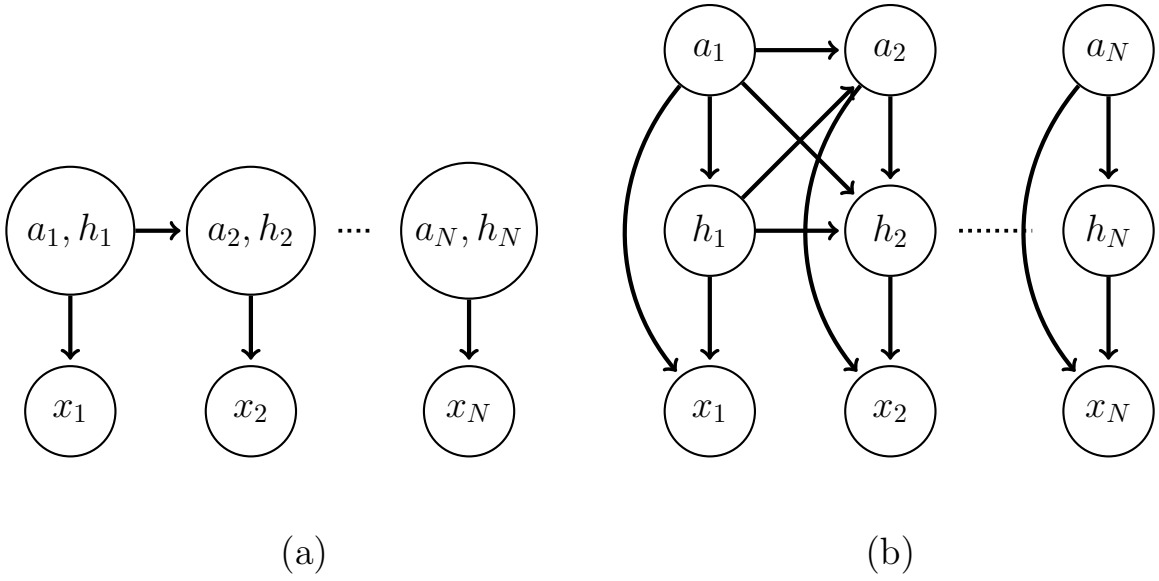


Figure 8.1: (a) An R-HMM chain. (b) An equivalent representation where labels and hidden states are intertwined.

hidden Markov model (HMM). The downside of HMMs is that they assume each label a_i is independent of labels before the previous label a_{i-1}. This independence assumption can be limiting, particularly when the label space is small. To relax this assumption we can refine each label a_i with a hidden state h_i, which is not observed in the training data, and model the joint distribution $p(a_1 \dots a_N, x_1 \dots x_N, h_1 \dots h_N)$. This refinement HMM (R-HMM), illustrated in Figure 8.1, is able to propagate information forward through the hidden state as well as the label.

Unfortunately, estimating the parameters of an R-HMM is complicated by the unobserved hidden variables. A standard approach is to use the expectation-maximization (EM) algorithm which has no guarantee of finding the global optimum of its objective function. The problem of local optima prevents EM from yielding statistically consistent parameter estimates: even with very large amounts of data, EM is not guaranteed to estimate parameters which are close to the "correct" model parameters.

In this work, we derive a spectral algorithm for learning the parameters of R-HMMs.

Unlike EM, this technique is guaranteed to find the true parameters of the underlying model under mild conditions on the singular values of the model. The algorithm we derive is simple and efficient, relying on singular value decomposition followed by standard matrix operations.

We also describe the connection of R-HMMs to L-PCFGs. Cohen *et al.* [2012] present a spectral algorithm for L-PCFG estimation, but the naive transformation of the L-PCFG model and its spectral algorithm to R-HMMs is awkward and opaque. We therefore work through the non-trivial derivation the spectral algorithm for R-HMMs.

We note that much of the prior work on spectral algorithms for discrete structures in NLP has shown limited experimental success for this family of algorithms (see, for example, Luque *et al.* [2012]). Our experiments demonstrate empirical success for the R-HMM spectral algorithm. The spectral algorithm performs competitively with EM on a phoneme recognition task, and is more stable with respect to the number of hidden states.

Cohen *et al.* [2013] present experiments with a parsing algorithm and also demonstrate it is competitive with EM. Our set of experiments comes as an additional piece of evidence that spectral algorithms can function as a viable, efficient and more principled alternative to the EM algorithm.

8.2 Related Work

Recently, there has been a surge of interest in spectral methods for learning HMMs [Hsu *et al.*, 2008; Foster *et al.*, 2012; Jaeger, 2000; Siddiqi *et al.*, 2010; Song *et al.*, 2010]. Like these previous works, our method produces consistent parameter estimates; however, we estimate parameters for a supervised learning task. Balle *et al.* [2011] also consider a supervised problem, but our model is quite different since we estimate a joint distribution $p(a_1 \dots a_N, x_1 \dots x_N, h_1 \dots h_N)$ as opposed to a conditional distribution and use feature functions over both the labels and observations of the training data. These feature functions also go beyond those previously employed in other spectral work [Siddiqi *et al.*, 2010; Song *et al.*, 2010]. Experiments show that features of this type are crucial for performance.

Spectral learning has been applied to related models beyond HMMs including: head

automata for dependency parsing [Luque *et al.*, 2012], tree-structured directed Bayes nets [Parikh *et al.*, 2011], finite-state transducers [Balle *et al.*, 2011], and mixture models [Anandkumar *et al.*, 2012a,c].

Of special interest is Cohen *et al.* [2012], who describe a derivation for a spectral algorithm for L-PCFGs. This derivation is the main driving force behind the derivation of our R-HMM spectral algorithm. For work on L-PCFGs estimated with EM, see Petrov *et al.* [2006], Matsuzaki *et al.* [2005], and Pereira and Schabes [1992]. Petrov *et al.* [2007] proposes a split-merge EM procedure for phoneme recognition analogous to that used in latent-variable parsing.

8.3 The R-HMM Model

We decribe in this section the notation used throughout and the formal details of R-HMMs.

8.3.1 Definition of an R-HMM

An R-HMM is a 7-tuple $\langle l, m, n, \pi, o, t, f \rangle$ for integers $l, m, n \geq 1$ and functions π, o, t, f where

- $[l]$ is a set of labels.
- $[m]$ is a set of hidden states.
- $[n]$ is a set of observations.
- $\pi(a, h)$ is the probability of generating $a \in [l]$ and $h \in [m]$ in the first position in the labeled sequence.
- $o(x|a, h)$ is the probability of generating $x \in [n]$, given $a \in [l]$ and $h \in [m]$.
- $t(b, h'|a, h)$ is the probability of generating $b \in [l]$ and $h' \in [m]$, given $a \in [l]$ and $h \in [m]$.
- $f(*|a, h)$ is the probability of generating the stop symbol $*$, given $a \in [l]$ and $h \in [m]$.

See Figure 8.1(b) for an illustration. At any time step of a sequence, a label a is associated with a hidden state h. By convention, the end of an R-HMM sequence is signaled by the symbol $*$.

For the subsequent illustration, let N be the length of the sequence we consider. A *full sequence* consists of labels $a_1 \dots a_N$, observations $x_1 \dots x_N$, and hidden states $h_1 \dots h_N$. The model assumes

$$\begin{aligned} &p(a_1 \dots a_N, x_1 \dots x_N, h_1 \dots h_N) \\ &\quad = \pi(a_1, h_1) \times \prod_{i=1}^{N} o(x_i | a_i, h_i) \times \prod_{i=1}^{N-1} t(a_{i+1}, h_{i+1} | a_i, h_i) \times f(* | a_N, h_N) \end{aligned}$$

A *skeletal sequence* consists of labels $a_1 \dots a_N$ and observations $x_1 \dots x_N$ without hidden states. Under the model, it has probability

$$p(a_1 \dots a_N, x_1 \dots x_N) = \sum_{h_1 \dots h_N} p(a_1 \dots a_N, x_1 \dots x_N, h_1 \dots h_N)$$

An equivalent definition of an R-HMM is given by organizing the parameters in matrix form. Specifically, an R-HMM has parameters $\left\langle \pi^a, o_x^a, T^{b|a}, f^a \right\rangle$ where $\pi^a \in \mathbb{R}^m$ is a column vector, $o_x^a \in \mathbb{R}^{1 \times m}$ is a row vector, $T^{b|a} \in \mathbb{R}^{m \times m}$ is a matrix, and $f^a \in \mathbb{R}^{1 \times m}$ is a row vector, defined for all $a, b \in [l]$ and $x \in [n]$. Their entries are set to

- $[\pi^a]_h = \pi(a, h)$ for $h \in [m]$
- $[o_x^a]_h = o(x | a, h)$ for $h \in [m]$
- $[T^{b|a}]_{h',h} = t(b, h' | a, h)$ for $h, h' \in [m]$
- $[f^a]_h = f(* | a, h)$ for $h \in [m]$

8.4 The Forward-Backward Algorithm

Given an observation sequence $x_1 \dots x_N$, we want to infer the associated sequence of labels under an R-HMM. This can be done by computing the *marginals* of $x_1 \dots x_N$

$$\mu(a, i) = \sum_{a_1 \dots a_N \colon a_i = a} p(a_1 \dots a_N, x_1 \dots x_N)$$

Input: a sequence of observations $x_1 \dots x_N$; operators $\langle C^{b|a}, C^{*|a}, c_a^1, c_x^a \rangle$

Output: $\mu(a, i)$ for all $a \in [l]$ and $i \in [N]$

[Forward case]

- $\alpha_a^1 \leftarrow c_a^1$ for all $a \in [l]$.
- For $i = 1 \dots N-1$

$$\alpha_b^{i+1} \leftarrow \sum_{a \in [l]} C^{b|a}(c_{x_i}^a) \times \alpha_a^i \text{ for all } b \in [l]$$

[Backward case]

- $\beta_a^N \leftarrow C^{*|a}(c_{x_N}^a)$ for all $a \in [l]$
- For $i = N-1 \dots 1$

$$\beta_a^i \leftarrow \sum_{b \in [l]} \beta_b^{i+1} \times C^{b|a}(c_{x_i}^a) \text{ for all } a \in [l]$$

[Marginals]

- $\mu(a, i) \leftarrow \beta_a^i \times \alpha_a^i$ for all $a \in [l], i \in [N]$

Figure 8.2: The forward-backward algorithm (in matrix form) for an R-HMM.

for all labels $a \in [l]$ and positions $i \in [N]$. Then the most likely label at each position i is given by

$$a_i^* = \arg\max_{a \in [l]} \mu(a, i)$$

The marginals can be computed using a tensor variant of the forward-backward algorithm, shown in Figure 8.2. The algorithm takes additional quantities $\langle C^{b|a}, C^{*|a}, c_a^1, c_x^a \rangle$ called the *operators*:

- Tensors $C^{b|a} \in \mathbb{R}^{m \times m \times m}$ for $a, b \in [l]$
- Tensors $C^{*|a} \in \mathbb{R}^{1 \times m \times m}$ for $a \in [l]$
- Column vectors $c_a^1 \in \mathbb{R}^m$ for $a \in [l]$
- Row vectors $c_x^a \in \mathbb{R}^{1 \times m}$ for $a \in [l]$ and $x \in [n]$

The following proposition states that these operators can be defined in terms of the R-HMM parameters to guarantee the correctness of the algorithm.

Proposition 8.4.1. *Given an R-HMM with parameters* $\langle \pi^a, o_x^a, T^{b|a}, f^a \rangle$*, for any vector* $v \in \mathbb{R}^m$ *define the operators:*

$$C^{b|a}(v) = T^{b|a} \operatorname{diag}(v) \qquad c_a^1 = \pi^a$$
$$C^{*|a}(v) = f^a \operatorname{diag}(v) \qquad c_x^a = o_x^a$$

Then the algorithm in Figure 8.2 correctly computes marginals $\mu(a, i)$ *under the R-HMM.*

Proof. At any time step $i \in [N]$ in the algorithm in Figure 8.2, for all label $a \in [l]$ we have a column vector $\alpha_a^i \in \mathbb{R}^m$ and a row vector $\beta_a^i \in \mathbb{R}^{1 \times m}$. The value of these vectors at each index $h \in [m]$ can be verified as

$$[\alpha_a^i]_h = \sum_{\substack{a_1 \dots a_i, h_1 \dots h_i: \\ a_i = a, h_i = h}} p(a_1 \dots a_i, x_1 \dots x_{i-1}, h_1 \dots h_i)$$
$$[\beta_a^i]_h = \sum_{\substack{a_i \dots a_N, h_i \dots h_N: \\ a_i = a, h_i = h}} p(a_{i+1} \dots a_N, x_i \dots x_N, h_{i+1} \dots h_N | a_i, h_i)$$

Thus $\beta_a^i \alpha_a^i$ is a scalar equal to

$$\sum_{\substack{a_1 \dots a_N, h_1 \dots h_N: \\ a_i = a}} p(a_1 \dots a_N, x_1 \dots x_N, h_1 \dots h_N)$$

which is the value of the marginal $\mu(a, i)$. □

Note that the running time of the algorithm as written is $O(l^2 m^3 N)$.[1]

Proposition 8.4.1 can be generalized to the following theorem. This theorem implies that the operators can be linearly transformed by some invertible matrices as long as the transformation leaves the embedded R-HMM parameters intact. This observation is central to the derivation of the spectral algorithm which estimates the linearly transformed operators but not the actual R-HMM parameters.

[1] We can reduce the complexity to $O(l^2 m^2 N)$ by pre-computing the matrices $C^{b|a}(c_x^a)$ for all $a, b \in [l]$ and $x \in [n]$ after parameter estimation.

Theorem 8.4.1. *Given an R-HMM with parameters* $\langle \pi^a, o_x^a, T^{b|a}, f^a \rangle$*, assume that for each* $a \in [l]$ *we have invertible* $m \times m$ *matrices* G^a *and* H^a*. For any row vector* $v \in \mathbb{R}^{1 \times m}$ *define the operators:*

$$C^{b|a}(v) = G^b T^{b|a} \operatorname{diag}(vH^a)(G^a)^{-1} \qquad c_a^1 = G^a \pi^a$$
$$C^{*|a}(v) = f^a \operatorname{diag}(vH^a)(G^a)^{-1} \qquad c_x^a = o_x^a (H^a)^{-1}$$

Then the algorithm in Figure 8.2 correctly computes marginals $\mu(a, i)$ *under the R-HMM.*

The proof is similar to that of Cohen *et al.* [2012].

8.5 Spectral Estimation of R-HMMs

In this section, we derive a consistent estimator for the operators $\langle C^{b|a}, C^{*|a}, c_a^1, c_x^a \rangle$ in Theorem 8.4.1 through the use of singular-value decomposition (SVD) followed by the method of moments.

Section 8.5.1 describes the decomposition of the R-HMM model into random variables which are used in the final algorithm. Section 8.5.2 can be skimmed through on the first reading, especially if the reader is familiar with other spectral algorithms. It includes a detailed account of the derivation of the R-HMM algorithm.

For a first reading, note that an R-HMM sequence can be seen as a right-branching L-PCFG tree. Thus, in principle, one can convert a sequence into a tree and run the inside-outside algorithm of Cohen *et al.* [2012] to learn the parameters of an R-HMM. However, projecting this transformation into the spectral algorithm for L-PCFGs is cumbersome and unintuitive. This is analogous to the case of the Baum-Welch algorithm for HMMs [Rabiner, 1989], which is a special case of the inside-outside algorithm for PCFGs [Lari and Young, 1990].

8.5.1 Random Variables

We first introduce the random variables underlying the approach then describe the operators based on these random variables. From $p(a_1 \dots a_N, x_1 \dots x_N, h_1 \dots h_N)$, we draw an R-HMM sequence $(a_1 \dots a_N, x_1 \dots x_N, h_1 \dots h_N)$ and choose a time step i uniformly at random

from $[N]$. The random variables are then defined as

$$
\begin{aligned}
X &= x_i \\
A_1 &= a_i \text{ and } A_2 = a_{i+1} && \text{(if } i = N,\ A_2 = *) \\
H_1 &= h_i \text{ and } H_2 = h_{i+1} \\
F_1 &= (a_i \dots a_N, x_i \dots x_N) && \text{(future)} \\
F_2 &= (a_{i+1} \dots a_N, x_{i+1} \dots x_N) && \text{(skip-future)} \\
P &= (a_1 \dots a_i, x_1 \dots x_{i-1}) && \text{(past)} \\
R &= (a_i, x_i) && \text{(present)} \\
D &= (a_1 \dots a_N, x_1 \dots x_{i-1}, x_{i+1} \dots x_N) && \text{(destiny)} \\
B &= [[i = 1]]
\end{aligned}
$$

Figure 8.3 shows the relationship between the random variables. They are defined in such a way that the future is independent of the past and the present is independent of the destiny conditioning on the current node's label and hidden state.

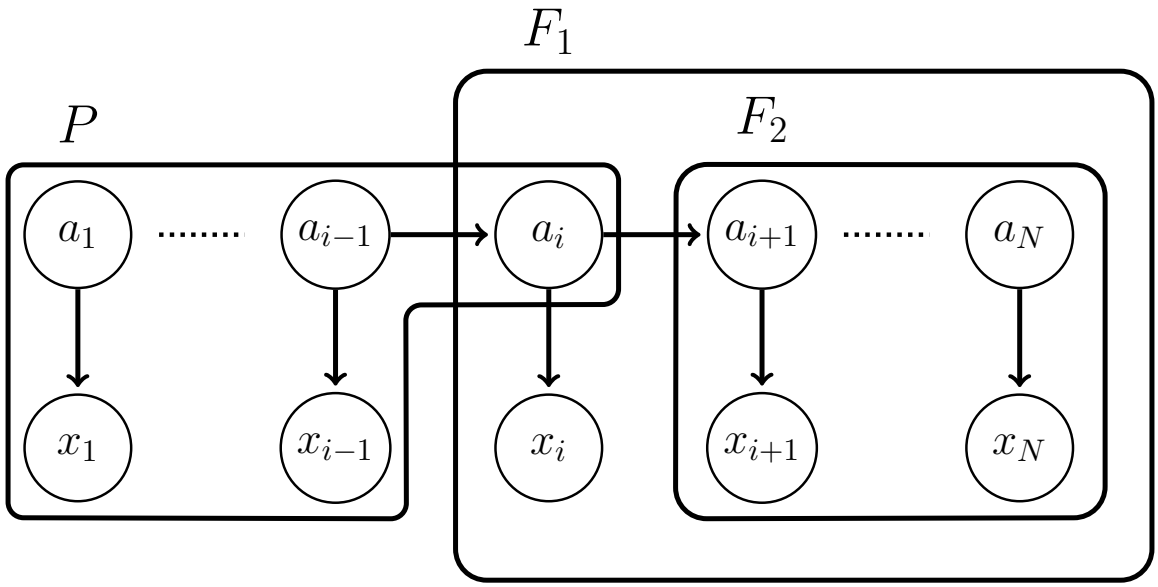
Next, we assume a set of feature functions over the random variables.

- ϕ maps F_1, F_2 respectively to $\phi(F_1), \phi(F_2) \in \mathbb{R}^{d_1}$.
- ψ maps P to $\psi(P) \in \mathbb{R}^{d_2}$.
- ξ maps R to $\xi(R) \in \mathbb{R}^{d_3}$.
- υ maps D to $\upsilon(D) \in \mathbb{R}^{d_4}$.

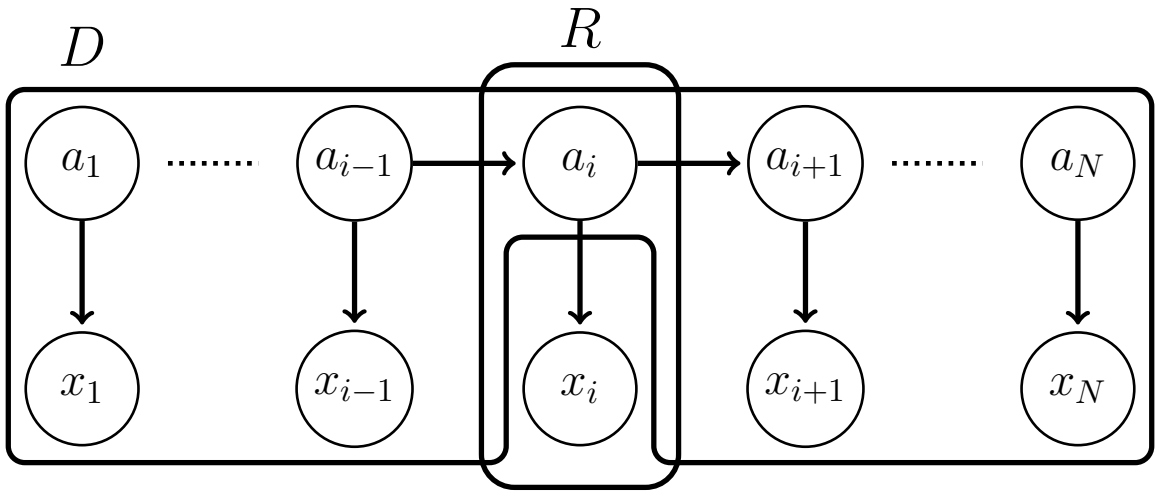
We will see that the feature functions should be chosen to capture the influence of the hidden states. For instance, they might track the next label, the previous observation, or important combinations of labels and observations.

Finally, we assume projection matrices

$$
\begin{aligned}
\Phi^a &\in \mathbb{R}^{m \times d_1} & \Psi^a &\in \mathbb{R}^{m \times d_2} \\
\Xi^a &\in \mathbb{R}^{m \times d_3} & \Upsilon^a &\in \mathbb{R}^{m \times d_4}
\end{aligned}
$$



(a)



(b)

Figure 8.3: Given an R-HMM sequence, we define random variables over observed quantities so that conditioning on the current node, (a) the future F_1 is independent of the past P and (b) the present R is independent of the density D.

defined for all labels $a \in [l]$. These matrices will project the feature vectors of ϕ, ψ, ξ, and υ from (d_1, d_2, d_3, d_4)-dimensional spaces to an m-dimensional space. We refer to this

reduced dimensional representation by the following random variables:

$$\underline{F}_1 = \Phi^{A_1}\phi(F_1) \qquad \text{(projected future)}$$
$$\underline{F}_2 = \Phi^{A_2}\phi(F_2) \qquad \text{(projected skip-future defined for } i < N)$$
$$\underline{P} = \Psi^{A_1}\psi(P) \qquad \text{(projected past)}$$
$$\underline{R} = \Xi^{A_1}\xi(R) \qquad \text{(projected present)}$$
$$\underline{D} = \Upsilon^{A_1}\upsilon(D) \qquad \text{(projected destiny)}$$

Note that they are all vectors in $\mathbb{R}^m$.

8.5.2 Estimation of the Operators

Since $\underline{F}_1$, $\underline{F}_2$, $\underline{P}$, $\underline{R}$, and $\underline{D}$ do not involve hidden variables, the following quantities can be directly estimated from the training data of skeletal sequences. For this reason, they are called *observable blocks*:

$$\Sigma^a = \boldsymbol{E}[\underline{F}_1\underline{P}^\top | A_1 = a] \qquad \forall a \in [l]$$
$$\Lambda^a = \boldsymbol{E}[\underline{R}\ \underline{D}^\top | A_1 = a] \qquad \forall a \in [l]$$
$$D^{b|a} = \boldsymbol{E}[[[A_2 = b]]\underline{F}_2\underline{P}^\top\underline{R}^\top | A_1 = a] \qquad \forall a, b \in [l]$$
$$D^{*|a} = \boldsymbol{E}[[[A_2 = *]]\underline{P}\underline{R}^\top | A_1 = a] \qquad \forall a \in [l]$$
$$d^a_x = \boldsymbol{E}[[[X = x]]\underline{D}^\top | A_1 = a] \qquad \forall a \in [l], x \in [n]$$

We treat $D^{*|a}$ as a tensor of size $1 \times m \times m$. The main result of this work is that under certain conditions, matrices Σ^a and Λ^a are invertible and the operators $\langle C^{b|a}, C^{*|a}, c^1_a, c^a_x \rangle$ in Theorem 8.4.1 can be expressed in terms of these observable blocks.

$$C^{b|a}(v) = D^{b|a}(v)(\Sigma^a)^{-1} \tag{8.1}$$
$$C^{*|a}(v) = D^{*|a}(v)(\Sigma^a)^{-1} \tag{8.2}$$
$$c^a_x = d^a_x(\Lambda^a)^{-1} \tag{8.3}$$
$$c^1_a = \boldsymbol{E}[[[A_1 = a]]\underline{F}_1 | B = 1] \tag{8.4}$$

To derive this result, we use the following definition to help specify the conditions on the expectations of the feature functions.

Definition 8.5.1. *For each $a \in [l]$, define matrices $I^a \in \mathbb{R}^{d_1 \times m}$, $J^a \in \mathbb{R}^{d_2 \times m}$, $K^a \in \mathbb{R}^{d_3 \times m}, W^a \in \mathbb{R}^{d_4 \times m}$ by*

$$[I^a]_{k,h} = \boldsymbol{E}[[\phi(F_1)]_k | A_1 = a, H_1 = h]$$
$$[J^a]_{k,h} = \boldsymbol{E}[[\psi(P)]_k | A_1 = a, H_1 = h]$$
$$[K^a]_{k,h} = \boldsymbol{E}[[\xi(R)]_k | A_1 = a, H_1 = h]$$
$$[W^a]_{k,h} = \boldsymbol{E}[[\upsilon(D)]_k | A_1 = a, H_1 = h]$$

In addition, let $\Gamma^a \in \mathbb{R}^{m \times m}$ be a diagonal matrix with $[\Gamma^a]_{h,h} = P(H_1 = h | A_1 = a)$.

We now state the conditions for the correctness of Eq. (8.1-8.4). For each label $a \in [l]$, we require that

Condition 6.1 I^a, J^a, K^a, W^a have rank m.

Condition 6.2 $[\Gamma^a]_{h,h} > 0$ for all $h \in [m]$.

The conditions lead to the following proposition.

Proposition 8.5.1. *Assume Condition 8.5.2 and 8.5.2 hold. For all $a \in [l]$, define matrices*

$$\Omega_1^a = \boldsymbol{E}[\phi(F_1)\psi(P)^\top | A_1 = a] \in \mathbb{R}^{d_1 \times d_2}$$
$$\Omega_2^a = \boldsymbol{E}[\xi(R)\upsilon(D)^\top | A_1 = a] \in \mathbb{R}^{d_3 \times d_4}$$

Let $u_1^a \dots u_m^a \in \mathbb{R}^{d_1}$ and $v_1^a \dots v_m^a \in \mathbb{R}^{d_2}$ be the top m left and right singular vectors of Ω^a. Similarly, let $l_1^a \dots l_m^a \in \mathbb{R}^{d_3}$ and $r_1^a \dots r_m^a \in \mathbb{R}^{d_4}$ be the top m left and right singular vectors of Ψ^a. Define projection matrices

$$\Phi^a = [u_1^a \dots u_m^a]^\top \qquad \Psi^a = [v_1^a \dots v_m^a]^\top$$
$$\Xi^a = [l_1^a \dots l_m^a]^\top \qquad \Upsilon^a = [r_1^a \dots r_m^a]^\top$$

Then the following $m \times m$ matrices

$$G^a = \Phi^a I^a \qquad \mathcal{G}^a = \Psi^a J^a$$
$$H^a = \Xi^a K^a \qquad \mathcal{H}^a = \Upsilon^a W^a$$

are invertible.

The proof resembles that of lemma 2 of Hsu *et al.* [2012]. Finally, we state the main result that shows $\left\langle C^{b|a}, C^{*|a}, c_a^1, c_x^a \right\rangle$ in Eq. (8.1-8.4) using the projections from proposition 8.5.1 satisfy Theorem 8.4.1.

Theorem 8.5.1. *Assume conditions 6.1 and 6.2 hold. Let $\langle \Phi^a, \Psi^a, \Xi^a, \Upsilon^a \rangle$ be the projection matrices from proposition 8.5.1. Then the operators in Eq. (8.1-8.4) satisfy Theorem 8.4.1.*

Proof Sketch. It can be verified that $c_a^1 = G^a \pi^a$. For the others, under the conditional independence illustrated in Figure 8.3 we can decompose the observable blocks in terms of the R-HMM parameters and invertible matrices

$$\begin{aligned} \Sigma^a &= G^a \Gamma^a (\mathcal{G}^a)^\top & \Lambda^a &= H^a \Gamma^a (\mathcal{H}^a)^\top \\ D^{b|a}(v) &= G^b T^{b|a} \operatorname{diag}(v H^a) \Gamma^a (\mathcal{G}^a)^\top & & \\ D^{*|a}(v) &= f^a \operatorname{diag}(v H^a) \Gamma^a (\mathcal{G}^a)^\top & d_x^a &= o_x^a \Gamma^a (\mathcal{H}^a)^\top \end{aligned}$$

using techniques similar to those sketched in Cohen *et al.* [2012]. By proposition 8.5.1, Σ^a and Λ^a are invertible, and these observable blocks yield the operators that satisfy Theorem 8.4.1 when placed in Eq. (8.1-8.3). □

In summary, these results show that with the proper selection of feature functions, we can construct projection matrices $\langle \Phi^a, \Psi^a, \Xi^a, \Upsilon^a \rangle$ to obtain operators $\left\langle C^{b|a}, C^{*|a}, c_a^1, c_x^a \right\rangle$ which satisfy the conditions of Theorem 8.4.1.

8.6 The Spectral Estimation Algorithm

In this section, we give an algorithm to estimate the operators $\left\langle C^{b|a}, C^{*|a}, c_a^1, c_x^a \right\rangle$ from samples of skeletal sequences. Suppose the training set consists of M skeletal sequences $(a^{(j)}, x^{(j)})$ for $j \in [M]$. Then M samples of the random variables can be derived from this training set as follows

- At each $j \in [M]$, choose a position i_j uniformly at random from the positions in $(a^{(j)}, x^{(j)})$. Sample the random variables $(X, A_1, A_2, F_1, F_2, P, R, D, B)$ using the procedure defined in section 8.5.1.

This process yields M samples

$$(x^{(j)}, a_1^{(j)}, a_2^{(j)}, f_1^{(j)}, f_2^{(j)}, p^{(j)}, r^{(j)}, d^{(j)}, b^{(j)}) \text{ for } j \in [M]$$

Assuming $(a^{(j)}, x^{(j)})$ are i.i.d. draws from the PMF $p(a_1 \dots a_N, x_1 \dots x_N)$ over skeletal sequences under an R-HMM, the tuples obtained through this process are i.i.d. draws from the joint PMF over $(X, A_1, A_2, F_1, F_2, P, R, D, B)$.

The algorithm in Figure 8.8 shows how to derive estimates of the observable representations from these samples. It first computes the projection matrices $\langle \Phi^a, \Psi^a, \Xi^a, \Upsilon^a \rangle$ for each label $a \in [l]$ by computing empirical estimates of Ω_1^a and Ω_2^a in proposition 8.5.1, calculating their singular vectors via an SVD, and setting the projections in terms of these singular vectors. These projection matrices are then used to project (d_1, d_2, d_3, d_4)-dimensional feature vectors

$$\left(\phi(f_1^{(j)}), \phi(f_2^{(j)}), \psi(p^{(j)}), \xi(r^{(j)}), \upsilon(d^{(j)})\right)$$

down to m-dimensional vectors

$$\left(\underline{f}_1^{(j)}, \underline{f}_2^{(j)}, \underline{p}^{(j)}, \underline{r}^{(j)}, \underline{d}^{(j)}\right)$$

for all $j \in [M]$. It then computes correlation between these vectors in this lower dimensional space to estimate the observable blocks which are used to obtain the operators as in Eq. (8.1-8.4). These operators can be used in algorithm 8.2 to compute marginals.

As in other spectral methods, this estimation algorithm is consistent; that is, the marginals $\hat{\mu}(a, i)$ computed with the estimated operators approach the true marginal values given more data. For details, see Cohen *et al.* [2012] and Foster *et al.* [2012].

8.7 Experiments

We apply the spectral algorithm for learning R-HMMs to the task of phoneme recognition. The goal is to predict the correct sequence of phonemes $a_1 \dots a_N$ for a given a set of speech frames $x_1 \dots x_N$. Phoneme recognition is often modeled with a fixed-structure HMM trained with EM, which makes it a natural application for spectral training.

We train and test on the TIMIT corpus of spoken language utterances [Garofolo and others, 1988]. The label set consists of $l = 39$ English phonemes following a standard

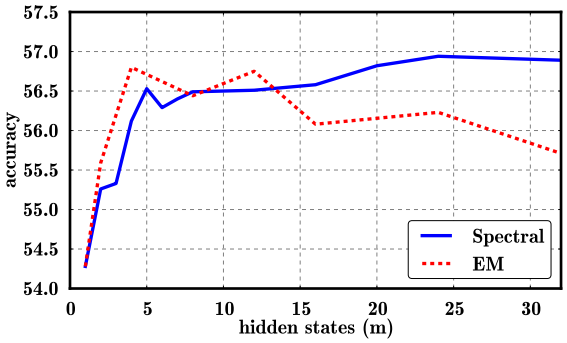


Figure 8.4: Accuracy of the spectral algorithm and EM on TIMIT development data for varying numbers of hidden states m. For EM, the highest scoring iteration is shown.

phoneme set [Lee and Hon, 1989]. For training, we use the `sx` and `si` utterances of the TIMIT training section made up of $M = 3696$ utterances. The parameter estimate is smoothed using the method of Cohen *et al.* [2013].

Each utterance consists of a speech signal aligned with phoneme labels. As preprocessing, we divide the signal into a sequence of N overlapping frames, 25ms in length with a 10ms step size. Each frame is converted to a feature representation using MFCC with its first and second derivatives for a total of 39 continuous features. To discretize the problem, we apply vector quantization using euclidean k-means to map each frame into $n = 10000$ observation classes. After preprocessing, we have 3696 skeletal sequence with $a_1 \dots a_N$ as the frame-aligned phoneme labels and $x_1 \dots x_N$ as the observation classes.

For testing, we use the `core` test portion of TIMIT, consisting of 192 utterances, and for development we use 200 additional utterances. Accuracy is measured by the percentage of frames labeled with the correct phoneme. During inference, we calculate marginals μ for each label at each position i and choose the one with the highest marginal probability, $a_i^* = \arg\max_{a \in [l]} \mu(a, i)$.

The spectral method requires defining feature functions ϕ, ψ, ξ, and υ. We use binary-valued feature vectors which we specify through features templates, for instance the template $a_i \times x_i$ corresponds to binary values for each possible label and output pair (ln binary dimensions).

$$
\begin{aligned}
\phi(F_1) \quad & a_{i+1} \times x_i,\ a_{i+1},\ x_i,\ \texttt{np}(a_i \dots a_N) \\
\psi(P) \quad & (a_{i-1}, x_{i-1}),\ a_{i-1},\ x_{i-1},\ \texttt{pp}(a_1 \dots a_i) \\
\xi(R) \quad & x_i \\
\upsilon(D) \quad & a_{i-1} \times x_{i-1},\ a_{i-1},\ x_{i-1},\ \texttt{pp}(a_1 \dots a_i), \\
& \texttt{pos}(a_1 \dots a_N)
\end{aligned}
$$

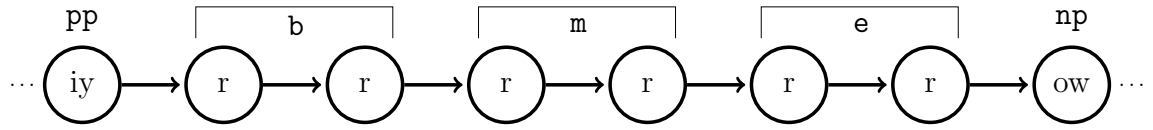


Figure 8.5: The feature templates for phoneme recognition. The simplest features look only at the current label and observation. Other features indicate the previous phoneme type used before a_i (pp), the next phoneme type used after a_i (np), and the relative position (beginning, middle, or end) of a_i within the current phoneme (pos). The figure gives a typical segment of the phoneme sequence $a_1 \dots a_N$.

Figure 8.5 gives the full set of templates. These feature functions are specially for the phoneme labeling task. We note that the HTK baseline explicitly models the position within the current phoneme as part of the HMM structure. The spectral method is able to encode similar information naturally through the feature functions.

We implement several baseline for phoneme recognition: UNIGRAM chooses the most likely label, $\arg\max_{a \in [l]} p(a|x_i)$, at each position; HMM is a standard HMM trained with maximum-likelihood estimation; EM(m) is an R-HMM with m hidden states estimated using EM; and SPECTRAL(m) is an R-HMM with m hidden states estimated with the spectral method described in this work. We also compare to HTK, a fixed-structure HMM with three segments per phoneme estimated using EM with the HTK speech toolkit. See Young *et al.* [1997] for more details on this method.

An important consideration for both EM and the spectral method is the number of hidden states m in the R-HMM. More states allow for greater label refinement, with the downside of possible overfitting and, in the case of EM, more local optima. To determine the best number of hidden states, we optimize both methods on the development set for a range of m values between 1 to 32. For EM, we run 200 training iterations on each value

Method	Accuracy
EM(4)	56.80
EM(24)	56.23
SPECTRAL(24), no `np`, `pp`, `pos`	55.45
SPECTRAL(24), no `pos`	56.56
SPECTRAL(24)	56.94

Figure 8.6: Feature ablation experiments on TIMIT development data for the best spectral model ($m = 24$) with comparisons to the best EM model ($m = 4$) and EM with $m = 24$.

of m and choose the iteration that scores best on the development set. As the spectral algorithm is non-iterative, we only need to evaluate the development set once per m value. Figure 8.4 shows the development accuracy of the two method as we adjust the value of m. EM accuracy peaks at 4 hidden states and then starts degrading, whereas the spectral method continues to improve until 24 hidden states.

Another important consideration for the spectral method is the feature functions. The analysis suggests that the best feature functions are highly informative of the underlying hidden states. To test this empirically we run spectral estimation with a reduced set of features by ablating the templates indicating adjacent phonemes and relative position. Figure 8.6 shows that removing these features does have a significant effect on development accuracy. Without either type of feature, development accuracy drops by 1.5%.

We can interpret the effect of the features in a more principled manner. Informative features yield greater singular values for the matrices Ω_1^a and Ω_2^a, and these singular values directly affect the sample complexity of the algorithm; see Cohen *et al.* [2012] for details. In sum, good feature functions lead to well-conditioned Ω_1^a and Ω_2^a, which in turn require fewer samples for convergence.

Figure 8.7 gives the final performance for the baselines and the spectral method on the TIMIT test set. For EM and the spectral method, we use best performing model from the development data, 4 hidden states for EM and 24 for the spectral method. The experiments show that R-HMM models score significantly better than a standard HMM and

Method	Accuracy
UNIGRAM	48.04
HMM	54.08
EM(4)	55.49
SPECTRAL(24)	55.82
HTK	55.70

Figure 8.7: Performance of baselines and spectral R-HMM on TIMIT test data. Number of hidden states m optimized on development data (see Figure 8.4). The improvement of the spectral method over the EM baseline is significant at the $p \leq 0.05$ level (and very close to significant at $p \leq 0.01$, with a precise value of $p \leq 0.0104$).

comparatively to the fixed-structure HMM. In training the R-HMM models, the spectral method performs competitively with EM while avoiding the problems of local optima.

8.8 Conclusion

This work derives a spectral algorithm for the task of supervised sequence labeling using an R-HMM. Unlike EM, the spectral method is guaranteed to provide a consistent estimate of the parameters of the model. In addition, the algorithm is simple to implement, requiring only an SVD of the observed counts and other standard matrix operations. We show empirically that when equipped with informative feature functions, the spectral method performs competitively with EM on the task of phoneme recognition.

Input: samples of $(X, A_1, A_2, F_1, F_2, P, R, D, B)$; feature functions ϕ, ψ, ξ, and υ; number of hidden states m

Output: estimates $\left\langle \hat{C}^{b|a}, \hat{C}^{*|a}, \hat{c}_a^1, \hat{c}_x^a \right\rangle$ of the operators used in the algorithm in Figure 8.2

[Singular Value Decomposition]

- For each label $a \in [l]$, compute empirical estimates of

$$\Omega_1^a = \boldsymbol{E}[\phi(F_1)\psi(P)^\top | A_1 = a]$$
$$\Omega_2^a = \boldsymbol{E}[\xi(R)\upsilon(D)^\top | A_1 = a]$$

  and obtain their singular vectors via an SVD. Use the top m singular vectors to construct projections $\left\langle \hat{\Phi}^a, \hat{\Psi}^a, \hat{\Xi}^a, \hat{\Upsilon}^a \right\rangle$.

[Sample Projection]

- Project (d_1, d_2, d_3, d_4)-dimensional samples of

$$(\phi(F_1), \phi(F_2), \psi(P), \xi(R), \upsilon(D))$$

  with matrices $\left\langle \hat{\Phi}^a, \hat{\Psi}^a, \hat{\Xi}^a, \hat{\Upsilon}^a \right\rangle$ to obtain m-dimensional samples of

$$(\underline{F}_1, \underline{F}_2, \underline{P}, \underline{R}, \underline{D})$$

[Method of Moments]

- For each $a, b \in [l]$ and $x \in [n]$, compute empirical estimates $\left\langle \hat{\Sigma}^a, \hat{\Lambda}^a, \hat{D}^{b|a}, \hat{D}^{*|a}, \hat{d}_x^a \right\rangle$ of the observable blocks

$$\Sigma^a = \boldsymbol{E}[\underline{F}_1 \underline{P}^\top | A_1 = a] \qquad D^{b|a} = \boldsymbol{E}[[[A_2 = b]]\underline{F}_2 \underline{P}^\top \underline{R}^\top | A_1 = a]$$
$$\Lambda^a = \boldsymbol{E}[\underline{R}\ \underline{D}^\top | A_1 = a] \qquad D^{*|a} = \boldsymbol{E}[[[A_2 = *]]\underline{P}\underline{R}^\top | A_1 = a]$$
$$d_x^a = \boldsymbol{E}[[[X = x]]\underline{D}^\top | A_1 = a]$$

  and also $\hat{c}_a^1 = \boldsymbol{E}[[[A_1 = a]]\underline{F}_1 | B = 1]$. Finally, set

$$\hat{C}^{b|a}(v) \leftarrow \hat{D}^{b|a}(v)(\hat{\Sigma}^a)^{-1} \qquad \hat{C}^{*|a}(v) \leftarrow \hat{D}^{*|a}(v)(\hat{\Sigma}^a)^{-1}$$
$$\hat{c}_x^a \leftarrow \hat{d}_x^a(\hat{\Lambda}^a)^{-1}$$

Figure 8.8: The spectral estimation algorithm for an R-HMM.

Chapter 9

Conclusions

In this thesis, we have considered how to address the computational difficulties associated with models that involve hidden or unobserved variables for natural language processing (NLP)—a family of models that underlie many state-of-the-art approaches. In particular, we have posed the question: can we develop *provable* algorithms for handling these difficulties?

We have supplied a positive answer by developing algorithms with *guarantees* for a variety of unsupervised and semi-supervised tasks that can be solved with latent-variable models. Our main weapon is *spectral methods*: techniques that use matrix or tensor factorization such as singular value decomposition (SVD). Our work builds on the recent advance in spectral methods, notably the SVD-based algorithm for learning hidden Markov models (HMMs) [Hsu *et al.*, 2008] and the non-negative matrix factorization (NMF) algorithm for learning topic models [Arora *et al.*, 2012a], to attack the following language processing problems:

1. **Inducing hierarchical word clusters (Chapter 5).** We have derived the first provably correct algorithm for clustering under the class-based language model proposed by Brown *et al.* [1992]. Unlike the original greedy algorithm, our algorithm is guaranteed to recover the true clusters given sufficiently many samples (Theorem 5.4.4). It is also much more scalable in practice—up to an order of magnitude (Table 5.3).

2. **Inducing word embeddings (Chapter 6).** We have derived a spectral algorithm

for inducing word embeddings with an SVD on a simple co-occurrence count matrix. We give a novel model-based interpretation of the previous approaches based on canonical correlation anaylsis (CCA) by Dhillon *et al.* [2011a; 2012]. The algorithm is a consistent estimator of the underlying model (Theorem 6.3.1), provides a unified view of a number of spectral methods in the literature, and yields embeddings that achieve state-of-the-art results on various lexical and semi-supervised tasks.

3. **Unsupervised part-of-speech (POS) tagging (Chapter 7).** We have derived a spectral algorithm for learning HMMs with an "anchor" assumption, which is particularly appropriate for unsupervised POS tagging. We have extended the NMF method of Arora *et al.* [2012a] to develop a consistent estimator of anchor HMMs (Theorem 7.3.1). In experiments, the algorithm is competitive with strong baselines on the universal POS tagset; moreover, it produces an interpretable model in which hidden states are automatically lexicalized.

4. **Phoneme recognition (Chapter 8).** We have derived a spectral algorithm for learning "refinement" HMMs (R-HMMs): HMMs with additional unobserved variables further refining hidden states. The algorithm extends the observable operator learning method of Hsu *et al.* [2008] and is guaranteed to find the true parameters of R-HMMs under mild conditions (Theorem 8.5.1). In experiments, the spectral algorithm is competitive with the expectation-maximization (EM) algorithm in accuracy on phoneme recognition.

These results support our earlier claims on the advantages of the presented spectral methods over existing methods. Namely, (i) our algorithms provide a new theoretical framework that is amenable to rigorous analysis, and (ii) they are also empirically effective, yielding results that are competitive with the state-of-the-art on many tasks.

We now conclude the thesis with a critical view of spectral methods and a discussion of future directions.

9.1 Limitations of the Existing Spectral Framework

(This discussion is confined to the existing spectral framework and not meant to assess the intrinsic soundness of the approach. Many of the issues may be resolved in the future.)

Spectral methods are sometimes accused of being a rigid approach, that is, an approach difficult to generalize to arbitrary problems of interest. In practice, it is desirable to have an approach in which a given problem can be modeled as freely as possible. This is because modeling is an art that requires much empirical experimentation and domain-specific expertise; there is usually no "true" model for the problem known in advance.

However, in spectral methods, a large portion of work is in *simplifying* the problem into a form that is amenable to the framework. From a variational perspective, this involves mandating the objective to take a particular form such as trace or Rayleigh quotient maximization with orthogonality constraints (e.g., see (4.22) for spectral clustering or Ando and Zhang [2005]) or unconstrained squared error minimization shown in (4.2). From a modeling perspective, this involves restricting the class of models under consideration. For instance, the methods of Hsu *et al.* [2008] and Arora *et al.* [2012a] consider simple probabilistic models such as HMMs and topic models. While their methods are certainly extendable to other related models (as done in Chapter 7 and 8 of the thesis), it is not clear how they can be used for much more general model families.

To be clear, there is ongoing progress in further generalizing the spectral framework. The tensor decomposition method of Anandkumar *et al.* [2014] can be used to learn a wide variety of latent-variable models; the algorithm of Chaganty and Liang [2014] can be used to estimate a broader class of graphical models; another tensor-based method of Janzamin *et al.* [2015] can be used to train a certain class of neural networks. But these methods are still relatively limited in flexibility, for instance compared to backpropagation training of neural networks [Rumelhart *et al.*, 1988]. In the latter approach, a model can be an arbitrary composition of linear and nonlinear functions, and the resulting objective is approximately optimized by gradient-based search.

There are many benefits in the restricted approach adopted in spectral methods. The restrictions are often designed to explicitly take advantage of our prior knowledge of the problem: for example, in Chapter 7, we exploit the tendency of a POS tag to have at least

one tag-revealing word associated with it through a restriction on the model structure. This step often results in a new understanding of the problem which can lead to other interesting algorithms. Also, framing the problem as decomposition makes it possible to leverage many techniques in linear algebra such as SVD, which is one of the few techniques known to solve a non-convex problem (e.g., see Section 4.2 on low-rank matrix approximation). Finally, the approach is naturally accomodating to theoretical analysis: with tools from mature mathematical branches such as matrix perturbation theory.

But, under the current framework, it is not obvious how to apply spectral methods to more general optimization objectives or more expressive model families that may be suitable for many empirical tasks. This can be seen as an undesirable aspect, particularly in an applied field such as NLP.

9.2 Future Work

There are several directions for future work.

9.2.1 Flexible Framework for Spectral Optimization

As discussed in the previous section, a limitation of the current spectral framework is that it is often nontrivial to extend a spectral method to much more general settings. A way to address this issue is to develop a more flexible framework for spectral optimization that can be readily applied to many problems of interest in NLP and machine learning without needing the problems to be significantly simplified.

One possibility is to develop a general recipe for breaking down the objective into simpler subproblems (e.g., by alternating minimization) that have a spectral solution or approximation. Task-specific versions of such a recipe have been derived. For instance, Ando and Zhang [2005] optimize a non-convex loss function by alternating between a convex objective and an objective whose solution is given by SVD. Janzamin *et al.* [2015] learn the parameters of the first layer of a neural network by tensor decomposition and use them to find the parameters of subsequent layers.

Another possibility is to consider manifold optimization [Absil *et al.*, 2009], which refers

to general optimization over a space of structured matrices called matrix manifold. For instance, finding the dominant eigenvalues and eigenvectors is a particular maximization problem over the (compact) Stiefel manifold.

Increased flexibility may come at a cost, for instance in theoretical guarantees (e.g., Ando and Zhang [2005] do not have guarantees of global optimality). But even a heuristic optimization framework in which it is easy to leverage powerful decomposition tools in linear algebra may be interesting in itself and effective in practice.

9.2.2 Online/Randomized Spectral Methods

In recent years, the amount of digitally available data has skyrocketed, especially unlabeled data. Accordingly, it has become increasingly important to develop algorithms that can scale to arbitrarily large inputs. Online or randomized algorithms that can be distributed across multiple machines are such algorithms.

Conventional spectral methods are batch algorithms and require all input data to be stored in memory. While they are already quite scalable (as demonstrated in Chapter 6 in which we perform SVD on matrices with billions of nonzeros), a fruitful research direction is to make online or randomized variants of these methods that can handle any size of data, preferably with strong guarantees. For example, Halko *et al.* [2011] develop a randomized, distributable algorithm for computing an SVD of a large matrix. Ma *et al.* [2015] develop an online algorithm for computing CCA, which can be parallelized with schemes such as the HOGWILD! algorithm of Recht *et al.* [2011]. Liberty [2013] develops a streaming, distributable algorithm for approximating a matrix with fixed memory (where the approximation has the property of preserving the covariance structure of the original matrix).

9.2.3 Spectral Methods for Other NLP Tasks

In the thesis, we have focused on developing spectral approaches for two specific tasks in NLP: inducing lexical representations and estimating parameters of latent-variable models for certain NLP problems. However, spectral methods are potentially appropriate for a much wider class of NLP tasks that can benefit from either the theoretically sound frame-

work or the powerful optimization tools that spectral methods can offer; many of them are left unexplored in the thesis. To give a few examples, generalized CCA for finding correlation between multiple views can have a natural application in multi-lingual tasks (e.g., see Rastogi *et al.* [2015]). Spectral methods may be derived for learning other latent-variable models used in NLP such as the IBM translation model [Brown *et al.*, 1993] or the decipherment model [Ravi and Knight, 2011]. It would be interesting to explore novel applications of spectral methods for other NLP tasks beside the ones investigated in the thesis.

Part IV

Bibliography

Bibliography

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

Alexander C Aitken. On least squares and linear combination of observations. *Proceedings of the Royal Society of Edinburgh*, 55:42–48, 1936.

A. Anandkumar, D. P. Foster, D. Hsu, S.M. Kakade, and Y.K. Liu. Two svds suffice: Spectral decompositions for probabilistic topic modeling and latent dirichlet allocation. *Arxiv preprint arXiv:1204.6703*, 2012.

Anima Anandkumar, Yi-kai Liu, Daniel J Hsu, Dean P Foster, and Sham M Kakade. A spectral algorithm for latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 917–925, 2012.

Animashree Anandkumar, Daniel Hsu, and Sham M Kakade. A method of moments for mixture models and hidden markov models. *arXiv preprint arXiv:1203.0683*, 2012.

Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6:1817–1853, 2005.

Francis J Anscombe. The transformation of poisson, binomial and negative-binomial data. *Biometrika*, pages 246–254, 1948.

Sanjeev Arora, Rong Ge, Yoni Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. A practical algorithm for topic modeling with provable guarantees. *arXiv preprint arXiv:1212.4777*, 2012.

Sanjeev Arora, Rong Ge, and Ankur Moitra. Learning topic models–going beyond svd. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 1–10. IEEE, 2012.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Random walks on context spaces: Towards an explanation of the mysteries of semantic word embeddings. *arXiv preprint arXiv:1502.03520*, 2015.

Raphaël Bailly, Xavier Carreras Pérez, Franco M Luque, and Ariadna Julieta Quattoni. Unsupervised spectral learning of wcfg as low-rank matrix completion. Association for Computational Linguistics, 2013.

Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, pages 671–680, 2008.

Borja Balle, Ariadna Quattoni, and Xavier Carreras. A spectral learning algorithm for finite state transducers. In *Machine Learning and Knowledge Discovery in Databases*, pages 156–171. Springer, 2011.

MSo Bartlett. The square root transformation in analysis of variance. *Supplement to the Journal of the Royal Statistical Society*, pages 68–78, 1936.

Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

Leonard E Baum, John Alonzo Eagon, et al. An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology. *Bull. Amer. Math. Soc*, 73(3):360–363, 1967.

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. Painless unsupervised learning with features. In *Human Language Technologies: The 2010*

Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 582–590. Association for Computational Linguistics, 2010.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

Eric Brill. Automatic grammar induction and parsing free text: A transformation-based approach. In *Proceedings of the workshop on Human Language Technology*, pages 237–242. Association for Computational Linguistics, 1993.

Peter F. Brown, Peter V. Desouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311, 1993.

Glenn Carroll and Eugene Charniak. *Two experiments on learning probabilistic dependency grammars from corpora*. Department of Computer Science, Univ., 1992.

Arun Tejasvi Chaganty and Percy Liang. Spectral experts for estimating mixtures of linear regressions. *arXiv preprint arXiv:1306.3729*, 2013.

Arun T Chaganty and Percy Liang. Estimating latent-variable graphical models using moments and likelihoods. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1872–1880, 2014.

Danqi Chen and Christopher D Manning. A fast and accurate dependency parser using neural networks. In *EMNLP*, pages 740–750, 2014.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Noam Chomsky. Three models for the description of language. *Information Theory, IRE Transactions on*, 2(3):113–124, 1956.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. Two decades of unsupervised pos induction: How far have we come? In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 575–584. Association for Computational Linguistics, 2010.

Kenneth Ward Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the second conference on Applied natural language processing*, pages 136–143. Association for Computational Linguistics, 1988.

Alan Kaylor Cline and Inderjit S Dhillon. Computation of the singular value decomposition. *Handbook of linear algebra*, pages 45–1, 2006.

S. B. Cohen and M. Collins. A provably correct learning algorithm for latent-variable PCFGs. In *Proceedings of ACL*, 2014.

S. B. Cohen, K. Stratos, M. Collins, D. P. Foster, and L. Ungar. Spectral learning of latent-variable PCFGs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, 2012.

S. B. Cohen, K. Stratos, M. Collins, D. P. Foster, and L. Ungar. Experiments with spectral learning of latent-variable pcfgs. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

Dipanjan Das and Slav Petrov. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 600–609. Association for Computational Linguistics, 2011.

Sanjoy Dasgupta and Philip M. Long. Performance guarantees for hierarchical clustering. *J. Comput. Syst. Sci.*, 70(4):555–569, June 2005.

Sanjoy Dasgupta. Learning mixtures of gaussians. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 634–644. IEEE, 1999.

Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Paramveer Dhillon, Dean P Foster, and Lyle H Ungar. Multi-view learning of word embeddings via CCA. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 199–207, 2011.

Paramveer S. Dhillon, Dean Foster, and Lyle Ungar. Multi-view learning of word embeddings via CCA. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, 2011.

Paramveer S. Dhillon, Jordan Rodu, Dean P. Foster, and Lyle H. Ungar. Two step cca: A new spectral method for estimating vector models of words. In *Proceedings of the International Conference on Machine learning*, 2012.

Paramveer S. Dhillon, Dean P. Foster, and Lyle H. Ungar. Eigenwords: Spectral word embeddings. *Journal of Machine Learning Research*, 16:3035–3078, 2015.

William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973.

David Donoho and Victoria Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in neural information processing systems*, page None, 2003.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Sean R Eddy and Richard Durbin. Rna sequence analysis using covariance models. *Nucleic acids research*, 22(11):2079–2088, 1994.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305, 1973.

Dean P Foster, Sham M Kakade, and Tong Zhang. Multi-view dimensionality reduction via canonical correlation analysis. *Toyota Technological Institute, Chicago, Illinois, Tech. Rep. TTI-TR-2008-4*, 2008.

D. P. Foster, J. Rodu, and L.H. Ungar. Spectral dimensionality reduction for hmms. *Arxiv preprint arXiv:1203.6130*, 2012.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

Pasi Franti, Timo Kaukoranta, D-F Shen, and K-S Chang. Fast and memory efficient implementation of the exact pnn. *Image Processing, IEEE Transactions on*, 9(5):773–777, 2000.

Stephen H. Friedberg, Arnold J. Insel, and Lawrence E. Spence. *Linear Algebra*. Pearson Education, Inc., 4 edition, 2003.

Jianfeng Gao, Joshua Goodman, Jiangbo Miao, et al. The use of clustering techniques for language modeling–application to asian languages. *Computational Linguistics and Chinese Language Processing*, 6(1):27–60, 2001.

J. S. Garofolo et al. Getting started with the darpa timit cd-rom: An acoustic phonetic continuous speech database. *National Institute of Standards and Technology (NIST), Gaithersburgh, MD*, 107, 1988.

Nicolas Gillis and François Glineur. Low-rank matrix approximation with weights or missing data is np-hard. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1149–1165, 2011.

Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

Gene H Golub and Hongyuan Zha. The canonical correlations of matrix pairs and their numerical computation. In *Linear algebra for signal processing*, pages 27–49. Springer, 1995.

Stephen Guattery and Gary L Miller. On the quality of spectral separators. *SIAM Journal on Matrix Analysis and Applications*, 19(3):701–719, 1998.

Aria Haghighi and Dan Klein. Prototype-driven learning for sequence models. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 320–327. Association for Computational Linguistics, 2006.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004.

Christopher J Hillar and Lek-Heng Lim. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):45, 2013.

Ngoc-Diep Ho. *Nonnegative matrix factorization algorithms and applications*. PhD thesis, ÉCOLE POLYTECHNIQUE, 2008.

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.

Daniel Hsu, Sham M Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *arXiv preprint arXiv:0811.4413*, 2008.

Daniel Hsu, Sham M Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

Hisashi Ito, S-I Amari, and Kingo Kobayashi. Identifiability of hidden markov information sources and their minimum degrees of freedom. *Information Theory, IEEE Transactions on*, 38(2):324–333, 1992.

Herbert Jaeger. Observable operator models for discrete stochastic time series. *Neural Computation*, 12(6):1371–1398, 2000.

Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *CoRR abs/1506.08473*, 2015.

Mark Johnson. Why doesn't em find good hmm pos-taggers? In *EMNLP-CoNLL*, pages 296–305, 2007.

Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

Sham M Kakade and Dean P Foster. Multi-view regression via canonical correlation analysis. In *Learning theory*, pages 82–96. Springer, 2007.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Dan Klein and Christopher D Manning. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 478. Association for Computational Linguistics, 2004.

Reinhard Kneser and Hermann Ney. Improved clustering techniques for class-based statistical language modelling. In *Third European Conference on Speech Communication and Technology*, 1993.

Terry Koo, Xavier Carreras, and Michael Collins. Simple semi-supervised dependency parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2008.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

Volodymyr Kuleshov, Arun Tejasvi Chaganty, and Percy Liang. Tensor factorization via matrix factorization. *arXiv preprint arXiv:1501.07320*, 2015.

K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer speech & language*, 4(1):35–56, 1990.

K.F. Lee and H.W. Hon. Speaker-independent phone recognition using hidden markov models. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(11):1641–1648, 1989.

Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

Erich Leo Lehmann and George Casella. *Theory of point estimation*, volume 31. Springer Science & Business Media, 1998.

Tao Lei, Yu Xin, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. Low-rank tensors for scoring dependency structures. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1381–1391, 2014.

Omer Levy and Yoav Goldberg. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Computational Natural Language Learning*, page 171, 2014.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 2177–2185, 2014.

Omer Levy, Yoav Goldberg, Ido Dagan, and Israel Ramat-Gan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3, 2015.

Shen Li, Joao V Graça, and Ben Taskar. Wiki-ly supervised part-of-speech tagging. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1389–1398. Association for Computational Linguistics, 2012.

Percy Liang and Dan Klein. Analyzing the errors of unsupervised learning. In *ACL*, pages 879–887, 2008.

P. Liang. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology, 2005.

Edo Liberty. Simple and deterministic matrix sketching. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 581–588. ACM, 2013.

Lek-Heng Lim. Singular values and eigenvalues of tensors: a variational approach. *arXiv preprint math/0607648*, 2006.

Franco M Luque, Ariadna Quattoni, Borja Balle, and Xavier Carreras. Spectral learning for non-deterministic dependency parsing. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 409–419. Association for Computational Linguistics, 2012.

Zhuang Ma, Yichao Lu, and Dean Foster. Finding linear structure in large datasets with scalable canonical correlation analysis. *arXiv preprint arXiv:1506.08170*, 2015.

Sven Martin, Jörg Liermann, and Hermann Ney. Algorithms for bigram and trigram word clustering. *Speech communication*, 24(1):19–37, 1998.

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. Probabilistic cfg with latent annotations. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 75–82. Association for Computational Linguistics, 2005.

Ryan T McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al.

Universal dependency annotation for multilingual parsing. In *ACL (2)*, pages 92–97, 2013.

Bernard Merialdo. Tagging english text with a probabilistic model. *Computational linguistics*, 20(2):155–171, 1994.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, pages 746–751, 2013.

George A Miller and Walter G Charles. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28, 1991.

Scott Miller, Jethran Guinness, and Alex Zamanian. Name tagging with word clusters and discriminative training. In *HLT-NAACL*, volume 4, pages 337–342. Citeseer, 2004.

Leon Mirsky. Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, 11(1):50–59, 1960.

Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856, 2002.

Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. Learning to classify text from labeled and unlabeled documents. *AAAI/IAAI*, 792, 1998.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics, 2013.

Patrick Pantel and Dekang Lin. Discovering word senses from text. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–619. ACM, 2002.

A. Parikh, L. Song, and E.P. Xing. A spectral algorithm for latent tree graphical models. In *Proceedings of the 28th International Conference on Machine Learning*, 2011.

K Pearson. On lines and planes of closest fit to system of points in space. philiosophical magazine, 2, 559-572, 1901.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the Empiricial Methods in Natural Language Processing*, volume 12, 2014.

F. Pereira and Y. Schabes. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th annual meeting on Association for Computational Linguistics*, pages 128–135. Association for Computational Linguistics, 1992.

Fernando Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of english words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 183–190. Association for Computational Linguistics, 1993.

Slav Petrov and Dan Klein. Improved inference for unlexicalized parsing. In *HLT-NAACL*, volume 7, pages 404–411, 2007.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics, 2006.

Slav Petrov, Adam Pauls, and Dan Klein. Learning structured models for phone recognition. In *Proc. of EMNLP-CoNLL*, 2007.

Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*, 2011.

Slav Petrov. Products of random latent variable grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27. Association for Computational Linguistics, 2010.

Eduard Prugovečki. *Quantum mechanics in Hilbert space*, volume 41. Academic Press, 1971.

Liqun Qi. Eigenvalues of a real supersymmetric tensor. *Journal of Symbolic Computation*, 40(6):1302–1324, 2005.

L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Pushpendre Rastogi, Benjamin Van Durme, and Raman Arora. Multiview lsa: Representation learning via generalized cca. In *Proceedings of NAACL*, 2015.

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics, 2009.

Sujith Ravi and Kevin Knight. Bayesian inference for zodiac and other homophonic ciphers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 239–247. Association for Computational Linguistics, 2011.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 693–701, 2011.

Doug Rohde. SVDLIBC (available at `http://tedlab.mit.edu/~dr/SVDLIBC/`), 2007.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

Tobias Schnabel and Hinrich Schütze. Flors: Fast and simple domain adaptation for part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 2:15–26, 2014.

Jamshid Shanbehzadeh and PO Ogunbona. On the computational complexity of the lbg and pnn algorithms. *Image Processing, IEEE Transactions on*, 6(4):614–616, 1997.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

Sajid M Siddiqi, Byron Boots, and Geoffrey J Gordon. Reduced-rank hidden markov models. *arXiv preprint arXiv:0910.0902*, 2009.

S. Siddiqi, B. Boots, and G. J. Gordon. Reduced-rank hidden Markov models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS-2010)*, 2010.

Noah A Smith and Jason Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics, 2005.

Noah A Smith and Jason Eisner. Guiding unsupervised grammar induction using contrastive estimation. In *Proc. of IJCAI Workshop on Grammatical Inference Applications*, pages 73–82, 2005.

Richard Socher, Christopher D Manning, and Andrew Y Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9, 2010.

Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. Parsing With Compositional Vector Grammars. In *ACL*. 2013.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

Le Song, Byron Boots, Sajid M Siddiqi, Geoffrey J Gordon, and Alex Smola. Hilbert space embeddings of hidden markov models. 2010.

Le Song, Eric P Xing, and Ankur P Parikh. Kernel embeddings of latent tree graphical models. In *Advances in Neural Information Processing Systems*, pages 2708–2716, 2011.

Le Song, Eric P Xing, and Ankur P Parikh. A spectral algorithm for latent tree graphical models. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1065–1072, 2011.

Nathan Srebro, Tommi Jaakkola, et al. Weighted low-rank approximations. In *Proceedings of the International Conference on Machine learning*, volume 3, pages 720–727, 2003.

Robert G. D. Steel. Relation between poisson and multinomial distributions. Technical Report BU-39-M, Cornell University, 1953.

GW Stewart and Ji-Guang Sun. Matrix perturbation theory (computer science and scientific computing), 1990.

Gilbert Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press Wellesley, MA, 4 edition, 2009.

Karl Stratos, Alexander M Rush, Shay B Cohen, and Michael Collins. Spectral learning of refinement hmms. In *Proceedings of CoNLL*, 2013.

Karl Stratos, Do-kyum Kim, Michael Collins, and Daniel Hsu. A spectral algorithm for learning class-based n-gram models of natural language. *Proceedings of the Association for Uncertainty in Artificial Intelligence*, 2014.

Karl Stratos, Michael Collins, and Daniel Hsu. Model-based word embeddings from decompositions of count matrices. In *Proceedings of the 53rd Annual Meeting of the Association*

for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1282–1291, Beijing, China, July 2015. Association for Computational Linguistics.

Karl Stratos, Michael Collins, and Daniel Hsu. Unsupervised part-of-speech tagging with anchor hidden markov models. *Transactions of the Association for Computational Linguistics*, 4:245–257, 2016.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12, 2013.

Sebastiaan A Terwijn. On the learnability of hidden markov models. In *Grammatical Inference: Algorithms and Applications*, pages 261–268. Springer, 2002.

Kristina Toutanova and Mark Johnson. A bayesian lda-based model for semi-supervised part-of-speech tagging. In *Advances in Neural Information Processing Systems*, pages 1521–1528, 2007.

Joel A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, pages 1–46, 2011.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

Jakob Uszkoreit and Thorsten Brants. Distributed word clustering for large scale class-based language modeling in machine translation. In *ACL*, pages 755–762, 2008.

Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Stephen A Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 20(3):1364–1377, 2009.

Santosh Vempala and Grant Wang. A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences*, 68(4):841–860, 2004.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 836–841. Association for Computational Linguistics, 1996.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Dorothea Wagner and Frank Wagner. *Between min cut and graph bisection*. Springer, 1993.

Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

Hermann Weyl. Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). *Mathematische Annalen*, 71(4):441–479, 1912.

CF Jeff Wu. On the convergence properties of the em algorithm. *The Annals of statistics*, pages 95–103, 1983.

Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, et al. *The HTK book*, volume 2. Entropic Cambridge Research Laboratory Cambridge, 1997.

Tianyi Zhou, Jeff A Bilmes, and Carlos Guestrin. Divide-and-conquer learning by anchoring a conical hull. In *Advances in Neural Information Processing Systems*, pages 1242–1250, 2014.

Will Y Zou, Richard Socher, Daniel M Cer, and Christopher D Manning. Bilingual word embeddings for phrase-based machine translation. In *EMNLP*, pages 1393–1398, 2013.

Part V

Appendices

Appendix A

Appendix for Chapter 5

A.1 Clustering Algorithm of Brown *et al.* [1992]

We review the clustering algorithm of Brown *et al.* [1992]. We first show that maximizing the log likelihood of the data is equivalent to maximizing the mutual information of clusters under the Brown model. Then we show an efficient scheme to track changes in the mutual information.

Given a word sequence $x_1 \dots x_N \in [n]^N$ and a cluster mapping $C : [n] \mapsto [m]$, the empirical log likelihood of the sequence is given by

$$\hat{L}(C) := \sum_{x,x' \in [n]} \#(x, x') \times \log \left(\frac{\#(C(x), C(x'))}{\#(C(x))} \times \frac{\#(x')}{\#(C(x'))} \right) \tag{A.1}$$

where $\#(E)$ denotes the count of the event E. The empirical mutual information of clusters is given by

$$\hat{I}(C) := \sum_{x,x' \in [n]} \frac{\#(C(x), C(x'))}{N} \times \log \left(\frac{\#(C(x), C(x')) \times N}{\#(C(x)) \times \#(C(x'))} \right) \tag{A.2}$$

To see $\arg\max_{C:[n]\mapsto[m]} \hat{L}(C) = \arg\max_{C:[n]\mapsto[m]} \hat{I}(C)$, note that

$$\begin{aligned}
\hat{L}(C) &\propto \sum_{x,x'\in[n]} \frac{\#(x,x')}{N} \times \log\left(\frac{\#(C(x),C(x'))}{\#(C(x))\times\#(C(x'))} \times \#(x')\right) \\
&= \sum_{x,x'\in[n]} \frac{\#(x,x')}{N} \log \frac{\#(C(x),C(x'))\times N}{\#(C(x))\times\#(C(x'))} + \sum_{x,x'\in[n]} \frac{\#(x,x')}{N} \log \frac{\#(x')}{N} \\
&= \sum_{x,x'\in[n]} \frac{\#(C(x),C(x'))}{N} \log \frac{\#(C(x),C(x'))\times N}{\#(C(x))\times\#(C(x'))} + \sum_{x\in[n]} \frac{\#(x)}{N} \log \frac{\#(x)}{N} \\
&= \hat{I}(C) - \hat{H}
\end{aligned}$$

where $\hat{H} := -\sum_{x\in[n]} \frac{\#(x)}{N} \log \frac{\#(x)}{N}$ is the empirical entropy of words (independent of C).

In each iteration of the Brown *et al.* [1992] algorithm, there is a search over $O(k^2)$ cluster pairs to find one whose merge results in the smallest decrease in mutual information. A naive approach is to take, for each pair, $O(k^2)$ time to calculate the mutual information of the new set of clusters $\mathcal{C}'$ obtained by merging the pair:

$$\sum_{c,c'\in\mathcal{C}'} \frac{\#(c,c')}{N} \times \log\left(\frac{\#(c,c')\times N}{\#(c)\times\#(c'))}\right) \tag{A.3}$$

where N is the number of words in the corpus. Instead, the algorithm maintains a lookup table $\mathcal{L}$ where $\mathcal{L}(c,c')$ contains the loss of mutual information when clusters c and c' are merged. Clearly, the runtime of the search using this table is $O(k^2)$, rather than $O(k^4)$ of the naive approach.

By fastidiously maintaining relevant quantities, it is possible to update $\mathcal{L}$ in $O(k^2)$ time after a merge. First, we track the empirical bigram and unigram cluster probabilities:

$$\hat{p}(c,c'|\mathcal{C}) = \frac{\#(c,c')}{N} \quad \forall c,c' \in \mathcal{C} \qquad\qquad \hat{p}(c|\mathcal{C}) = \frac{\#(c)}{N} \quad \forall c \in \mathcal{C}$$

where $\mathcal{C}$ is the current set of clusters, $|\mathcal{C}| = k$. Note that $\hat{p}$ upon a merge can be easily calculated. For instance, if $\mathcal{C}' = \mathcal{C}\backslash\{\alpha,\beta\}\cup\{\alpha+\beta\}$ denotes the new set of clusters obtained by merging $\alpha,\beta \in \mathcal{C}$ as $\alpha+\beta$, then $\hat{p}(c,\alpha+\beta|\mathcal{C}') = \hat{p}(c,\alpha|\mathcal{C}) + \hat{p}(c,\beta|\mathcal{C})$. Using these probabilities, we can also track the contribution of $c,c' \in \mathcal{C}$ in mutual information:

$$\hat{q}(c,c|\mathcal{C}) = \hat{p}(c,c'|\mathcal{C}) \times \log \frac{\hat{p}(c,c'|\mathcal{C})}{\hat{p}(c|\mathcal{C})\times\hat{p}(c'|\mathcal{C})}$$

Again, $\hat{q}$ upon a merge can be easily calculated (since $\hat{p}$ is). It is convenient to track the contribution of an individual cluster $c \in \mathcal{C}$:

$$\hat{q}(c|\mathcal{C}) = \sum_{c' \in \mathcal{C}} \hat{q}(c, c'|\mathcal{C}) + \sum_{c' \in \mathcal{C}} \hat{q}(c', c|\mathcal{C}) - \hat{q}(c, c|\mathcal{C})$$

Now we define the central quantity $\mathcal{L}(\alpha, \beta|\mathcal{C})$: the loss of mutual information upon merging $\alpha, \beta \in \mathcal{C}$ to result in $\mathcal{C}' = \mathcal{C} \backslash \{\alpha, \beta\} \cup \{\alpha + \beta\}$.

$$\begin{aligned}\mathcal{L}(\alpha, \beta|\mathcal{C}) = \hat{q}(\alpha|\mathcal{C}) + \hat{q}(\beta|\mathcal{C}) - \hat{q}(\alpha, \beta|\mathcal{C}) - \hat{q}(\beta, \alpha|\mathcal{C}) \\ - \sum_{c \in \mathcal{C}'} \hat{q}(\alpha + \beta, c|\mathcal{C}') - \sum_{c \in \mathcal{C}'} \hat{q}(c, \alpha + \beta|\mathcal{C}') + \hat{q}(\alpha + \beta, \alpha + \beta|\mathcal{C}')\end{aligned} \tag{A.4}$$

This is already an improvement over using Eq. (A.3) since it takes $O(k)$ to compute instead of $O(k^2)$. But a critical observation is that if both $c \neq \alpha + \beta$ and $c' \neq \alpha + \beta$, then we can compute $\mathcal{L}(c, c'|\mathcal{C}')$ from $\mathcal{L}(c, c'|\mathcal{C})$ in constant time. Let $\mathcal{C}_1 = \mathcal{C} \backslash \{c, c'\} \cup \{c + c'\}$ and $\mathcal{C}_2 = \mathcal{C}' \backslash \{c, c'\} \cup \{c + c'\}$. Then

$$\begin{aligned}\mathcal{L}(c, c'|\mathcal{C}') = \mathcal{L}(c, c'|\mathcal{C}) - \hat{q}(c + c', \alpha|\mathcal{C}_1) - \hat{q}(\alpha, c + c'|\mathcal{C}_1) - \hat{q}(c + c', \beta|\mathcal{C}_1) - \hat{q}(\beta, c + c'|\mathcal{C}_1) \\ + \hat{q}(c + c', \alpha + \beta|\mathcal{C}_2) + \hat{q}(\alpha + \beta, c + c'|\mathcal{C}_2)\end{aligned} \tag{A.5}$$

The final algorithm is given in Figure A.1. The algorithm also limits the number of clusters to be at most $m + 1$ in each iteration, where m is a tunable parameter. The total runtime of the algorithm is $O(N + nm^2)$ where n is the number of word types.

A.2 Incorporating Richer Context

We assume a function ϕ such that for $i \in [N]$, it returns a set of positions other than i. For example, we may define $\phi(i) = \{i - 1, i + 1\}$ to look at one position to the left and to the right. Let $s = |\phi(i)|$ and enumerate the elements of $\phi(i)$ as $j_1, \ldots, j_s$. Define $B^{(j)} \in \mathbb{R}^{n \times n}, v^{(j)} \in \mathbb{R}^n$ for all $j \in \phi(i)$ as follows:

$$\begin{aligned} B^{(j)}_{x,x'} &= P(X_i = x, X_j = x') && \forall x, x' \in [n] \\ v^{(j)}_x &= P(X_j = x) && \forall x \in [n] \end{aligned}$$

The new definitions of $B \in \mathbb{R}^{n \times ns}, v \in \mathbb{R}^{ns}$ are given by $B = [B^{(j_1)}, \ldots, B^{(j_s)}]$ and $v = [(v^{(j_1)})^\top, \ldots, (v^{(j_s)})^\top]^\top$. Letting $\Omega \in \mathbb{R}^{n \times ns}$ as in Eq. (5.1), it is easy to verify Theorem 5.4.1 using similar techniques.

BrownClustering

Input: corpus of length N containing n word types, number of active clusters $m \leq n$

Output: hierarchical clustering of n word types with m leaf clusters

Variables to maintain: cluster probabilities $\hat{p}$, contribution in mutual information $\hat{q}$, loss in mutual information upon a merge $\mathcal{L}$

Algorithm:

1. Process the corpus to collect relevant statistics. Let $w^{(1)} \dots w^{(n)} \in [n]$ be word types sorted in decreasing frequency.
2. Initialize active clusters $\mathcal{C} \leftarrow \{\{w^{(1)}\}, \dots, \{w^{(m)}\}\}$ and compute $\hat{p}$ and $\hat{q}$. Compute $\mathcal{L}(c, c')$ from scratch using Eq. (A.4) for all $c, c' \in \mathcal{C}$.
3. For $i = (m+1) \dots (n+m-1)$:
   (a) If $i \leq n$: let $\mathcal{C} \leftarrow \mathcal{C} \cup \{\{w^{(i)}\}\}$, and update $\hat{p}$, $\hat{q}$, and $\mathcal{L}$.
   (b) Compute $(\alpha, \beta) \leftarrow \arg\min_{c,c' \in \mathcal{C}} \mathcal{L}(c, c'|\mathcal{C})$.
   (c) Perform the merge of α and β in $\mathcal{C}$. Update $\hat{p}$ and $\hat{q}$.
   (d) For $c, c' \in \mathcal{C}$:
      i. If $c = \alpha+\beta$ or $c' = \alpha+\beta$: recompute $\mathcal{L}(c, c'|\mathcal{C})$ from scratch using Eq. (A.4).
      ii. If $c \neq \alpha+\beta$ and $c' \neq \alpha+\beta$: update $\mathcal{L}(c, c'|\mathcal{C})$ in constant time using Eq. (A.5).
4. Prune the hierarchy to have m leaf clusters.

Figure A.1: The $O(N + nm^2)$ clustering algorithm of Brown *et al.* [1992]. The variables are explained in the main text.

A.3 Consistency of Clusters: Proof of Theorem 5.4.4

Write the rank-m SVD of Ω as $\Omega = USV^\top$, and similarly write the rank-m SVD of $\hat{\Omega}$ as $\hat{U}\hat{S}\hat{V}^\top$. Since Ω has rank m, it follows by Eckart-Young that

$$\|\hat{U}\hat{S}\hat{V}^\top - \hat{\Omega}\| \leq \|\Omega - \hat{\Omega}\|.$$

Therefore, by the triangle inequality,

$$\|\hat{U}\hat{S}\hat{V}^\top - USV^\top\| \leq 2\|\Omega - \hat{\Omega}\| = 2\varepsilon\sigma_m(\Omega).$$

This implies, via applications of Wedin's theorem and Weyl's inequality,

$$\|U_\perp^\top \hat{U}\| \leq 2\varepsilon \quad \text{and} \quad \|\hat{U}_\perp^\top U\| \leq \frac{2\varepsilon}{1-2\varepsilon} \tag{A.6}$$

where $U_\perp \in \mathbb{R}^{n\times(n-m)}$ is a matrix whose columns form an orthonormal basis for the orthogonal complement of the range of U, and $\hat{U}_\perp \in \mathbb{R}^{n\times(n-m)}$ is similarly defined (and note that $\varepsilon < 1/2$ by assumption).

Recall that by Theorem 5.4.1, there exists an orthogonal matrix $Q \in \mathbb{R}^{m\times m}$ such that $U = \sqrt{O}Q^\top$. Define $\hat{Q} := \hat{U}^\top\sqrt{O} = \hat{U}^\top UQ$, and, for all $c \in [m]$, $\hat{q}_c := \hat{Q}e_c$. The fact that $\|UQe_c\| = 1$ implies

$$\|\hat{q}_c\| = \sqrt{1 - \|\hat{U}_\perp\hat{U}_\perp^\top UQe_c\|^2} \leq 1.$$

Therefore, by Eq. (A.6),

$$1 \geq \|\hat{q}_c\| \geq \|\hat{q}_c\|^2 \geq 1 - \left(\frac{2\varepsilon}{1-2\varepsilon}\right)^2. \tag{A.7}$$

We also have, for $c \neq c'$,

$$\hat{q}_c^\top \hat{q}_{c'} \leq \|\hat{U}_\perp^\top UQe_c\|\|\hat{U}_\perp^\top UQe_{c'}\| \leq \left(\frac{2\varepsilon}{1-2\varepsilon}\right)^2, \tag{A.8}$$

where the first inequality follows by Cauchy-Schwarz, and the second inequality follows from (A.6). Therefore, by Eq. (A.7) and Eq. (A.8), we have for $c \neq c'$,

$$\|\hat{q}_c - \hat{q}_{c'}\|^2 \geq 2\left(1 - 2\left(\frac{2\varepsilon}{1-2\varepsilon}\right)^2\right). \tag{A.9}$$

Let $\bar{o}_x := O_{x,C(x)}^{1/2}$. Recall that $\sqrt{O}^\top e_x = \bar{o}_x e_{C(x)} \in \mathbb{R}^m$, so $\hat{Q}\sqrt{O}^\top e_x = \bar{o}_x\hat{q}_{C(x)}$ and $\|\hat{Q}\sqrt{O}^\top e_x\| = \bar{o}_x\|q_{C(x)}\|$. By the definition of $\hat{Q}$, we have

$$\hat{U} - \sqrt{O}\hat{Q}^\top = \hat{U} - UU^\top\hat{U} = U_\perp U_\perp^\top\hat{U}$$

This implies, for any $x \in [n]$,

$$\begin{aligned}\|\hat{U}^\top e_x - \bar{o}_x\hat{q}_{C(x)}\| &= \|(\hat{U} - \sqrt{O}\hat{Q}^\top)^\top e_x\| \\ &= \|\hat{U}^\top U_\perp U_\perp^\top e_x\| \leq 2\varepsilon\end{aligned} \tag{A.10}$$

by Eq. (A.6). Moreover, by the triangle inequality,

$$|\|\hat{U}^\top e_x\| - \bar{o}_x \|q_{C(x)}\|| \le 2\varepsilon. \tag{A.11}$$

Since $\hat{M}^\top e_x = \|\hat{U}^\top e_x\|^{-1} \hat{U}^\top e_x$, we have

$$\begin{aligned}
\|\hat{M}^\top e_x - \hat{q}_{C(x)}\| &= \left\| \frac{1}{\|\hat{U}^\top e_x\|} \hat{U}^\top e_x - \hat{q}_{C(x)} \right\| \\
&\le \frac{1}{\bar{o}_x} \|\hat{U}^\top e_x - \bar{o}_x \hat{q}_{C(x)}\| + |1 - \|\hat{q}_{C(x)}\|| + \frac{|\bar{o}_x \|\hat{q}_{C(x)}\| - \|\hat{U}^\top e_x\||}{\bar{o}_x} \\
&\le \frac{4\varepsilon}{\bar{o}_x} + \left(\frac{2\varepsilon}{1 - 2\varepsilon} \right)^2,
\end{aligned} \tag{A.12}$$

where the first inequality follow by the triangle inequality and norm homogeneity, and the second inequality uses Eq. (A.10), Eq. (A.11), and Eq. (A.7). Using Eq. (A.12), we may upper bound the distance $\|\hat{M}^\top e_x - \hat{M}^\top e_{x'}\|$ when $C(x) = C(x')$; using Eq. (A.9) and Eq. (A.12), we may lower bound the distance $\|\hat{M}^\top e_x - \hat{M}^\top e_{x''}\|$ when $C(x) \neq C(x'')$. The theorem then follows by invoking the assumption on ε.

A.4 Sample Complexity: Proof of Theorem 5.4.6

Instead of estimating B, u, and v from a single long sentence, we estimate them (via maximum likelihood) using N i.i.d. sentences, each of length 2. We make this simplification because dealing with a single long sentence requires the use of tail inequalities for fast mixing Markov chains, which is rather involved. Using N i.i.d. sentences allows us to appeal to standard tail inequalities such as Chernoff bounds because the maximum likelihood estimates of B, u, and v are simply sample averages over the N i.i.d. sentences. Since the length of each sentence is 2, we can omit the random choice of index (since it is always 1). Call these N sentences $(X_1^{(1)}, X_2^{(1)}), \ldots, (X_1^{(N)}, X_2^{(N)}) \in [n]^2$.

We use the following error decomposition for $\hat{\Omega}$ in terms of the estimation errors for $\hat{B}$,

$\hat{u}$, and $\hat{v}$:

$$\begin{aligned}\hat{\Omega} - \Omega = &(I - \operatorname{diag}(u)^{-1/2} \operatorname{diag}(\hat{u})^{1/2})(\hat{\Omega} - \Omega) \\ &+ (\hat{\Omega} - \Omega)(I - \operatorname{diag}(\hat{v})^{1/2} \operatorname{diag}(v)^{-1/2}) \\ &- (I - \operatorname{diag}(u)^{-1/2} \operatorname{diag}(\hat{u})^{1/2})(\hat{\Omega} - \Omega)(I - \operatorname{diag}(\hat{v})^{1/2} \operatorname{diag}(v)^{-1/2}) \\ &+ (I - \operatorname{diag}(u)^{-1/2} \operatorname{diag}(\hat{u})^{1/2})\Omega \\ &+ \Omega(I - \operatorname{diag}(\hat{v})^{1/2} \operatorname{diag}(v)^{-1/2}) \\ &- (I - \operatorname{diag}(u)^{-1/2} \operatorname{diag}(\hat{u})^{1/2})\Omega(I - \operatorname{diag}(\hat{v})^{1/2} \operatorname{diag}(v)^{-1/2}) \\ &+ \operatorname{diag}(u)^{-1/2}(\hat{B} - B) \operatorname{diag}(v)^{-1/2}.\end{aligned}$$

Above, I denotes the $n \times n$ identity matrix.

Provided that $\epsilon_1 := \|I - \operatorname{diag}(u)^{-1/2} \operatorname{diag}(\hat{u})^{1/2}\|$ and $\epsilon_2 := \|I - \operatorname{diag}(\hat{v})^{1/2} \operatorname{diag}(v)^{-1/2}\|$ are small enough, we have

$$\|\hat{\Omega} - \Omega\| \le \frac{\epsilon_1 + \epsilon_2 + \epsilon_1\epsilon_2}{1 - \epsilon_1 - \epsilon_2}\|\Omega\| \tag{A.13}$$

$$+ \frac{\|\operatorname{diag}(u)^{-1/2}(\hat{B} - B) \operatorname{diag}(v)^{-1/2}\|}{1 - \epsilon_1 - \epsilon_2}. \tag{A.14}$$

Observe that

$$\epsilon_1 = \max_{x \in [n]} |1 - \sqrt{\hat{u}_x / u_x}|, \qquad \epsilon_2 = \max_{x \in [n]} |1 - \sqrt{\hat{v}_x / v_x}|.$$

Using Bernstein's inequality and union bounds, it can be shown that with probability at least $1 - \delta$,

$$\epsilon_1 \le c \cdot \left(\sqrt{\frac{\log(n/\delta)}{u_{\min} N}} + \frac{\log(n/\delta)}{u_{\min} N} \right),$$

$$\epsilon_2 \le c \cdot \left(\sqrt{\frac{\log(n/\delta)}{v_{\min} N}} + \frac{\log(n/\delta)}{v_{\min} N} \right),$$

for some absolute constant $c > 0$, where $u_{\min} := \min_{x \in [n]} u_x$ and $v_{\min} := \min_{x \in [n]} v_x$. Therefore we can bound the first term in $\|\hat{\Omega} - \Omega\|$ bound (Eq. (A.13)), and the denominator in the second term (Eq. (A.14)).

It remains to bound the numerator of the second term of the $\|\hat{\Omega} - \Omega\|$ bound (Eq. (A.14)), which is the spectral norm of a sum of N i.i.d. random matrices, each with mean zero. We

focus on a single random sentence (X_1, X_2) (dropping the superscript). The contribution of this sentence to the sum defining $\operatorname{diag}(u)^{-1/2}(\hat{B} - B)\operatorname{diag}(v)^{-1/2}$ is the zero-mean random matrix

$$Y := \frac{1}{N}\left(\operatorname{diag}(u)^{-1/2} e_{X_1} e_{X_2}^\top \operatorname{diag}(v)^{-1/2} - \Omega\right).$$

We will apply the matrix Bernstein inequality [Tropp, 2011]; to do so, we must bound the following quantities:

$$\|Y\|, \qquad \|\boldsymbol{E}\, YY^\top\|, \qquad \|\boldsymbol{E}\, Y^\top Y\|.$$

To bound $\|Y\|$, we simply use

$$\|Y\| \le \frac{1}{N}\left(\frac{1}{\sqrt{u_{\min} v_{\min}}} + \|\Omega\|\right).$$

To bound $\|\boldsymbol{E}\, YY^\top\|$, observe that

$$N^2\, \boldsymbol{E}\, YY^\top = \sum_{x,x'} \frac{P(X_1 = x, X_2 = x')}{P(X_1 = x)P(X_2 = x')} e_x e_x^\top - \Omega\Omega^\top.$$

The spectral norm of the summation is

$$\begin{aligned}
&\left\| \sum_{x,x'} \frac{P(X_1 = x, X_2 = x')}{P(X_1 = x)P(X_2 = x')} e_x e_x^\top \right\| \\
&= \max_{x \in [n]} \sum_{x'} \frac{P(X_1 = x, X_2 = x')}{P(X_1 = x)P(X_2 = x')} \\
&= \max_{x \in [n]} \sum_{x'} \frac{O_{x,C(x)} \pi_{C(x)} T_{C(x'),C(x)} O_{x',C(x')}}{O_{x,C(x)} \pi_{C(x)} P(C_2 = C(x')) O_{x',C(x')}} \\
&= \max_{x \in [n]} \sum_{x'} \frac{P(C_2 = C(x') | C_1 = C(x))}{P(C_2 = C(x'))} =: n_1.
\end{aligned}$$

Therefore $\|\boldsymbol{E}\, YY^\top\| \le (n_1 + \|\Omega\|^2)/N^2$. To bound $\|\boldsymbol{E}\, Y^\top Y\|$, we use

$$N^2\, \boldsymbol{E}\, Y^\top Y = \sum_{x,x'} \frac{P(X_1 = x, X_2 = x')}{P(X_1 = x)P(X_2 = x')} e_{x'} e_{x'}^\top - \Omega^\top\Omega,$$

and bound the summation as

$$\begin{aligned}
&\left\| \sum_{x,x'} \frac{P(X_1 = x, X_2 = x')}{P(X_1 = x)P(X_2 = x')} e_{x'} e_{x'}^\top \right\| \\
&= \max_{x' \in [n]} \sum_x \frac{P(X_1 = x, X_2 = x')}{P(X_1 = x)P(X_2 = x')} \\
&= \max_{x' \in [n]} \sum_x \frac{O_{x,C(x)} \pi_{C(x)} T_{C(x'),C(x)} O_{x',C(x')}}{O_{x,C(x)} \pi_{C(x)} P(C_2 = C(x')) O_{x',C(x')}} \\
&= \max_{x' \in [n]} \sum_x \frac{P(C_2 = C(x') | C_1 = C(x))}{P(C_2 = C(x'))} =: n_2.
\end{aligned}$$

Therefore $\| \boldsymbol{E} Y^\top Y \| \le (n_2 + \|\Omega\|^2)/N^2$.

The matrix Bernstein inequality implies that with probability at least $1 - \delta$,

$$\begin{aligned}
&\| \operatorname{diag}(u)^{-1/2} (\hat{B} - B) \operatorname{diag}(v)^{-1/2} \| \\
&\le c' \cdot \left(\sqrt{\frac{\max\{n_1, n_2\} \log(n/\delta)}{N}} + \frac{\log(n/\delta)}{\sqrt{u_{\min} v_{\min}} N} \right) \\
&\qquad c' \cdot \|\Omega\| \cdot \left(\sqrt{\frac{\log(n/\delta)}{N}} + \frac{\log(n/\delta)}{N} \right)
\end{aligned}$$

for some absolute constant $c' > 0$.

We can now finally state the sample complexity bound. Let $\kappa_m(\Omega) := \sigma_1(\Omega)/\sigma_m(\Omega) = \|\Omega\|/\sigma_m(\Omega)$ be the rank-m condition number of Ω. There is an absolute constant $c'' > 0$ such that for any $\epsilon \in (0, 1)$, if

$$N \ge c'' \cdot \frac{\kappa_m(\Omega)^2 \log(n/\delta) \max\{n_1, n_2, 1/u_{\min}, 1/v_{\min}\}}{\epsilon^2},$$

then with probability at least $1 - \delta$, $\|\hat{\Omega} - \Omega\|/\sigma_m(\Omega) \le \epsilon$.

We note that n_1 and n_2 are bounded by $n/\min_{c \in [m]} \pi_c$ and $n/\min_{c \in [m]} (T\pi)_c$, respectively. However, these bounds can be considerably improved in some cases. For instance, if all transition probabilities in T and initial cluster probabilities in π are near uniform, then both n_1 and n_2 are approximately n.

Appendix B

Appendix for Chapter 6

B.1 Proof of Theorem 6.3.1

We first define some random variables. Let ρ be the number of left/right context words to consider in CCA. Let $(W_1, \ldots, W_N) \in [n]^N$ be a random sequence of words drawn from the Brown model where $N \geq 2\rho + 1$, along with the corresponding sequence of hidden states $(H_1, \ldots, H_N) \in [m]^N$. Independently, pick a position $I \in [\rho+1, N-\rho]$ uniformly at random; pick an integer $J \in [-\rho, \rho]\backslash\{0\}$ uniformly at random. Define $B \in \mathbb{R}^{n\times n}$, $u, v \in \mathbb{R}^n$, $\tilde{\pi} \in \mathbb{R}^m$, and $\tilde{T} \in \mathbb{R}^{m\times m}$ as follows:

$$\begin{aligned}
B_{w,c} &:= P(W_I = w, W_{I+J} = c) && \forall w, c \in [n] \\
u_w &:= P(W_I = w) && \forall w \in [n] \\
v_c &:= P(W_{I+J} = c) && \forall c \in [n] \\
\tilde{\pi}_h &:= P(H_I = h) && \forall h \in [m] \\
\tilde{T}_{h',h} &:= P(H_{I+J} = h' | H_I = h) && \forall h, h' \in [m]
\end{aligned}$$

First, we show that $\Omega^{\langle a \rangle}$ has a particular structure under the Brown assumption. For the choice of positive vector $s \in \mathbb{R}^m$ in the theorem, we define $s_h := (\sum_w o(w|h)^a)^{-1/2}$ for all $h \in [m]$.

Lemma B.1.1. *$\Omega^{\langle a \rangle} = A\Theta^\top$ where $\Theta \in \mathbb{R}^{n\times m}$ has rank m and $A \in \mathbb{R}^{n\times m}$ is defined as:*

$$A := \operatorname{diag}(O\tilde{\pi})^{-a/2} O^{\langle a \rangle} \operatorname{diag}(\tilde{\pi})^{a/2} \operatorname{diag}(s)$$

Proof. Let $\tilde{O} := O\tilde{T}$. It can be algebraically verified that $B = O\,\mathrm{diag}(\tilde{\pi})\tilde{O}^\top$, $u = O\tilde{\pi}$, and $v = \tilde{O}\tilde{\pi}$. By Assumption 6.3.1, each entry of $B^{\langle a\rangle}$ has the form

$$\begin{aligned} B^{\langle a\rangle}_{w,c} &= \left(\sum_{h\in[m]} O_{w,h}\times\tilde{\pi}_h\times\tilde{O}_{c,h}\right)^a \\ &= \left(O_{w,\mathcal{H}(w)}\times\tilde{\pi}_{\mathcal{H}(w)}\times\tilde{O}_{c,\mathcal{H}(w)}\right)^a \\ &= O^a_{w,\mathcal{H}(w)}\times\tilde{\pi}^a_{\mathcal{H}(w)}\times\tilde{O}^a_{c,\mathcal{H}(w)} \\ &= \sum_{h\in[m]} O^a_{w,h}\times\tilde{\pi}^a_h\times\tilde{O}^a_{c,h} \end{aligned}$$

Thus $B^{\langle a\rangle} = O^{\langle a\rangle}\,\mathrm{diag}(\tilde{\pi})^a(\tilde{O}^{\langle a\rangle})^\top$. Therefore,

$$\begin{aligned} \Omega^{\langle a\rangle} &= \left(\mathrm{diag}(u)^{-1/2}B\,\mathrm{diag}(v)^{-1/2}\right)^{\langle a\rangle} \\ &= \mathrm{diag}(u)^{-a/2}B^{\langle a\rangle}\,\mathrm{diag}(v)^{-a/2} \\ &= \mathrm{diag}(O\tilde{\pi})^{-a/2}O^{\langle a\rangle}\,\mathrm{diag}(\tilde{\pi})^{a/2}\,\mathrm{diag}(s) \\ &\quad \mathrm{diag}(s)^{-1}\,\mathrm{diag}(\tilde{\pi})^{a/2}(\tilde{O}^{\langle a\rangle})^\top\,\mathrm{diag}(\tilde{O}\tilde{\pi})^{-a/2} \end{aligned}$$

This gives the desired result. □

Next, we show that the left component of $\Omega^{\langle a\rangle}$ is in fact the emission matrix O up to (nonzero) scaling and is furthermore orthonormal.

Lemma B.1.2. *The matrix A in Lemma B.1.1 has the expression $A = O^{\langle a/2\rangle}\,\mathrm{diag}(s)$ and has orthonormal columns.*

Proof. By Assumption 6.3.1, each entry of A is simplified as follows:

$$\begin{aligned} A_{w,h} &= \frac{o(w|h)^a\times\tilde{\pi}_h^{a/2}\times s_h}{o(w|\mathcal{H}(w))^{a/2}\times\tilde{\pi}^{a/2}_{\mathcal{H}(w)}} \\ &= o(w|h)^{a/2}\times s_h \end{aligned}$$

This proves the first part of the lemma. Note that:

$$[A^\top A]_{h,h'} = \begin{cases} s_h^2\times\sum_w o(w|h)^a & \text{if } h = h' \\ 0 & \text{otherwise} \end{cases}$$

Thus our choice of s gives $A^\top A = \mathcal{I}_{m\times m}$. □

Proof of Theorem 6.3.1. With Lemma B.1.1 and B.1.2, the proof is similar to the proof of Theorem 5.4.1. □