Automatic Annotation of Spoken Language Using Out-of-Domain Resources and Domain Adaptation

Anna Margolis

A dissertation submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

University of Washington

2011

Program Authorized to Offer Degree: Department of Electrical Engineering
This is to certify that I have examined this copy of a doctoral dissertation by

Anna Margolis

and have found that it is complete and satisfactory in all respects, and that any and all revisions required by the final examining committee have been made.

Chair of the Supervisory Committee:

Mari Ostendorf

Reading Committee:

Mari Ostendorf

Maya R. Gupta

Timothy J. Hazen

Karen Livescu

Date: ___________________________
In presenting this dissertation in partial fulfillment of the requirements for the doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this dissertation is allowable only for scholarly purposes, consistent with “fair use” as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to Proquest Information and Learning, 300 North Zeeb Road, Ann Arbor, MI 48106-1346, 1-800-521-0600, to whom the author has granted “the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform.”

Signature______________________________

Date______________________________
University of Washington

Abstract

Automatic Annotation of Spoken Language Using Out-of-Domain Resources and Domain Adaptation

Anna Margolis

Chair of the Supervisory Committee:
Professor Mari Ostendorf
Electrical Engineering

Speech recognition systems produce a word sequence from an acoustic signal, but many applications require the word sequence to be additionally annotated for such things as emphasis, punctuation, or dialog acts. This annotation can be accomplished by statistical classifiers trained from hand-labeled data, but it is impractical to hand label training data for every new style and language. In this work, we investigate the use of existing out-of-domain speech corpora and textual data from the Web in order annotate speech in new target domains. We also investigate the use of domain adaptation methods that use unlabeled data from the new domain together with the labeled out-of-domain data.

In the first part, we investigate a set of domain adaptation methods via analysis, simulation, and experiments on document classification tasks. We analyze a “feature restriction” approach that uses only features found in the target domain, and we compare it with feature learning methods structural correspondence learning (SCL) (Blitzer et al. 2006) and latent semantic analysis (LSA). We show that these methods can be justified by similar assumptions. We then investigate instance weighting, analyzing its effect under regularized learning, and comparing weight estimation methods for document classification.

In the second part, we consider several spoken language annotation problems. We first investigate prosodic event detection across different speaking styles; degradation due to mismatched style is small, but no substantial improvement is achieved using out-of-the-box
adaptation methods that we investigate. Next, we consider dialog act tagging across different languages, using machine translation. We find that feature restriction and SCL both improve recall of one type of dialog act (backchannels), by utilizing correlations between domain-specific words and utterance length. Finally, we investigate the use of Web-based textual conversations for detecting questions and sentence boundaries in spoken conversations. We show that adaptation methods such as bootstrapping and SCL can use unlabeled speech data to incorporate acoustic features, and have the capacity to improve performance of the text-trained model. Our work suggests approaches for using Web text to annotate speech, without hand-annotated speech training data.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Figures</td>
<td>iii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>vii</td>
</tr>
<tr>
<td>Chapter 1: Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Chapter 2: Review of Published Work on Domain Adaptation</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Background and Notation</td>
<td>8</td>
</tr>
<tr>
<td>2.3 Instance Weighting for Covariate Shift</td>
<td>11</td>
</tr>
<tr>
<td>2.4 Feature Representation Approaches</td>
<td>24</td>
</tr>
<tr>
<td>2.5 Self-labeling Approaches</td>
<td>36</td>
</tr>
<tr>
<td>2.6 Methods Based on the Clustering/Manifold Assumption</td>
<td>45</td>
</tr>
<tr>
<td>2.7 Applications of Domain Adaptation</td>
<td>49</td>
</tr>
<tr>
<td>2.8 Conclusions</td>
<td>50</td>
</tr>
<tr>
<td>Chapter 3: Experiments with Feature Representation Methods for Domain Adaptation</td>
<td>59</td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>59</td>
</tr>
<tr>
<td>3.2 Feature Restriction</td>
<td>59</td>
</tr>
<tr>
<td>3.3 Latent Semantic Analysis and Principal Component Analysis</td>
<td>75</td>
</tr>
<tr>
<td>3.4 Structural Correspondence Learning</td>
<td>77</td>
</tr>
<tr>
<td>3.5 Experiments with Feature Representation Methods on Synthetic Datasets</td>
<td>80</td>
</tr>
<tr>
<td>3.6 Experiments with Feature Representation Methods on Real Datasets</td>
<td>88</td>
</tr>
<tr>
<td>3.7 Automatically choosing dimensions for LSA and SCL</td>
<td>98</td>
</tr>
<tr>
<td>3.8 Conclusions</td>
<td>105</td>
</tr>
<tr>
<td>Chapter 4: Experiments with Instance Weighting for Domain Adaptation</td>
<td>107</td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>107</td>
</tr>
<tr>
<td>4.2 Instance Weighting and Regularized Learning</td>
<td>107</td>
</tr>
</tbody>
</table>
4.3 Experiment with Instance Weighting on a Synthetic Dataset ............... 112
4.4 Experiments with Instance Weighting on Real Datasets .................. 117
4.5 Conclusions ......................................................... 126

Chapter 5: Cross-Genre Training for Automatic Prosody Classification ........ 129
5.1 Introduction ......................................................... 129
5.2 Background ......................................................... 130
5.3 Related Work ....................................................... 133
5.4 Methods ............................................................. 139
5.5 Cross-Corpus Training .............................................. 144
5.6 Adaptation Approaches ............................................. 148
5.7 Feature Analysis ..................................................... 159
5.8 Conclusions ......................................................... 161

Chapter 6: Domain Adaptation with Unlabeled Data for Dialog Act Tagging .... 165
6.1 Introduction .......................................................... 165
6.2 Related Work ....................................................... 166
6.3 Methods .............................................................. 167
6.4 Results ............................................................... 169
6.5 Conclusions ......................................................... 178

Chapter 7: Automatic Annotation of Spoken Conversations Using Textual Con-
vversations ............................................................. 180
7.1 Introduction .......................................................... 180
7.2 Related Work ....................................................... 180
7.3 Data ................................................................. 185
7.4 Question Detection .................................................. 187
7.5 Sentence Boundary Detection ........................................ 196
7.6 Conclusions ........................................................ 208

Chapter 8: Conclusions .................................................... 210
8.1 Summary ............................................................. 210
8.2 Contributions ....................................................... 213
8.3 Future Work ......................................................... 215
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>A classification scenario in which covariate shift mismatch occurs and source-domain training is improved using instance weighting. Right: equal value contours of ( p(y = \text{blue class}</td>
<td>x) ). Left: blue and red samples make up the training data; cyan and pink samples follow the same conditional class distribution but are drawn from a Gaussian centered at ((2, 0)). Solid line represents decision boundary of logistic regression model trained on unweighted blue and red samples; dashed line trained on weighted samples. (Note that cyan and pink samples are not used in training, as the true distribution is assumed known.)</td>
</tr>
<tr>
<td>3.1</td>
<td>( Y = \text{sign}(X_s + X_b + 0.5 + \epsilon) ) for ( X_s, X_b, X_t ) as described above. Magenta and cyan data points represent the two classes in the target domain, corresponding to red and blue in the source domain. Lower plot shows the target distribution for ( X_b, X_t ); upper plot shows the source distribution for ( X_s, X_b ) as well as the target distribution projected onto that space. The full source domain model (solid decision boundary) does better than the shared-only source domain model (dashed decision boundary).</td>
<td>67</td>
</tr>
<tr>
<td>3.2</td>
<td>( Y = \text{sign}(X_b + X_t + 0.5 + \epsilon) ). There is no difference between the full source domain model and the shared-only source domain model.</td>
<td>68</td>
</tr>
<tr>
<td>3.3</td>
<td>( Y = \text{sign}(X_s + X_t + 0.5 + \epsilon) ). Positive correspondence condition: the shared-only source domain model (dashed) does better than the full source domain model (solid).</td>
<td>69</td>
</tr>
<tr>
<td>3.4</td>
<td>( Y = \text{sign}(X_s + X_b + X_t + 0.5 + \epsilon) ). Positive correspondence condition: the shared-only source domain model (dashed) does better than the full source domain model (solid).</td>
<td>70</td>
</tr>
<tr>
<td>3.5</td>
<td>( Y = \text{sign}(X_s - X_t + \epsilon) ). Negative correspondence condition: both models do poorly on the target domain, but the full source domain model (solid) does better than the shared-only model (which has the wrong sign for ( \beta_b )).</td>
<td>71</td>
</tr>
<tr>
<td>3.6</td>
<td>Performance of SCL (left) and LSA (right) on the SCL Dataset, as a function of the feature scaling factor using different numbers of SCL or LSA dimensions. The “baseline cross-dom” is the baseline transfer result (using the original feature set); “gold-standard” is the in-domain (target-trained) result.</td>
<td>84</td>
</tr>
</tbody>
</table>
3.7 Performance of SCL, LSA, feature-restriction, baseline transfer and gold standard on the SCL Dataset as the correspondence decays between the true source and target prediction weights. SCL and LSA used $d = 200$ dimensions and scale factor 5. Shown results are averages from 5 runs, with max and min shown in the error bars. ................................. 85

3.8 Performance of SCL (left) and LSA (right) on the Generative Dataset, as a function of the feature scaling factor using different numbers of SCL or LSA dimensions. .................................................. 87

3.9 Performance of SCL, LSA, feature-restriction, baseline transfer and gold standard on the Generative Dataset as the proportion of domain-specific to shared features increases. Shown results are averages from 5 runs, with max and min shown in the error bars. .................................................. 89

3.10 Comparison of feature restriction, LSA, and SCL performance for the Sentiment task, along with baseline and gold standard. ................................. 93

3.11 Comparison of feature restriction, LSA, and SCL performance for 20News-groups tasks, along with baseline and gold standard. The settings of SCL and LSA were tuned on the “comp-vs-rec” task. ................................. 95

3.12 Performance of automatic dimension pruning for LSA and SCL on the Generative Dataset. Shown results are averages from 5 runs, with max and min shown in the error bars. .................................................. 101

3.13 Results of automatic dimension pruning for LSA and SCL on the Sentiment Classification dataset. .................................................. 103

3.14 Results of automatic dimension pruning for LSA and SCL on the 20News-groups dataset. .................................................. 104

4.1 Empirical average error on target domain for a simulated ridge regression example, as a function of $\gamma$ and $\lambda$. Black dot shows the minimum error point. 113

4.2 Estimation error, domain mismatch error, approximation error, and sum (target domain excess error, as defined in Eqn. 4.2) as a function of $\gamma$ for different values of $\lambda$. Estimation error has been estimated from the simulation; domain mismatch and approximation error are theoretical values. ................................. 116

4.3 Instance weighting performance on the contrived covariate shift task in which the source domain contains both books and electronics reviews while the target domain contains only electronics reviews. Performance shown for baseline (no weighting); weighting based on discriminative modeling method for $w(x)$ using weight $w(x_i)^\lambda$ for example $x_i$; weighting based on language modeling (LM) method for $w(x)$ using $w(x_i)^\lambda$ or $w(x_i)^{\frac{1}{\gamma}}$; and “oracle” weighting which includes only electronics examples weighted inversely to their proportion in the training set. The best value of $\lambda$ (in terms of best achieved accuracy over all $\gamma$) is shown for each method, as well as $\lambda = 1$. ................................. 121
4.4 Performance of instance weighting on the original Sentiment Classification tasks. The value of $C$ was tuned to $C = 10$ on the B $\rightarrow$ E task. For comparison with the results in Section 3.6.2, we also present weighted and baseline results using $C = 0.1$, and the SCL results (which also used $C = 0.1$).

4.5 Performance of instance weighting on the 20Newsgroups tasks. The value of $C$ was tuned to $C = 10^5$ on the “comp-vs-rec” task. For comparison with the results in Section 3.6.3, we also present weighted and baseline results using $C = 100$, and the SCL results (which also used $C = 100$).

5.1 Example of the ToBI annotations for tones and breaks from BU-RNC. Top shows F0 and energy contours (from Praat (Boersma and Weenink 2007)); second level shows tones tier; third level shows breaks tier.

5.2 Baseline classification error rates for different feature sets (“A”=acoustic, “T”=textual features); training on either BU-RNC, SWBD, or both sets concatenated; training a BoosTexter (“Boost”) or logistic regression (LR) model.

5.3 Performance of co-training and self-training (error rate on target domain test set). Note that the plots are on different scales.

5.4 Some empirical feature distributions in the training sets for each domain.

6.1 Per-class recall of weighted classifiers in column 2 of Table 6.2. Bar height represents average per-class recall; colors indicate contribution of each class: I=incomplete, S=statement, Q=question, B=backchannel. (Maximum possible bar height is 100%, each color 25%.)

7.1 ROC curves with AUC values for question detection on MRDA; comparison between systems trained on MRDA using lexical and/or prosodic features, and Wiki talk pages using lexical features. Starred (+) points occur near a false positive rate of 16.7%, which is the equal error rate point for the MRDA (lex only) system; they represent the analysis points used for Table 7.2.

7.2 ROC curves and AUC values for adaptation, baseline Wiki, and Wiki + MRDA dev on the question detection task. Starred (+) points represent the same false positive rate marked in Figure 7.1 and used for analysis in Tables 7.2 and 7.5.

7.3 ROC curves with AUC values for sentence boundary detection on MRDA; comparison between systems trained on MRDA using lexical and/or prosodic features, and Wiki talk pages using lexical features. Starred (+) points occur near a false positive rate of 11%, which is the equal error rate point for the MRDA (lex only) system; they represent the analysis points used for Table 7.7.
7.4 ROC curves and AUC values for adaptation, baseline Wiki, and Wiki + MRDA dev on the sentence boundary detection task. Starred (+) points represent the same false positive rate marked in Figure 7.3 and used for analysis in Tables 7.7 and 7.9.

7.5 AUC values on the MRDA test set for the system trained on Wiki + MRDA development data, with and without prosodic features, as a function of the amount of MRDA data added.
LIST OF TABLES

Table Number  Page

1.1  An example of a raw word string and the kinds of annotations that might be applied. (Example is from the Switchboard corpus.) 2

2.1  A summary of some notation used in this section 9

2.2  Summary of tasks used for experiments in published work on unsupervised domain adaptation. 51

2.3  Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued). 52

2.4  Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued). 53

2.5  Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued). 54

2.6  Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued). 55

2.7  Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued). 56

2.8  Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued). 57

2.9  Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued). 58

3.1  Error rates (%) for the full and shared-only models for the source domain evaluated on the target domain, for each of the cases depicted in Figure 3.1 through 3.5. 66

4.1  Statistics on the weights estimated using the different methods. 123

5.1  Characteristics of the train, development, and test sets used in this chapter.
    BU-RNC is divided into news stories; SWBD is divided into conversations between two people. The number of speakers is the number of distinct speakers in the set (in SWBD, speakers often participate in multiple conversations). 141
5.2 Corpus-level feature normalization results (classification error rate %) for the BoosTexter classifier using acoustic+textual features; “norm. all” normalizes all numeric features, “norm. acoustic” normalizes only the acoustic numeric features. Normalization generally harmed cross-corpus training performance.

5.3 Results of class proportion adjustment (classification error rate %) for the logistic regression classifier using acoustic+textual features. “Cheating” uses labels in the test set to compute the class proportions; “unsupervised” uses the EM method described in Saerens et al. (2002).

5.4 Results of instance weighting (classification error rate %) for the logistic regression classifier using acoustic+textual features. “Discriminative” uses the method based on Bickel et al. (2007), as described above; “KLIEP” uses the method from Sugiyama et al. (2008).

5.5 Domain discrimination performance (error rate) using different feature sets. Parentheses indicate which textual feature set was used (textual features differed between the accent and break tasks).

5.6 In-corpus and cross-corpus error rates of logistic regression classifiers on subsets of the test data: domain-discriminated and undiscriminated examples.

5.7 Top ranked features (in decreasing order) for training accuracy on each training set, using single-feature, single-threshold classifiers.

6.1 Proportion of utterances in each DA category in each domain’s training set.

6.2 Overall accuracy and average per-class recall on each test set, using in-domain, in-domain translated, and cross-domain training. Starred results under the accuracy column are significantly different from the corresponding cross-domain baseline under McNemar’s test ($p < 0.05$). (Significance is not calculated for the average per-class recall column.) “Majority” classifies everything as statement.

7.1 Examples of dialog in MRDA (top) and Wiki (bottom). (Note that the Wiki posts are shown here with their original punctuation and capitalization.)

7.2 Question detection rates (%) by question type for each system (L=lexical features, P=prosodic features). Detection rates are given near a false positive rate of 16.7% (starred points in Figure 7.1), which is the equal error rate point for the MRDA (L) system. Boldface gives the best result for each type.

7.3 Examples for each MRDA question category as defined in this chapter, based on Dhillon et al. (2004).

7.4 F measure for the baseline MRDA and Wiki question detection systems, for which both $C$ and the decision threshold were tuned on the MRDA development set.
7.5 Detection rates of adapted systems for each question type, near a false positive rate of 16.7% (starred points in Figure 7.2). Boldface indicates adaptation results better than baseline; italics indicate worse than baseline.

7.6 F measure for the adapted Wiki systems for question detection; the adaptation parameters and decision threshold were tuned on the MRDA development set.

7.7 Sentence boundary detection rates (%) by boundary type for each system (L=lexical features, P=prosodic features). Detection rates are given at a false positive rate near 11% (starred points in Figure 7.3), which is the equal error rate point for the MRDA (L) system. Boldface gives the best result for each type. Final row represents sentence boundaries whose type is not annotated.

7.8 F measure for the baseline MRDA and Wiki sentence boundary detection systems, for which both $C$ and the decision threshold were tuned on the MRDA development set.

7.9 Detection rates of adapted systems by sentence boundary type, near a false positive rate of 11% (starred points in Figure 7.3). All adapted results are better than baseline.

7.10 F measure for the adapted Wiki systems for sentence boundary detection; the adaptation parameters and decision threshold were tuned on the MRDA dev set.
ACKNOWLEDGMENTS

This dissertation could not have happened without the significant contributions of many others. It is difficult to thank everyone who influenced and advised me before and during graduate school, so I’m going to focus on mentioning those who contributed most directly to my work during the last 3-4 years. My biggest thanks, of course, goes to my advisor, Mari Ostendorf, whose incredible experience and commitment has made this work possible. Mari accepted me as a student when I knew nothing about speech or language processing and helped guide me to the point of being able to generate and argue for my own ideas. Mari’s knowledge of the field helped me understand the broader research setting of my work. She allowed me to explore my own ideas while offering guidance and motivation when I was discouraged. She contributed technical expertise where I had none, and her detailed reading of my work (and uncompromising standards) helped to improve it substantially.

Next, I thank Toyota Technological Institute at Chicago, which generously provided me with both a productive place to work and a stimulating research environment during the last years of my dissertation work. I am especially grateful to Karen Livescu, who served as an alternate advisor while I was there. Karen directly contributed several research ideas behind this work, and I especially benefited from her extensive knowledge of speech recognition paired with her willingness to work in new areas. While at TTI-C, I was able to work with Sham Kakade, and also briefly with John Blitzer, who both provided ideas that contributed to this work. I am grateful to T.J. Hazen who supervised me during a very useful internship at Lincoln Lab. And, I am grateful to all my committee members, Mari Ostendorf, Maya Gupta, T.J. Hazen, and Karen Livescu, who each carefully read my draft and suggested valuable changes, and Donald Percival who served as the GSR.

Several people contributed tools and data besides those that are formally cited later. SRI provided tools and data for extracting prosody features in Chapters 5 and 7, including
tools for pitch processing and forced alignment, and in some cases the force-aligned data; in particular I thank Wen Wang, Arindam Mandal, and Harry Bratt. The word-level prosody features used in those chapters were based on code contributed to by various people over many years, including Dustin Hillard, Izhak Shafran, and people at SRI. Alex Marin designed the framework for the text normalization scripts that were used in several chapters. Sasha Calhoun and Jean Carletta at University of Edinburgh provided prosody labels for Switchboard used in Chapter 5. And while on the topic of tools and data, I thank Lee Damon and the other computing staff at SSLI, who have reliably provided the necessarily environment for doing this kind of research, and have patiently responded to my many requests.

I feel extremely fortunate to have been able to work alongside several very smart and generous fellow students, and I cannot overestimate their role in helping me produce this dissertation. I would like to particularly thank Amitai Axelrod, Brian Hutchinson, Jon Malkin, Alex Marin, Julie Medero, Wei Wu, and Bin Zhang, who provided feedback on practice talks, helped me solve technical problems, and contributed to my research ideas through many useful discussions. There are many others who contributed less directly, by introducing me to technical tools, attending discussion groups, or providing their friendship; no doubt I will exclude some people, but I would like to mention Chris Bartels, Amy Dashiell, Kevin Duh, Sangyun Hahn, Dustin Hillard, Jeremy Kahn, Arild Næss, Scott Otterson, Amar Subramanya, and Mei Yang.

I thank my parents and brother for their enthusiastic interest and support, which they offered even when I was incapable of explaining to them what I was doing. Finally, I thank my husband, William Trimble, whose passion for science, and exceptional interest in explaining it, have been inspiring. This work probably would not have happened without his patient encouragement, love, and friendship.

This research was partially funded by the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA) grant number W911NF-09-0131. All statements of fact, opinion or conclusions contained herein are those of the author and
should not be construed as representing the official views or policies of IARPA, the ODNI or the U.S. Government. Partial funding was also provided by an internship at Toyota Technological Institute at Chicago; by Mari Ostendorf’s endowed professorship in System Design Methodologies; and by a gift from Microsoft.
Chapter 1

INTRODUCTION

Automatic speech recognition systems transcribe words from speech, but neglect much additional information about structure and function that would be understood by a human listener. Some aspects of this information are conveyed by prosody—the rhythm and intonation of the words—and some may be understood from context or background knowledge. Statistical models are often used to annotate the word sequence for aspects of this information, for example by inserting punctuation, labeling emphasis, or classifying utterance intent. These annotations are useful for downstream machine processing tasks, such as automatic summarization, translation, or analysis of social behavior. They are also useful for human processing, i.e., by increasing “readability.” Table 1.1 shows an example of an unannotated word string followed by different kinds of annotations that could be applied by statistical models.

The statistical models used for these tasks might be classifiers at the word or utterance level, and might use features based on both the word sequence and the audio waveform. These classifiers are often trained and tested using a hand-labeled corpus from a single setting and domain, like a collection of research meetings, talk shows, lectures, or telephone conversations. However, hand-labeling of data is time consuming and expensive, and it is generally not feasible to hand label data for every new target setting. Fortunately, many resources already exist, including both expert-labeled corpora in other domains, and natural text data containing punctuation and other kinds of “markup.” The goal of this research is to investigate the use of existing resources for building annotation systems for new domains. We investigate the performance of statistical classifiers trained with out-of-domain labeled data, and apply unsupervised domain adaptation methods, which make use of unlabeled data in the new domain in addition to the existing labeled resource.

Out-of-domain training and unsupervised domain adaptation would be particularly use-
Table 1.1: An example of a raw word string and the kinds of annotations that might be applied. (Example is from the Switchboard corpus.)

<table>
<thead>
<tr>
<th>raw sequence</th>
<th>segmentation</th>
<th>punctuation</th>
<th>dialog acts</th>
<th>prosodic events</th>
</tr>
</thead>
<tbody>
<tr>
<td>all right are you ready now okay like i said i guess it</td>
<td>all right</td>
<td>all right.</td>
<td>all right.</td>
<td>all right.</td>
</tr>
<tr>
<td>would be the work force you know as far as changes in the</td>
<td>are you ready now</td>
<td>are you ready now?</td>
<td>[statement]</td>
<td>[statement]</td>
</tr>
<tr>
<td>generations</td>
<td>okay</td>
<td>okay.</td>
<td>are you ready now?</td>
<td>are you ready now?</td>
</tr>
<tr>
<td>like i said i guess it would be the work force you</td>
<td>like i said i guess it</td>
<td>like i said i</td>
<td>like i said i guess it would be the work force you know as far as changes in the</td>
<td>like i said i guess it would be the work force you know as far as changes in the</td>
</tr>
<tr>
<td>know as far as changes in the generations.</td>
<td>would be the work force you know as far as changes in the generations.</td>
<td>guess it would be the work force you know as far as changes in the generations.</td>
<td>generations.</td>
<td></td>
</tr>
</tbody>
</table>
ful for many spoken language annotation tasks in new domains or languages for which annotated corpora are unavailable. For example, there are few widely available corpora tagged with dialogue acts in languages other than English—the Linguistic Data Consortium, a popular repository of annotated language resources, offers dialog-act-tagged corpora only in English and Spanish. Machine translation systems are available for many other languages, however, so annotation of dialog acts in such other languages may be accomplished using English training data together with machine translation. For prosodic event annotations at the word or syllable level, there was until recently only one widely-available annotated English corpus, representing professionally-read news stories. Annotated telephone conversations are now also available from the Linguistic Data Consortium, but other speaking styles, such as multi-party or face-to-face conversations, still lack any widely available labeled data. So a useful strategy for annotating data in new styles might exploit labeled corpora in different styles. For sentence boundary detection, there exists plenty of widely-available annotated English speech data, but other languages lack such data. Therefore, data from those languages in the textual domain might be useful for annotating sentence boundaries in the spoken domain.

In the last few years, there has been a large amount of work on the problem of domain adaptation (or transfer learning)—machine learning under conditions where the train and test data differ in distribution. Many new methods have been proposed and tested on a variety of tasks and datasets. Although there has been a significant amount of work with certain text processing tasks (parsing, part-of-speech tagging, document and sentiment classification), little work has been done with spoken language annotation tasks such as the ones considered here. Significant research has been conducted on adaptation for speech recognition under specific conditions (e.g., speaker adaptation for the acoustic model, topic adaptation for the language model). In contrast with text processing tasks, which use only word-based features, or speaker adaptation, which uses only acoustic features, many spoken language annotation tasks benefit from both word-based features, derived from the word sequence, and continuous acoustic features, derived from the speech waveform. Our work

\footnote{http://www.ldc.upenn.edu}
applies domain adaptation methods to settings with both acoustic and textual features. We also apply adaptation methods to several types of “domains” relevant to language processing, including read vs. spontaneous, original language vs. machine translated, and textual vs. spoken. Thus, our work provides novel settings for the application of published domain adaptation methods, and should be of relevance not just to the spoken language processing tasks considered, but also to the study of domain adaptation.

As one setting, we propose to use text-based conversations from Internet forums in order to detect questions and sentence boundaries in spoken conversations. Participants in spoken conversations use both lexical and spoken-language cues (e.g. pitch) to indicate structure and intent. Participants in text-based conversations do not use the same spoken cues, but use other extra-lexical cues not present in speech, such as punctuation. Hence, we can use the punctuation in text-based conversations to train annotation models for the spoken language domain. The spoken and text domains differ not just in the use of spoken-language cues, but also in other stylistic ways—for example, the spontaneous speech contains aborted sentences. We analyze the errors made by the text-trained model, and we apply domain adaptation methods to incorporate the spoken language cues and to adapt the model to the spoken language domain.

In addition to text-based vs. spoken conversations, we consider three other adaptation settings: (1) detection of word-level emphasis and phrase boundaries in read news stories vs. spontaneous conversations; (2) classification of speech acts in multi-party meetings vs. two-party telephone conversations; (3) classification of speech acts in English vs. Spanish conversations, where one side has been passed through a machine translation system. In each case, we analyze the domain differences and apply a selection of domain adaptation methods.

This thesis is organized as follows. Chapters 2-4 focus on domain adaptation, and Chapters 5-7 focus on the spoken language processing tasks. Related work on domain adaptation is reviewed in Chapter 2, and related work on the tasks is reviewed in Chapters 5-7.

In Chapter 2, we review four kinds of approaches to unsupervised domain adaptation: instance weighting, change-of-feature-representation, self-labeling, and cluster-based. We
include an overview of applications found in the domain adaptation literature.

In Chapter 3 we investigate empirically three domain adaptation approaches based on the change-of-feature-representation idea: feature restriction, latent semantic analysis (LSA), and Structural Correspondence Learning (SCL) (Blitzer et al. 2006). Feature restriction is a simple approach for problems such as document classification; it avoids training with features from the source domain that are not found in the target domain. By contrast, LSA and SCL learn new feature representations which combine features from both domains. We describe some conditions under which feature restriction leads to improved target domain performance, including “positive correspondence” or “corrupted feature” conditions, but also show that it can degrade target domain performance in other cases. LSA and SCL can be motivated from assumptions similar to the positive correspondence condition. LSA and SCL are more powerful; we showed in simulated experiments that SCL is more robust to certain conditions than LSA. However, both approaches have the potential to degrade performance by learning features that describe variation between domains; we therefore investigate an approach that attempts to automatically remove such features. We compare these approaches on two text classification datasets (topic classification and sentiment classification) that have been used frequently in the domain adaptation literature.

In Chapter 4, we consider instance weighting for domain adaptation. We investigate instance weighting combined with regularized linear models, and compare two approaches suggested in the literature for estimating instance weights. Experiments are conducted on the same text classification datasets used in Chapter 3. A proposed weight estimation method based on length-adapted language model weights works well in an artificial scenario, but instance weighting is generally ineffective for the real text classification tasks.

In Chapter 5, we consider methods for training a prosodic classifier using labeled training data from a different corpus and genre than the one on which the system will be deployed. Using a read news corpus and a conversational telephone corpus, two binary classification tasks are considered: word-level pitch accent and phrase boundary detection. We ask whether classifiers based on acoustic or lexical/syntactic features transfer better across genres, and we examine the feasibility of applying unsupervised domain adaptation methods to improve cross-genre performance. Importantly, we show that for both tasks, it is possible
to build a single, general classifier using both training sets that achieves most of the performance of separate classifiers trained for each genre. However, when training on only the mismatched genre, it is difficult to do better than the naive “baseline” classifier using any unsupervised adaptation method.

In Chapter 6, we investigate the classification of utterances into high-level dialog act categories using lexical features, under conditions where the train and test data differ by genre and/or language. We handle the cross-language cases with machine translation of the test utterances. We analyze and compare two feature-based approaches to using unlabeled data in adaptation: restriction to a shared feature set, and an implementation of SCL. We find that both methods lead to increased detection of backchannels in the cross-language cases by utilizing correlations between backchannel words and utterance length.

In Chapter 7, we investigate the use of punctuation in textual Internet conversations for annotating questions and sentence unit boundaries in spoken conversations. We compare the text-trained model with models trained on manually-labeled, domain-matched spoken utterances with and without acoustic-prosodic features. For question detection on pre-segmented utterances, the baseline text-trained model achieves 90% of the Area-Under-the-Curve (AUC) of the domain-matched model with prosodic features, but does especially poorly on declarative questions. For sentence boundary detection, the text-trained model achieves 85% of the AUC of the domain-matched model with prosodic features. We investigate three domain adaptation approaches to utilize unlabeled speech data with prosodic features: bootstrapping, co-training, and SCL. We find that bootstrapping and SCL are modestly effective on the question detection task, and all three methods are effective on the sentence boundary task. However, in all these approaches we use a small labeled dataset in the speech domain as a development set, in order to set parameters and select methods. We therefore compare these adaptation methods to an approach that simply uses the development data in training, without tuning parameters. We find that the latter approach achieves better results than adaptation methods, but we argue that adaptation might be more useful in cases where the development set is too small to be useful for training.

Chapter 8 contains our conclusion, where we summarize the contributions of this work and discuss future directions.
Chapter 2

REVIEW OF PUBLISHED WORK ON DOMAIN ADAPTATION

2.1 Introduction

In supervised learning, it is typically assumed that the labeled training data comes from the same distribution as the test data to which the system will be applied. In recent years, machine-learning researchers have investigated methods to handle mismatch between the training and test domains, with the goal of building a classifier using the labeled data in the old domain that will perform well on the test data in the new domain. This problem is called domain adaptation or transfer learning, and it is a common scenario in speech processing applications. Labeled training data are often produced by an expensive hand-annotation process, and may consist of only one or two annotated corpora which are used to train virtually all systems regardless of the target domain. Often little or no labeled data is available for the new domain.

In this chapter we review the statistical machine learning literature dealing with the problem of “domain adaptation” or “transfer learning”. We focus on unsupervised domain adaptation methods, which use only unlabeled data in the target domain. By contrast, supervised adaptation methods, such as that of Daume (2007), use some labeled data from the target domain. As mentioned in Chapter 1, we are interested in adaptation methods for problems such as dialog act tagging in new languages, when there is no available labeled data in the new language, but there is labeled data in English; or sentence boundary detection in languages for which there is no available labeled data from the speech domain, but there is labeled data from the text domain.

We consider four main types of approach: instance weighting for covariate shift; changes in feature representation; self-labeling methods; and cluster-based learning. Covariate shift methods re-weight training samples in the old domain to try to match the new domain, putting more weight on samples in populous regions in the new domain. Feature represen-
tation approaches try to find a new feature representation of the data, either to make the new and old distributions look similar, or to find an abstracted representation for domain-specific features. Self-labeling methods incorporate unlabeled target domain examples into the training algorithm by making an initial guess about their labels and then iteratively refining the guesses or labeling more examples. Cluster-based methods rely on the assumption that samples connected by high-density paths are likely to have the same label.

Domain adaptation is a large area of research, with related work under several frameworks (and several names). Some recent reviews include Jiang (2008) and Pan and Yang (2010). A recent book (Quiñonero-Candela et al., 2008) investigates train/test distribution mismatch in machine learning (particularly, but not exclusively, focused on covariate shift). Some of the organization here roughly follows that of Jiang (2008).

2.2 Background and Notation

In supervised classification, data consist of feature vectors \( x \in X \) and class labels \( y \in C \). Each data point \( (x, y) \) is assumed to be drawn independently and identically distributed (I.I.D.) from an unknown random distribution \( p(x, y) \). (Note that \( x \) can be a high-dimensional vector.) The goal of the learning algorithm is to use a set of training data \( \{(x_i, y_i)\}_{i=1}^N \) to create a mapping \( F : X \rightarrow C \), which can be used to generate predictions \( \hat{y} = F(x) \) for any unseen input \( x \). Here we mostly consider the parametric learning scenario, where the set of \( F \) considered is a family of functions indexed by a parameter vector \( \theta \). Ultimately the goal of selecting \( \theta \) is to minimize error of the predictions \( \hat{y} = F_\theta(x) \). This is often formalized as “risk minimization”: given some loss function \( l(x, y, \theta) \), which measures the “cost” of prediction \( \hat{y} = F_\theta(x) \) when the true label is \( y \), the goal of learning is to minimize \( E\{l(x, y, \theta)\} \) with respect to \( p(x, y) \). Since \( p(x, y) \) is not known during training, the learner tries to minimize the empirical mean loss over the training samples: \( \frac{1}{N} \sum_i l(x_i, y_i, \theta) \). We use notation such as \( p(x, y|\theta) \) and \( p(y|x; \theta) \) to describe distribution functions that depend on some parameters \( \theta \), where the distribution family is assumed and the parameters are determined during training. “Generative” classifiers model \( p(x, y|\theta) \), most commonly using maximum likelihood, in which \( l(x_i, y_i, \theta) = - \log(p(x_i, y_i|\theta)) \). “Discriminative” classifiers model \( p(y|x; \theta) \), an unnormalized version, or a decision boundary margin directly. Many
learning algorithms also add a regularization term to the objective, which penalizes complex solutions. For example, the regularization term may be a function of the norm of the parameter vector, such as $||\theta||_2$ ($L_2$ regularization) or $||\theta||_1$ ($L_1$ regularization).

<table>
<thead>
<tr>
<th>$X$</th>
<th>feature space</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$</td>
<td>set of labels</td>
</tr>
<tr>
<td>$(x_i, y_i)$</td>
<td>an example named $i$ consisting of an observed feature vector $x_i$ and a label $y_i$</td>
</tr>
<tr>
<td>$p(x, y)$</td>
<td>joint distribution of feature vectors and label values</td>
</tr>
<tr>
<td>$p(y</td>
<td>x)$</td>
</tr>
<tr>
<td>$p(x)$</td>
<td>distribution of feature vectors</td>
</tr>
<tr>
<td>$p_s(\cdot)$</td>
<td>a distribution in the source domain</td>
</tr>
<tr>
<td>$p_t(\cdot)$</td>
<td>a distribution in the target domain</td>
</tr>
<tr>
<td>$\hat{y}$</td>
<td>a hypothesized label</td>
</tr>
<tr>
<td>$\theta$</td>
<td>set of parameters to be learned during training (assuming a model family $F_\theta(x)$)</td>
</tr>
<tr>
<td>$l(x_i, y_i, \theta)$</td>
<td>training loss of the model on a particular sample $(x_i, y_i)$ as a function of $\theta$</td>
</tr>
<tr>
<td>$D_s$</td>
<td>a set of examples $(x_i, y_i)$ drawn IID from $p_s(x, y)$</td>
</tr>
<tr>
<td>$n_s$</td>
<td>the number of examples in $D_s$</td>
</tr>
<tr>
<td>$D_t$</td>
<td>a set of examples $(x_i)$ drawn IID from $p_t(x)$</td>
</tr>
<tr>
<td>$n_t$</td>
<td>the number of examples in $D_t$</td>
</tr>
<tr>
<td>$w(x_i)$</td>
<td>instance weight for example $x_i$ (see Sec. 2.3)</td>
</tr>
</tbody>
</table>

The usual assumption is that the test data will follow the same distribution as the training data, $p(x, y)$. In real life this is often not the case. Domain adaptation methods attempt to handle this mismatch explicitly, making various assumptions about the type of mismatch and the data/labels available.

In the literature, “domain adaptation”, “covariate shift”, “sample selection bias”, “transfer learning”, “multi-task learning”, “robust learning”, and “concept drift” are all terms used to describe related scenarios. One distinction to be made is whether the method aims to
optimize performance on multiple tasks or domains simultaneously (*multi-task learning*), or simply optimize performance on one domain, given training data that is from a different domain (*domain adaptation*). *Transfer learning* is often used interchangeably with domain adaptation and/or multi-task learning (or both). *Concept drift* refers to a scenario where data arrives sequentially with changing distribution, and the goal is to predict the next batch given the previously-arrived data ([Klinkenberg and Joachims 2000](#)). The goal of *robust learning* is to build a classifier that is less sensitive to certain types of changes (such as feature change or deletion) in the test data. In the case of domain adaptation, a distinction exists between *supervised domain adaptation*, which assumes some labeled data in the test domain (often called the target domain), vs. *unsupervised domain adaptation*, which assumes only labeled data from the training (source) domain and unlabeled data from the target domain. This present review looks mainly at unsupervised domain adaptation methods, rather than supervised adaptation or multi-task learning. Although they do not require labels in the new domain, these methods do take advantage of unlabeled data in the new domain. In subsequent notation, we refer to $D_s = \{(x_i, y_i)\}$ as the labeled source domain training data and $D_t = \{(x_i)\}$ as the unlabeled target domain data available at training time. We also use $p_s(\cdot)$ and $p_t(\cdot)$ to refer to distributions in the source and target domains, respectively.

Unsupervised domain adaptation can be considered as a form of semi-supervised learning, which refers to methods using both labeled and unlabeled training data. Domain adaptation is distinguished from other semi-supervised learning methods because it assumes that the labeled data and the test data are drawn from different distributions, and it uses unlabeled data that is drawn from the same distribution as the test data. (It may also include unlabeled data drawn from the source/labeled distribution). Our review focuses on literature dealing with the problem of domain adaptation rather than semi-supervised learning in general. However, several of the approaches we describe are general semi-supervised learning methods, which do not make any assumption of mismatched distributions between source and target data.
2.3 Instance Weighting for Covariate Shift

Sample selection bias is a well-studied problem in statistics which attempts to estimate properties of a distribution \( p(x, y) \) from a sample drawn according to a different distribution. It is frequently formulated in terms of a binary “selection” random variable \( \Sigma \), which determines whether or not an example drawn from \( p(x, y) \) is included or rejected in the sample; the relationship between \( p(x, y) \) and the sampling distribution depends on \( p(\Sigma = 1|x, y) \), and the sampling distribution follows \( p(x, y|\Sigma = 1) \). Covariate shift describes the sample selection bias scenario where \( p(\Sigma = 1|x, y) = p(\Sigma = 1|x) \) (Quiñonero-Candela et al. 2008). From the perspective of domain adaptation for machine learning, the covariate shift assumption implies that the data distribution differs (\( p_s(x) \neq p_t(x) \)), but the conditional label probabilities are the same (\( p_s(y|x) = p_t(y|x) \equiv p(y|x) \)). Also, \( p_t(x) \) is assumed to have support within that of \( p_s(x) \) (Huang et al. 2007; Quiñonero-Candela et al. 2008). The covariate shift scenario might arise in cases where the training data has been biased toward one region of the input space or is selected in a non-I.I.D. manner, such as with active learning. It is closely related to the idea of sample-selection bias which has long been studied in statistics (Heckman 1979) and arises from, for instance, bias in survey participation. In recent years it has been explored for machine learning (Huang et al. 2007, Zadrozny 2004, Bickel et al. 2007, Dudik et al. 2005, Sugiyama et al. 2007, Sugiyama et al. 2008, Tsuboi et al. 2009, and others).

Shimodaira (2000) considered the problem of parametric model fitting for \( p(y|x) \) under covariate shift. Given a model family \( q(y|x; \theta) \) for the conditional distribution \( p(y|x) \), one wishes to find \( \theta^*_t \) according to:

\[
\theta^*_t = \arg\max_{\theta} E_{p_t(x,y)} \{ \log(q(y|x; \theta)) \}
\]

by using only examples from the source domain \( (x_i, y_i) \sim p_s(x, y), \ i = 1, \ldots, n \). In the case that \( q(y|x; \theta^0) = p(y|x) \) for some \( \theta^0 \), then \( \theta^0 \) is the optimum on both source and target.

\footnote{This is essentially identical to the Missing At Random (MAR) assumption in the literature on missing data. In particular, we treat the unlabeled target data as missing \( y \) values, and assume the fact that a label is missing or observed does not depend on \( y \), given \( x \).}
domain. So
\[
\theta^0 = \theta^*_t \equiv \arg\max_\theta E_{p_t(x,y)} \{ \log (q(y|x; \theta)) \}
\]
\[
\theta^0 = \theta^*_s \equiv \arg\max_\theta E_{p_s(x,y)} \{ \log (q(y|x; \theta)) \}.
\]

But Shimodaira showed that this is not the case under model misspecification, when there is no \( \theta^0 \) such that \( q(y|x; \theta^0) = p(y|x) \); in this case \( \theta^*_t \neq \theta^*_s \) in general. Intuitively, a conflict arises because maximizing likelihood on the training data will result in parameter values which fit the model best to regions with high source density, but really we want to fit the model better in regions with high target density. However, under the assumptions of covariate shift, \( p_t(x, y) = p_t(x)p(y|x) = \frac{p_t(x)}{p_s(x)}p_s(x, y) \). Therefore:
\[
E_{p_s(x,y)} \left\{ \frac{p_t(x)}{p_s(x)} \log (q(y|x; \theta)) \right\} = E_{p_t(x,y)} \{ \log (q(y|x; \theta)) \}.
\]

Therefore, given \( n_s \) samples from \( p_s(x, y) \), Shimodaira suggested maximizing the “instance weighted” empirical estimate:
\[
\theta^{**} \equiv \arg\max_\theta \frac{1}{n_s} \sum_{(x_i, y_i) \in D_s} w(x_i) \log (q(y_i|x_i; \theta)),
\]
where \( w(x_i) = \frac{p_t(x_i)}{p_s(x_i)} \). Clearly this objective is an unbiased estimate of \( E_{p_t(x,y)} \{ \log (q(y|x; \theta)) \} \), and Shimodaira showed that as \( n \to \infty \), \( \theta^{**} \to \theta^*_t \). Therefore, \( \theta^{**} \) provides a consistent estimate of \( \theta^*_t \).

The theory of Shimodaira (2000) for maximum likelihood fitting, based on importance sampling, has been applied also under the more general risk minimization scenario seen in machine learning, e.g. Huang et al. (2007); Bickel et al. (2007); Tsuboi et al. (2009); Cortes et al. (2008). Given a loss function \( l(x, y, \theta) \), we want to find parameter values \( \theta \) to minimize:
\[
E_{p_t(x,y)} \{ l(x, y, \theta) \} = E_{p_s(x,y)} \left\{ \frac{p_t(x, y)}{p_s(x, y)} l(x, y, \theta) \right\}.
\]

Under covariate shift assumptions, this can be estimated from the weighted empirical risk:
\[
\sum_{(x_i, y_i) \in D_s} w(x_i) l(x_i, y_i, \theta).
\]
Solving this problem requires a learning algorithm that can handle instance weights, which is straightforward in many algorithms based on empirical loss minimization.

A number of additional issues complicate this solution, however. First, although the ideal instance weighting removes the bias of the empirical log likelihood estimate, it increases the variance over the uniform (no weighting) estimate (Shimodaira 2000). Second, it assumes that the instance weights \( w(x_i) = \frac{p_t(x_i)}{p_s(x_i)} \) are known exactly, when in many cases they must be estimated from samples. For the first problem, Shimodaira proposed using weights with a tuning parameter \( w(x_i) = \left( \frac{p_t(x_i)}{p_s(x_i)} \right)^\lambda \), where \( \lambda \in [0, 1] \) trades off between the uniform weighting \( (\lambda = 0) \) and the ideal unbiased weighting \( (\lambda = 1) \). When \( p_s(x) \) is unknown, Shimodaira suggested parametric or kernel density estimation of \( p_s(x) \), but did not consider the case when \( p_t(x) \) is also unknown.

In machine learning typically both \( p_t(x) \) and \( p_s(x) \) are unknown; samples are given in \( D_t, D_s \), but often the feature vectors \( x \) are high dimensional, making density estimation impractical. Thus, much work has focused on methods for estimating the weights without estimating the densities. Huang et al. (2007) proposed a novel procedure called “kernel mean matching” (KMM) to estimate weights \( w(x_i) \) on each \( x_i \in D_s \), based on the goal of making the weighted distribution of \( D_s \) look “similar” to the distribution of \( D_t \). Distribution similarity was measured as the difference in (weighted) sample means of the data mapped into a reproducing kernel Hilbert space, a statistic called maximum mean discrepancy, which was proposed by Gretton et al. (2007). This resulted in a quadratic program in the weights, with constraints to keep individual sample weights from being too large and a requirement that the sum of the weights be close to 1. It was also shown how to apply this re-weighting in SVM classifiers and in regularized linear regression; they reported good results on several real and synthetic test problems.

A significant amount of work on estimating the weights without directly estimating the densities has been conducted by the group of Sugiyama and colleagues. The “Kullback-Leibler importance estimation procedure” (KLIEP) was proposed by Sugiyama et al. (2008). There, similar to Huang et al. (2007), the goal was to estimate weights to maximize similarity between the target and weight-corrected source distributions, but “similarity” was formulated in terms of the Kullback-Leibler divergence \( KL(p_t(x)||w(x)p_s(x)) \). Using a sample
estimate of this divergence resulted ultimately in the objective $\sum_{x_i \in D_t} \log w(x_i)$. The weight function $w(x)$ was estimated as a linear combination of “basis functions”, such as Gaussians centered at target domain examples, in order to maximize this objective; the weights were constrained to match $\frac{1}{n_s} \sum_{x_i \in D_s} w(x_i) = 1$. Since the approach involves optimizing $w(x)$ directly on the target domain data, it is possible to perform cross-validation of the estimated weight function using held-out target domain data, which they noted is useful for “model selection” scenarios, e.g., choosing the basis functions. In their experiments, their approach compared well to the methods of [Bickel et al., 2007] and [Huang et al., 2007], and to kernel density estimation, on real benchmark and synthetic data sets. Modified approaches to estimating $w(x_i)$ in KLIEP were described in subsequent publications, such as log-linear KLIEP [Tsuboi et al., 2009] and Gaussian-mixture KLIEP [Yamada and Sugiyama, 2009]. Kanamori et al. [2009] proposed a similar approach, but with an optimization objective based on least-squares estimation of the weight function, instead of Kullback-Leibler divergence, the purpose being greater computational efficiency. Density ratio estimation in high-dimensional spaces was addressed in the work of [Sugiyama et al., 2010], by identifying a lower-dimensional subspace in which the two densities differ significantly and performing density ratio estimation only in that subspace.

Additional approaches for estimating the weights were described by [Rosset et al., 2005], [Zadrozny, 2004], [Cortes et al., 2008], and [Bickel et al., 2007]. Zadrozny [2004] adopted a generative model for covariate shift (found in other work as well) in which a binary random variable $s \in \{1, 0\}$ determines whether a target domain sample is selected into the training set $D_s$. The selection probabilities $p(s = 1|x)$ are inversely proportional to the desired weights since $w(x) = \frac{p_t(x)}{p_t(x|s=1)} = \frac{p(s=1)}{p(s=1|x)}$. The experiments of Zadrozny [2004] assumed this selection probability was known, but suggested that it could be learned automatically, avoiding the need to estimate densities explicitly—provided that samples $(x, s)$ were available. This would be the case if the source domain samples were actually a subset of the target domain samples, so that one had some data $x$ along with knowledge about whether or not it was selected. However, this information is not available in the domain-mismatch scenario that we have described—we have only positively-selected samples (the source data) and samples with unobserved selection status (the target data). Cortes et al. [2008] proposed a
weight estimation method based on clustering all the data and estimating one weight value for the training examples in each cluster, based on the proportion of source examples. Ren et al. (2008) proposed a different cluster-based method which selects training examples to balance the distribution across clusters. Rosset et al. (2005) proposed a method-of-moments procedure for estimating the sampling distribution: it assumes a parametric form for the distribution, and solves for the parameters by equating empirical moments of features in the training set with weighted empirical moments in the full set. This can be done with only the target data and the (positively-selected) source data.

A related approach described by Bickel et al. (2007) learns a classifier to estimate the weights; in particular, a binary selector random variable $\sigma$ is defined, which determines whether a sample is in the source or target domain. Unlike the source domain selection variable $s$ of Zadrozny (2004) and Rosset et al. (2005), the value of $\sigma_i$ is known for samples in both $D_s$ and $D_t$. The weight function $w(x)$ was written in terms of the ratio $\frac{p(\sigma=0|x)}{p(\sigma=1|x)}$, and they fitted a logistic regression model $p(\sigma|x; \Lambda)$ to predict domain on $D_s$ and $D_t$. The work of Bickel et al. (2007) was concerned with discriminative classifiers (specifically logistic regression). Rather than estimating the parameters $\Lambda$ corresponding to the weights first and then the parameters $\theta$ corresponding to the classifier $p(y|x; \theta)$, the proposed method learns both simultaneously in a combined objective:

$$
\sum_{(x_i, y_i) \in D_s} w(x; \Lambda) \log(p(y_i|x_i; \theta)) + \sum_{x_i \in D_s, D_t} \log(p(\sigma_i|x_i; \Lambda)) - \alpha||\Lambda||^2 - \gamma||\theta||^2,
$$

where the latter two terms represent regularization of the parameter vectors. Their joint optimization method yielded better results than a number of baselines, including sequential optimization of the weights and weighted model, density estimation, and the method of Huang et al. (2007), on spam filtering and text classification problems. However, it is not immediately clear why the combined objective works better than the sequential method: the theory of Shimodaira (2000) does not directly motivate fitting the weights to maximize weighted likelihood on $D_s$, as is implied by the second term in the objective. Related work by Bickel and Scheffer (2007) was concerned with estimating the parameters $\Lambda$, for the task of spam classification in different inboxes. They developed a model that relates the prior distributions over $\Lambda$ in different inboxes, which represent different target domains.
In Figure 2.1 we illustrate a simple binary classification problem that benefits from instance weighting for covariate shift correction. Blue and red training samples make up $D_s$, and are drawn from two Gaussians with different covariance matrices. The contour plot of $p(y = \text{blue class}|x)$ is shown on the right; the true decision boundary to minimize error would be at $p(y|x) = 0.5$ and is quadratic. A logistic regression model fit to $D_s$ leads to the linear decision boundary shown on the left (solid line). We consider the scenario where the target distribution follows the same true $p(y|x)$, but is drawn from a Gaussian centered at $(2,0)$; cyan and pink examples represent draws from this distribution corresponding to the blue and red classes, respectively. Using the known distributions of $p_s(x)$ and $p_t(x)$ to derive the correct weights, a logistic regression model is fit to the samples in $D_s$ using weighted maximum likelihood. This leads to the dashed line shown, which clearly fits the true decision boundary better in the region where $D_t$ are concentrated.

This example was designed to illustrate an ideal case in which instance weighting methods are useful. In particular, it satisfies the assumptions that:

- (i) $p_s(y|x) = p_t(y|x)$;
- (ii) $p_t(x)$ has support within that of $p_s(x)$; and
- (iii) $\theta^*_t \neq \theta^*_s$, which is a consequence of the fact that the model family (linear) does not match the very best ("true") decision boundary (quadratic) for $p(y|x)$

These are quite strict assumptions, and in many domain adaptation scenarios we cannot guarantee that they will hold. So the question we consider next is whether instance weighting methods will be useful under other conditions.

We consider (i) and (ii) first, which are assumptions about the type of domain mismatch only, and not about the learning algorithm or model. If assumption (i) is violated, then $p_s(y|x) \neq p_t(y|x)$ for some $x$. Obviously in this scenario there are no guarantees that anything could be learned about the target domain using labels only from the source domain. However, in many cases it is reasonable to assume that $p_s(y|x)$ and $p_t(y|x)$ are close for most values of $x$. And there is some work suggesting that instance weighting can be useful in other
Figure 2.1: A classification scenario in which covariate shift mismatch occurs and source-domain training is improved using instance weighting. Right: equal value contours of $p(y = \text{blue class}|x)$. Left: blue and red samples make up the training data; cyan and pink samples follow the same conditional class distribution but are drawn from a Gaussian centered at $(2, 0)$. Solid line represents decision boundary of logistic regression model trained on unweighted blue and red samples; dashed line trained on weighted samples. (Note that cyan and pink samples are not used in training, as the true distribution is assumed known.)
scenarios: Huang et al. (2007) demonstrated improvement when a biased sampling method dependent on $y$ (but not $x$) was used to derive the training examples, resulting in differing label proportions across domains. Similarly, the resampling approach of Ren et al. (2008) was shown to improve performance under different types of sample selection bias, including sampling dependent on $y$. Note that even when the sampling violates the assumption that $p_s(y|x) = p_t(y|x)$, if the classification problem is not very noisy then $y$ is well-correlated with $x$, and the biased sampling might be improved based on weights depending on $x$. Note also that if the sampling function depends on $y$, it can be corrected using weights that depend on $y$. Although typically these weights are not known and could not be estimated without labeled target data, Lin et al. (2002) described instance weighting for SVM training to correct for known differences in misclassification costs or class proportions; the weights depend on the class label of each training example.

If assumption (ii) is violated, then it does not hold that $E_{p_s(x,y)} \left\{ \frac{p_t(x)}{p_s(x)} l(x, y, \theta) \right\} = E_{p_t(x,y)} \left\{ l(x, y, \theta) \right\}$, because $p_s(x, y) = 0$ for some $x, y$ with $p_t(x, y) > 0$. Of course, it is still possible to do weighted empirical risk minimization; even though $w(x)$ is infinite for $x$ with $p_s(x) = 0$, such $x$ will not occur in $D_s$. However, weighted training is not guaranteed to be useful in this situation because it can only improve the fit of the learned model on the region where $p_s(x) > 0$, and that may not necessarily improve the fit on the entire space where $p_t(x) > 0$. In fact, it is conceivable that the fit over all of $p_t(x)$ will get worse. Assumption (ii) is actually quite strict and unlikely to be satisfied in many high-dimensional NLP problems, where domain mismatch is characterized by target domain n-gram cues that are not found in the source domain. This suggests that instance weighting may not be useful in many such problems. In fact, while Huang et al. (2007) and Sugiyama et al. (2008) reported success with their instance weighting methods on “real” benchmark datasets, they used an artificial sampling procedure to derive the source and target domains, which ensured that assumption (ii) held. However, Bickel et al. (2007) reported success using more realistic sampling scenarios, e.g., classification of research papers where source and target domain come from different years.

Assumption (iii) describes the relationship between the chosen model family and the data distribution. It suggests that, even if (i) and (ii) are satisfied, instance weighting will
be useful only when the classifier chosen is “mismatched” to the actual decision boundary. This issue was considered by Zadrozny (2004), who focused on the question of how various classifier models (generative classifiers, logistic regression, decision tree, and SVM) would be affected by covariate shift. Classifiers were typed as “global” or “local”, depending on whether or not the learner depends asymptotically on $p(x)$ in addition to $p(y|x)$; the author concluded that logistic regression, “Bayesian” generative classifiers, and SVMs (under the condition of separable data) are local classifiers that are not affected by covariate shift. However, (as also pointed out by Fan et al. (2005)), this analysis failed to take into account the relationship between the model and the data: parametric learning scenarios do not usually model the true $p(y|x)$, but instead fit parameters of an assumed model family. In general, any parametric classifier model can be affected by covariate shift if there is not a single solution $\theta$ in the model family that minimizes expected loss with respect to both $p_s(x, y)$ and $p_t(x, y)$ simultaneously. This occurs under model mismatch or misspecification, meaning that the model family considered does not contain the “best” model, which minimizes loss at all regions of the feature space. Linear decision boundary models, including logistic regression and linear kernel SVMs, can be affected by covariate shift if the true decision boundary is not linear, as demonstrated in Figure 2.1. Under these conditions, such learners are affected by covariate shift, and do indeed have the potential to improve from instance weighting. 2

Other work has noted that instance weighting for covariate shift is not useful when modeling $p(y|x; \theta)$ using a model family that contains the true distribution $p(y|x)$ (Shimodaira 2000; Fan et al. 2005; Storkey 2009; Tsuboi et al. 2009; Kanamori and Shimodaira 2009). It is also not useful in hard-margin SVMs (which do not model $p(y|x)$) when the classes are separable, as noted by both Zadrozny (2004) and Fan et al. (2005): if a solution can be found for which the loss is zero, then weighting some samples’ loss over others has no effect. 2

---

2We should distinguish two questions: (i) whether a classifier is sensitive to covariate shift; (ii) whether it could benefit from instance weighting. These questions were somewhat confused in the works of Zadrozny (2004) and Fan et al. (2005): the question being considered there was (ii), but it was phrased as (i). In the present work we are considering (ii). Note that even classifiers that cannot benefit from instance weighting can still be sensitive to covariate shift due to finite training sets, which might lack samples in important regions of the feature space. Unfortunately this problem cannot be corrected by weighting the available samples, but only by drawing more samples.
We illustrate generally that instance weighting is not useful for *unregularized* risk minimization learning algorithms when the minimizers of $E_{p_t(x,y)}\{l(x, y, \theta)\}$ and $E_{p_s(x,y)}\{l(x, y, \theta)\}$ are the same. We can write:

$$E_{p(x,y)}\{l(x, y, \theta)\} = \int_x p(x) \int_y p(y|x)l(x, y, \theta)dydx = \int_x p(x)G(x)dx,$$

where $G(x) \geq 0$. For each value $x_0$, there is a minimum possible $G(x_0)$ which is defined by $p(y|x)$, $l(\cdot)$, and the model family that we are allowed to choose from $\theta$. In particular, it is the value:

$$G^*(x_0) = \int_y p(y|x_0)l(x_0, y, \theta^*)dy,$$

where $\theta^*$ was chosen to minimize $G^*(x_0)$. We cannot generally achieve this value at each $x$, because we are not allowed to choose a different $\theta^*$ at each $x$—we are constrained to pick one for all $x$. Our choice typically depends on $p(x)$, because the constraint means there is a trade-off of different losses at different values of $x$. We can choose $\theta^*$ to have a very small $G(x_1)$, but then it might have a large $G(x_2)$. So the choice is sensitive to covariate shift, which changes the relative weights of $G(x_1)$ and $G(x_2)$ by changing $p(x_1)$ and $p(x_2)$.

However, in the case that our model family does contain a single value $\theta^*$ that achieves the minimum $G^*(x)$ at each $x$, then there is no longer a dependence on $p(x)$, which eliminates the justification for instance weighting. (A similar argument was shown by Hein (2009).)

Even if $\theta^*$ generally depends on $p(x)$, it may be the case that $\theta^*_s = \theta^*_t$ for a particular $p_s(x)$ and $p_t(x)$. In this case there would be no benefit for instance weighting either (at least according to the motivations in the work cited here). The purpose of instance weighting is to provide an unbiased estimate of $E_{p_t(x,y)}\{l(x, y, \theta)\}$ so that an estimate of $\theta^*_t = \text{argmin}_{\theta} E_{p_t(x,y)}\{l(x, y, \theta)\}$ can be found, but if $\theta^*_s = \theta^*_s$ as well, then one should just use the estimate of $E_{p_s(x,y)}\{l(x, y, \theta)\}$, since the weighted estimate has higher variance.

Intuitively, it is more likely that $\theta^*_s = \theta^*_t$ if the model space is large, because with a more complex decision boundary there is less trade-off to fit $p(y|x)$ at different values of $x$. Thus, we would expect that instance weighting would not be useful in very high dimensional feature spaces, or with complex model families. Hein (2009) phrased this in terms of the “capacity” of the classifier to fit different functions. He noted that since a large capacity classifier can minimize $G(x)$ pointwise, instance weighting is only useful for classifiers of small capacity;
he argued that KNN and Gaussian kernel SVMs have “maximal” capacity because they can learn any $p(y|x)$ given infinite data. Storkey (2009) argued that instance weighting is not useful for models/learning algorithms that display “Kolmogorov consistency”, meaning the conditional label likelihood given the labeled data is independent of additional unlabeled data. He argued that in general, when learning under covariate shift, instance weighting provides only a “computational benefit” rather than a “modeling benefit”, in that it allows one to select a model family with low complexity.

However, success has been reported with instance weighting in high-dimensional feature spaces and large capacity models. Huang et al. (2007) reported successful results using SVMs with Gaussian kernels on several benchmark machine learning datasets using artificial sampling procedures (which did not always match the covariate shift assumption), while Son et al. (2009) reported results using SVMs on a high-dimensional text-chunking task. Bickel et al. (2007) showed improvement with a kernel logistic regression classifier on tasks including classification of research paper topics when the train and test sets come from different time periods; detection of landmines in different regions; and email spam detection for an individual using a public pool source domain. In these cases, the utility of instance weighting seems to come from the fact that the objective has an additional regularization term. This potentially favors solutions with low model complexity but with greater than minimum-possible loss on the training set. In the weighted version of the objective, more importance is placed on minimizing loss of heavily-weighted samples compared with minimizing model complexity. Chapter 4 considers instance weighting under regularized models in more detail.

Another interesting application of instance weighting in the NLP setting was described by Duh and Kirchhoff (2011). The task considered was supervised training of relevance rankers of retrieved documents. Given an unranked list of documents corresponding to a particular query, they used domain adaptation methods to train a ranker tuned to that list of documents; in other words, the list of documents for the test query serves as $D_t$. Methods included unsupervised feature learning (based on Kernel PCA), and instance weighting; in the latter case, they viewed pairs of documents as the instances, and KLLIEP was used to derive weights for each training pair of documents, which were then incorporated into a
cost-sensitive version of boosting. Both adaptation approaches generally led to improved ranking performance.

Several works have analyzed the target domain generalization error under covariate shift or instance weighting. Hein (2009) focused on the general sample selection bias case, where \( p_s(y|x) \) and \( p_t(y|x) \) are not necessarily equal, and derived expressions for the target domain error of the theoretically optimal source domain “Bayes” classifier (implied by \( p_s(y|x) \)) in terms of \( p(x) \) and the sample selection distribution \( p(\Sigma = 1|x, y) \), assuming all of these are known. In particular, Hein gave a condition for the source and target Bayes classifiers to agree at given \( x \) in terms of the sample selection probability at that \( x \); and suggested using this to remove examples from training where the Bayes classifiers disagree. Cortes et al. (2010) derived bounds on the target domain error in terms of the source domain weighted empirical error; the bound reflects the impact of variance and bias, and suggests a strategy for picking instance weights for the training data. Analysis of the effect of instance weight estimation error on target domain generalization error under certain regularized learners was conducted by Cortes et al. (2008). And in the work of Mansour et al. (2009a), a bound was derived on the target domain generalization error, in terms of a proposed distribution distance, and instance weighting is proposed as a way to minimize this distribution distance term.

In instance weighting adaptation as well as other kinds of adaptation, certain learning and model hyper-parameters must be set without labeled target data. Sugiyama et al. (2007) and Shimodaira (2000) both addressed the problem of choosing model hyper-parameters, such as the regression function order, the regularization parameter \( \gamma \), or instance weighting adaptive parameter \( \lambda \), under sample selection bias scenarios when no held out labeled target data is available. Shimodaira proposed minimizing an information criteria objective that includes a weighted likelihood on the source examples, while Sugiyama et al. proposed \textit{importance-weighted cross-validation}, which performs leave-one-out cross-validation evaluated on weighted source examples. Lin et al. (2002) also addressed the problem of selecting hyper-parameters, specifically for SVMs under class-dependent instance weighting; they proposed either tuning on a weighted held-out set from the source domain, or modified versions of previously-proposed objectives used to approximate a held-out-set loss.
Recently, Cortes et al. (2010) proposed another way to alter weights in order to trade off bias and variance: averaging the weights within quantile bins. They propose optimizing the number of quantile bins based on a term in a proposed generalization error bound, but this still requires setting a tuning parameter within that bound.

A related domain adaptation approach uses weighted combinations of multiple source domains. In the work of McClosky et al. (2010), for parsing, a regression model was trained to predict target domain performance from domain similarity features; it was then used to select the optimal mixture of source domains for a given target domain. Rogati (2009) applied weighting to training corpora or instances in order to better match the target domain, for the tasks of machine translation and cross-language IR. Weighting was based on similarity measures such as cosine similarity or vocabulary overlap, or a domain-generative probability. In more theoretical work, Blitzer et al. (2008) and Mansour et al. (2009b) investigated target domain error bounds based on weighted combinations of source domains, or source and target domains.

2.3.1 Our Work

For our work, there are several questions regarding the application of instance weighting, such as whether our domains satisfy the covariate shift assumptions, and what kind of weight estimation method would be most appropriate. Also, we use linear regularized models in much of our work, and the trade-offs between the regularization and instance weights under these models has not been fully explored in the literature, to our knowledge. Therefore, we investigate the latter issue in the first part of Chapter 4, and then perform instance weighting experiments on two text classification datasets that illustrate naturally-occurring domain mismatch, the Sentiment Classification dataset used by Blitzer et al. (2007) and the 20Newsgroups dataset used by Dai et al. (2007a). A natural approach to “density estimation” for these textual problems is to use language models. We compare weight estimation methods based on discriminative modeling (proposed by Bickel et al. (2007)) and density estimation using language models, and we implement instance weighting using weighted logistic regression learning based on the LIBLINEAR package (Fan et al. 2008).
Although instance weighting is well-motivated theoretically, it cannot alleviate problems caused by different feature sets. For example, in Chapter 6 we perform dialog act classification using high-dimensional n-gram features, where some key features in the target domain are not seen in the source domain. In Chapter 7, we perform question and boundary detection on speech by training on text, but the text source data lacks the prosody features that are useful in the speech target domain. Therefore, we do not apply instance weighting to these problems. However, we do investigate instance weighting for cross-corpus prosody classification in Chapter 5, where the domains are expected to have the same feature set.

2.4 Feature Representation Approaches

Another class of unsupervised domain adaptation methods is based on changing the feature representation $x$ to better represent shared characteristics of the two domains. It makes the assumption that certain features are domain-specific while others are generalizable, or that there exist mappings from the original feature space to a latent feature space that is shared between domains. This is the basic idea in the approaches of Blitzer et al. (2006), Blitzer et al. (2007), Ando et al. (2005), Ben-David et al. (2007), Dredze et al. (2006), Pan et al. (2008), Jiang and Zhai (2007b), Satpal and Sarawagi (2007), Blitzer et al. (2009), Pan et al. (2010), Kolcz and Tsoi (2009), Chen et al. (2009), Huang and Yates (2009), Huang and Yates (2010b), Huang and Yates (2010a), Wang et al. (2008b), Guo et al. (2009), Raina et al. (2007), Aue and Gamon (2005), Arnold et al. (2007), and Pan et al. (2009).

We distinguish two classes of the feature representation approach. The first, which we call the “distribution similarity” approach, aims explicitly to make the source and target domain sample distributions similar, either by penalizing or removing features whose statistics vary between domains (Aue and Gamon 2005; Jiang and Zhai 2007b; Satpal and Sarawagi 2007; Arnold et al. 2007) or by learning a feature space embedding or projection in which a distribution divergence statistic is minimized (Pan et al. 2008, 2009; Chen et al. 2009). The second, which we call “latent feature learning,” aims to construct new features by analyzing large amounts of unlabeled source and target domain data (Blitzer et al. 2006, 2007; Huang and Yates 2010b, 2009, 2010a; Pan et al. 2010; Blitzer et al. 2009; Ciaramita and Chapelle 2010). We describe several approaches in detail next.
2.4.1 Distribution Similarity Approaches

Satpal and Sarawagi (2007) considered domain adaptation for conditional random fields, for which the goal is to learn a weight vector \( \mathbf{w} \) on “features” \( f_k(x, y) \) that are functions of both a structured input \( x \) (such as a sequence of words) and a structured output \( y \) (such as a sequence of labels). The goal of Satpal and Sarawagi was to select a subset of the features \( f_k(x, y) \) so as to minimize a distance between means of the two domains, while simultaneously maximizing classification performance on the source-domain training data. The “distance” measure between domains was actually the sum of distances between sample means for each feature. In practice, a soft feature selection problem was considered, such that entries \( w_k \) in the weight vector are encouraged to have smaller values for features \( f_k(x, y) \) that have a large “distance” between domains. This resulted in the objective:

\[
\arg\max_{\mathbf{w}} \sum_{i \in D_s} \sum_k w_k f_k(x_i, y_i) - \log(z_w(x_i)) - \lambda \sum_k |w_k|^\gamma d(E_s\{f_k(x, y)\}, E_t\{f_k(x, y)\}),
\]

where the first two terms compose the basic CRF objective and the last incorporates the feature distance penalty between domains. They noted that this can be interpreted as a regularization of \( \mathbf{w} \) where each component \( w_k \) is regularized proportional to the feature mean distance in feature \( k \). Since the feature mean estimates \( E\{f_k(x, y)\} \) depend on \( \mathbf{w} \), learning followed an iterative procedure whereby they computed the distances between feature means, updated the weights \( \mathbf{w} \); fixed the weights, and updated the feature means. A similar idea for maximum entropy classifiers was described by Arnold et al. (2007); however, instead of penalizing features with large divergence, they scaled each feature in the source domain so that its expected value matched that in the target domain. The work of Jiang and Zhai (2007b) was motivated by the goal of using features that are generalizable across domains. They presented a method using a regularized logistic regression classifier to allow the generalizable features to be regularized less in training, compared with the domain-specific features. However, their method for finding the generalizable features assumes that there are multiple source domains.

A simpler approach for NLP tasks is simply to avoid training on features (words or n-grams) that are absent from the target domain data. This approach was investigated
by Aue and Gamon (2005) for sentiment classification; they found that the approach was successful at improving cross-domain performance in some cases, but degraded performance in others. Dredze et al. (2007) reported no effect for cross-domain parsing. In Chapter 3 we analyze the feature restriction approach in further detail.

The approaches of Pan et al. (2008) and Chen et al. (2009) aim to find a feature representation in which the source and target distributions are similar. Both approaches are based on minimization of feature distribution difference as measured by the maximum mean discrepancy (MMD) statistic (Borgwardt et al. 2006), which is the distance between sample means in a reproducing kernel Hilbert space. The approach of Pan et al. (2008), called maximum mean discrepancy embedding (MMDE), is a kernel-learning method based on a maximum variance unfolding (MVU) (Weinberger et al. 2004). MVU tries to find a kernel to “unfold” a low-dimensional manifold of the data by maximizing variance subject to fixed distances between neighbors. The method of Pan et al. (2008) combines that unfolding objective with MMD minimization, and subsequently applies kernel PCA with the learned kernel. Note that this is primarily a transductive approach: it does not actually find the kernel but just the kernel matrix (i.e., the embedding) for the given source and target data. Furthermore, it requires solving a semi-definite program (SDP), which is an obstacle on large datasets. However, Pan et al. (2009) later proposed a new version called “transfer component analysis” that only requires solving an eigenvalue decomposition, and gives a kernel function that can be evaluated on new data. An interesting aspect of the methods of Pan et al. (2008, 2009) is that although source domain labels are available, they are not used in finding the new feature representation. Pan et al. (2008) noted that their method is related to colored MVU (Song et al. 2008), which modifies basic MVU by incorporating “side information” and can be used to try to maximize dependency between features and labels. In theory, this modification could be incorporated into the objective of Pan et al. (2008). The approach of Chen et al. (2009) has the same goal based on MMD minimization, but finds a linear projection rather than finding a kernel matrix. Effectively, it aims to find an orthogonal linear projection into a reduced dimensional space such that the sample means of the source and target data are close after being projected into that space. Simultaneously, their objective aims to minimize label prediction loss in the source domain, using a feature
representation composed of both projected and original features. As noted by Chen et al. (2009), the proposed method is more appropriate than MMDE for large datasets since it does not require solving a SDP, and furthermore, the learned projection can be applied to new test data.

The approaches of Pan et al. (2008), Pan et al. (2009) and Chen et al. (2009) bear some similarity to covariate shift approaches that aim to maximize source/target distribution similarity—in particular the method of Huang et al. (2007), which is also based on the MMD statistic. The difference is that these approaches are based on learning new feature representations, while the method of Huang et al. (2007) is based on weighting training samples.

2.4.2 Latent Feature Learning Approaches

Methods Based on Linear Projections

In domain adaptation scenarios, it is common that some features in the target domain are zero or constant in the source domain, and vice versa. For example, when adapting a sentiment classifier from the book review to the appliance review domain, the word “loud” may be an important cue word for the target domain, but may not occur at all in the labeled source data. When adapting a question detector from text to speech, the speech features are not present in the source domain. Let $X = [x_1, ..., x_d] = [X_s, X_t, X_b]$ be the original feature representation, where $X_s$ represent features that are nonzero only in the source domain, $X_t$ represent features that are nonzero only in the target domain, and $X_b$ occur in both domains. By training on the original feature representation $X$ using the source data, the classifier will not be able to use the features in $X_t$; however, using unlabeled source and target data together we might be able to derive a new feature representation that aggregates features in $X_s$, $X_b$ and $X_t$ which behave similarly. Many methods exist that construct aggregated features as linear combinations of the original features. The goal is to learn a set of $l$ feature weight vectors $\{\beta_k\}_{k=1}^l$ which are used to project the original feature vectors into a new feature space. The new feature representation will be $X = [f_1, ..., f_l]$
where:

\[ f_1 = \sum_{i=1}^{d} \beta_{1,i} x_i \]

\[ f_2 = \sum_{i=1}^{d} \beta_{2,i} x_i \]

... 

Because \( f_k \) is a sum of features including \( X_t \), the classifier can use those features implicitly without actually seeing them during training. Methods that derive such linear feature transformations include latent semantic analysis (LSA), principal component analysis (PCA), structural correspondence learning (SCL), and canonical correlation analysis (CCA). Generally, these methods use observed feature co-occurrences in the unlabeled source and target examples to derive the new feature space.

LSA and PCA are popular methods for unsupervised dimensionality reduction. These methods use the singular value decomposition (SVD) of the example-feature matrix to compute a low-rank data representation. LSA (Deerwester et al. 1990) was proposed for document retrieval and usually operates on data represented by sparse word count features; it is hoped that the dimensions in the reduced space will group together words that occur in similar contexts. PCA is similar to LSA but operates on data with the mean subtracted, so that the resulting top singular vectors represent the directions of highest variance in the original feature space, rather than including the mean. In the domain adaptation setting, PCA or LSA can be performed on the collection of source and target data together. Under ideal circumstances, feature representation in terms of the singular vectors can group together words in the source and target domains that occur in similar contexts, so target domain words that do not occur in training can still be used for classification in the latent space. Experiments with SVD on the example-feature matrix for domain adaptation on a variety of NLP tasks have been conducted by Ando (2004), Huang and Yates (2009), Ciaramita and Chapelle (2010), and Pan et al. (2010), with mixed results. In the work of Guo et al. (2009), Latent Dirichlet Allocation (LDA) (Blei et al. 2003), a probabilistic model for the generation of data from latent variables, was successfully used for domain adaptation on the task of named entity recognition. The LDA model was learned on unlabeled multi-
domain data, and the inferred distribution over latent topics (or the top topics in this distribution) was used to augment the feature representation of each word.

Structural correspondence learning (SCL), proposed by Blitzer et al. (2006, 2007), is a latent feature linear projection method which has recently generated a lot of interest. SCL is based on alternating structure optimization (ASO), which was proposed for semi-supervised learning (Ando et al. 2005). ASO uses many “auxiliary” learning tasks on the unlabeled data to learn the “predictive structure” of the data. Parameter vectors learned in the many auxiliary tasks are collected and used in an SVD decomposition which leads to a reduced dimensional feature space. Blitzer et al. proposed applying this method to domain adaptation by learning the auxiliary tasks over unlabeled source and target data together. This was called “structural correspondence learning” since it has the ability to learn correspondences between source-only and target-only features.

In SCL, the auxiliary learning problems consist of predicting the presence or absence of “pivot” features which are found in each domain. For each pivot feature \( l \), a linear classifier is learned to predict \( l \) based on \( \text{sign}(w_l \cdot x) \), where \( x \) is the vector of original word features (excluding the pivot feature). The learned weight vectors \( w_l \) (one for each pivot feature) are collected into a matrix, and then SVD is performed on the matrix, resulting in a reduced dimensional space formed by the top \( h \) singular vectors. The correspondence is learned because source and target domain features that get similar weights in \( w_l \) across a variety of different prediction tasks will be grouped together into a singular vector. For example, when adapting between books and kitchen appliances, Blitzer et al. (2007) found that “predictable” in the books domain and “defective” in the kitchen domain were given weights with the same sign in one singular vector, indicating that they have similar behavior. They proposed selecting the pivot features automatically either by high frequency in both domains, or by additionally requiring high mutual information with the label in the source domain. The mutual information method encourages the auxiliary prediction tasks to be related to the labels, which is a difference from the purely unsupervised methods such as LSA/PCA, which do not incorporate label information.

In the original version of SCL, both the new and original features are used in the final classifier, with higher weight on the new features. The selection of the optimal SCL param-
eters (number of dimensions of the SVD, weight on the final SCL features) could present a problem if there is no available labeled target domain data, although Blitzer et al. (2006) claimed that performance was not very sensitive to the number of dimensions, and that the weight could be set on a heldout set from the source domain. SCL has shown some success on a variety of tasks. In addition to the original part-of-speech tagging and sentiment classification tasks, it has been applied to cross-language sentiment classification using machine translation (Prettenhofer and Stein 2010; Wei and Pal 2010); conversation summarization (Sandu et al. 2010); and entity recognition (Ciaramita and Chapelle 2010). In these works it has mostly been applied to sparse word features; it was not effective for low-dimensional feature representations shared between domains in the work of Sandu et al. (2010). Some improvements to the SCL algorithm for sentiment classification were proposed by Tan and Cheng (2009), based on feature and example weighting. Recently, Ji et al. (2011) proposed another version of SCL, which learns separate predictors on each domain; they claimed this could eliminate some problems with contradictory predictor features across domains.

Blitzer et al. (2009) showed generally how domain transfer is possible when there exists a reduced dimensional subspace preserving label predictive information. In addition to SCL, they suggested Canonical Correlation Analysis (CCA) as one method to learn the reduced dimensional subspace. CCA is an unsupervised dimensionality reduction method based on two views (feature representations) for each example (which could be constructed artificially by splitting the feature set); it constructs a linear projection for each view into a new space such that projections are maximally correlated. Under certain assumptions about the relationship between the views, CCA has the ability to derive a subspace preserving label predictive information. Blitzer (2007) pointed out that CCA resembles SCL in some respects, and presented a squared-loss-based, “whitened” version of SCL that is identical to CCA. Note generally that CCA learns weights on one subset of features (one view) which are predictive of values computed from the other view; this resembles the auxiliary pivot prediction tasks in SCL, where weights are learned on non-pivot features in order to predict the pivot features.
**Other Latent Feature Learning Approaches**

Huang and Yates (2010b, 2009, 2010a) described an adaptation approach for sequence modeling tasks based on latent features that, like SCL, are learned via auxiliary prediction tasks trained on source and target data together. In this case an HMM is used to try to predict the observed word sequences; the hidden states found from decoding constitute the additional features. This is described as a kind of sequence smoothing, with particular motivation coming from rare words which might not be seen in training. Their method gave significant improvements on both same-domain and cross-domain part-of-speech tagging (with improved performance over SCL) as well as NP chunking. A related approach also led to cross-domain improvement on the task of semantic role labeling (Huang and Yates 2010b), and another (Huang and Yates 2010a) proposed using hidden states from multiple HMMs, which were initialized differently in training.

Additional approaches include feature clustering based on co-occurrence statistics (Pan et al. 2010; Dai et al. 2009b) and propagation of feature prediction weights via a feature co-occurrence graph (Chen et al. 2009). In the work of Pan et al. (2010), features are divided into domain-specific and domain-general, and a bipartite graph is constructed where nodes represent features and weights represent co-occurrence; spectral clustering on the graph creates shared feature clusters, which are used to augment the original feature representation for supervised training in the source domain. In their experiments on cross-domain sentiment classification, their method performed better than both SCL and LSA in many cases. In the method of Dai et al. (2009b), an example-feature-label graph is constructed based on co-occurrences; spectral clustering on this graph is used to derive a new feature representation.

**Learning Across Different Feature Spaces**

We mention some research dealing with the specific case where the source and target data are assumed to exist in different feature spaces. There exists extensive work on cross-language text classification and information retrieval, where the training and test data are in different languages. Some of this work used automatic translation of one domain fol-
ollowed by unsupervised domain adaptation to bridge the difference between the native and translated data (Wei and Pal 2010; Prettenhofer and Stein 2010; Ling et al. 2008c; Rigutini et al. 2005; Shi et al. 2010). Other methods derive a projection from each language into a “language-independent” space using training documents represented in both languages (which could come from parallel corpus, machine translation, or other methods). Examples of this approach are the methods of Dumais et al. (1997), Platt et al. (2010), and Vinokourov et al. (2003); the latter uses canonical correlation analysis (CCA) with each language constituting a view. Once the language-independent space is learned, new documents in one or the other language can be projected into that space and compared against documents in either language.

Dai et al. (2009a) described “translated learning”, which can use labeled data in one feature space to help train a model to classify examples in another feature space. They considered two tasks: using text data to help classify images, and cross-language document classification. The approach relies on a probabilistic “translation model” between the feature spaces, which can be derived from dictionaries or from data that has both kinds of features. The translation model, together with the generative model learned on the source domain feature space, allows them to derive generative probabilities on the target domain feature space by marginalizing over the unknown source features. In the work of Shi et al. (2010), label-conditional word translation probabilities are learned via the EM algorithm using only a bilingual dictionary, labeled source language data, unlabeled target language data.

2.4.3 Other Feature Representation Approaches

In addition to the methods described above, work on classifier robustness is relevant, and has been applied to domain adaptation, for example in the approach of Kolcz and Teo (2009). This approach does not make use of the unlabeled target domain data, but tries to construct a classifier that is robust to changes in the distribution between the source and target domain, for instance by discouraging heavy dependence on only one feature. Kolcz and Teo (2009) described several known methods, including “feature noise injection” and “feature bagging”. They proposed a new method called feature reweighting, in which
heavily-weighted features in the first pass of training are regularized more heavily in the second pass of training.

Finally, some feature normalization methods also can be considered as feature representation approaches to domain adaptation. Mean and variance normalization are simple methods that make the distributions in the source and target domains more similar, by forcing them to have the same mean and variance. In ASR, vocal tract length normalization (VTLN) methods derive a warping function to be applied to the frequency scale for each speaker; in one approach, a linear warping factor is estimated using maximum likelihood under a given HMM model (Andreou et al. 1994; Lee and Rose 1998). Thus the warping factor changes the feature representation for the speaker’s data so that it is better represented by the model.

2.4.4 Comparison of Assumptions

In Section 2.3 we discussed covariate shift approaches to domain adaptation, which have a similar objective to many of the feature representation approaches—namely to make the empirical source and target distributions look similar. Covariate shift approaches achieve this by instance weighting, while feature representation approaches achieve it by feature transformations. Although both kinds of methods have a similar goal, they address different scenarios and make different underlying assumptions. Covariate shift approaches assume that \( p_t(x) \) has support within \( p_s(x) \) and that \( p_s(y|x) = p_t(y|x) \) in the region where \( p_t(x), p_s(x) > 0 \). By contrast, many feature representation approaches are motivated by high dimensional NLP problems, where it is assumed that certain features are domain-specific, so these methods do not assume that \( p_t(x) \) has support within \( p_s(x) \). Assumptions about the relationship between \( p_s(y|x) \) and \( p_t(y|x) \) are less well defined for feature representation approaches, since \( x \) is changed to a new representation \( \phi(x) \) during adaptation. In general, these approaches assume that for the new feature representation \( \phi(x) \), \( p(y|\phi(x)) \) is similar across domains. The approach of Pan et al. (2008) was motivated by the scenario where the original features arise from a mix of domain-specific and domain-general factors, although they also successfully applied it in the high-dimensional NLP case. Satpal and
Sarawagi (2007) and Arnold et al. (2007) assumed that $p_s(x, y) \neq p_t(x, y)$, with differences limited to certain feature dimensions.

2.4.5 Measuring Distribution Similarity to Predict Cross-Domain Performance

There have been some theoretical analyses of domain adaptation based on the idea of distribution similarity in feature space. Ben-David et al. (2007) derived a bound on the target domain generalization error for a classifier trained on the source domain, under the assumptions that $p_s(x) \neq p_t(x)$. The bound includes the source domain training error rate and two measures of domain difference: the first measures how well a single classifier can perform on both domains simultaneously, and the second measures how close $p_s(x)$ and $p_t(x)$ are, using the $\mathcal{A}$-distance measure originally described by Kifer et al. (2004). They ignore the first and focus on the $\mathcal{A}$-distance, which is closely related to the error rate of the best discriminator between source and target samples. They argue that by choosing a feature representation that has both low source domain training error and low $\mathcal{A}$-distance between source and target samples, a smaller error bound can be achieved, and they illustrate an example where SCL achieves low values for the terms in the bounds. Mansour et al. (2009a) gave additional generalization bounds based on “discrepancy distance”, which is more appropriate than $\mathcal{A}$-distance for loss functions besides error rate (0-1 loss). However, unlike Ben-David et al. (2007), they proposed to derive weights on training instances to minimize the discrepancy distance, rather than finding a new feature representation. Thus, this is really an instance-weighting approach.

The significance of the works of Ben-David et al. (2007) and Mansour et al. (2009a) is that, under certain assumptions (such as the existence of a single classifier that can perform well on both domains simultaneously), we expect that the target domain generalization performance will be somewhat related to how similar $p_s(x)$ and $p_t(x)$ are (as well as to the source domain error rate). This captures the intuition that a larger difference between $p_s(x)$ and $p_t(x)$ means a larger possible difference in error rate between the domains. Since the results are only bounds (and furthermore, can only be approximated), they cannot be used to predict actual target domain performance, but given different possible feature

representation methods, we might compare the computable terms in the bound and discard the methods that lead to a large $A$-distance, for example.

In a more experimental paper, Van Asch and Daelemans (2010) proposed simpler domain similarity metrics based on relative frequencies of word features. Part-of-speech tagging experiments among many pairs of domains allowed them to compare the actual target domain accuracy against the difference metrics, with good correlation. A related analysis was conducted by Xue et al. (2008). Distribution similarity has also been used to analyze the latent feature learning approaches. In the work of Huang and Yates (2010a), Jenson-Shannon divergence between the source and target feature distributions was used to compare the original word-based representation and the distribution over the new learned features, which represent latent HMM states. In the work of Blitzer et al. (2007), classifiers were trained to predict domain membership for pairs of domains using the SCL feature representation; the empirical error rates of these classifiers (an approximation to $A$-distance) were shown to be correlated with the adaptation performance using the SCL method.

2.4.6 Our Work

In comparison with instance weighting, the assumptions behind some feature representation approaches are less clear. For example, feature restriction and LSA make some intuitive sense as domain adaptation methods, but the circumstances under which they will work and fail have not been clearly explored. That motivates our work in Chapter 3, where we discuss some possible justifications for feature restriction, LSA and SCL. We include empirical comparisons on synthetic data designed to examine the performance of these methods when the assumptions behind them degrade, and we also compare them on benchmark document classification datasets that have been widely used in domain adaptation work.

SCL has been shown to be successful on high-dimensional NLP problems like sentiment classification, where certain key target-domain features are unobserved in the source domain. In Chapter 6, we investigate the problem of classifying utterances into dialog acts, where the source and target data come from different languages, with one translated. This problem is similar to sentiment classification in that we classify utterances based on high-dimensional
n-gram features. It also suffers from missing target-domain features, which motivates our application of SCL to this problem.

Although SCL has mainly been presented as a way to incorporate word features, it might also potentially be useful for incorporating new “types” of features found in the target domain that are absent from the source domain. That application would be particularly useful for spoken language processing tasks which use a variety of feature types (such as word, acoustic, and conversational features). We investigate that scenario in Chapter 7, using SCL to incorporate prosody features found in the speech domain that are missing from the source text domain.

2.5 Self-labeling Approaches

Self-labeling approaches include self-training, co-training (Blum and Mitchell 1998), and Maximum Likelihood Linear Regression (MLLR) (Leggetter and Woodland 1995; Digalakis et al. 1995). These are iterative methods that train an initial model based on the labeled source data, use that to estimate labels on target data, then use the estimated labels for building another model. These methods have been applied in both supervised and unsupervised domain adaptation settings; we focus on their application to unsupervised settings.

2.5.1 Self-training

In self-training, $D_s$ are used to train an initial model, which is then used to guess labels or label probabilities for $D_t$. On the next round, $D_t$ are incorporated to train a new model. This is carried out repeatedly, either for a fixed number of rounds, or until convergence. Variations exist as to how the samples in $D_t$ are added and used. Some approaches add only the top $n$ samples with the highest label confidence on each round (balancing for class priors), while others use all the data on each round, repeatedly adjusting the labels for those data on subsequent rounds. The “hard” version adds samples with a single label, as though the labels were known with certainty, while the “soft” version incorporates label confidences when fitting the model on the next round. Self-training has a close relationship with the Expectation Maximization (EM) algorithm. We now review the EM algorithm in the context of semi-supervised learning, and then mention some implementations for
domain adaptation. We review both the standard EM algorithm and a popular variant called “classification EM” (or sometimes “hard-decision EM”). See Borman (2009) and Gupta and Chen (2011) for tutorials, and McAllester (2007) for discussion of hard-decision EM vs. standard EM.

**Standard EM**

The basic EM algorithm (Dempster et al. 1977) aims to maximize the log likelihood \( \log p(x|\theta) \) of observed data \( x \), where the computation of \( p(x|\theta) \) depends on some “hidden” or missing variables \( z \) that must be marginalized out. Consider \( n \) IID observations \( x_i, 1 \leq i \leq n \), with unobserved variables \( z_i \). The objective of EM in this context is to maximize over \( \theta \):

\[
L(\theta) \equiv \log(p(x_1, ..., x_n|\theta)) = \sum_{i=1}^{n} \log p(x_i|\theta) = \sum_{i=1}^{n} \log E_{z_i|x_i;\theta_l}\{p(x_i|z_i;\theta_l)\}. \quad (2.1)
\]

EM finds a local maximum of this function using an iterative procedure that alternately computes a distribution over hidden variable values \( p(z_i|x_i;\theta_l) \) at a given point \( \theta_l \), and then selects a new value \( \theta_{l+1} \) that maximizes:

\[
Q(\theta|\theta_l) = \sum_{i=1}^{n} E_{z_i|x_i;\theta_l}\{\log p(x_i, z_i|\theta_l)\}. \quad (2.2)
\]

Maximizing \( Q(\theta|\theta_l) \) is equivalent to maximizing \( Q(\theta_l) - Q(\theta_l|\theta_l) \), which is a lower bound on \( L(\theta) - L(\theta_l) \); this bound is guaranteed to be tight at \( \theta = \theta_l \) (Dempster et al. 1977). Therefore, this algorithm is guaranteed to improve \( p(x|\theta) \) on each step that \( \theta \) changes.

In the semi-supervised learning setting (Nigam et al. 2006, 2000), we want to maximize the likelihoods of the observed labeled data \( (x_i, y_i) \in \mathcal{L} \) and unlabeled data \( x_i \in \mathcal{U} \), with the labels of the unlabeled data as hidden variables. Therefore, the soft EM objective in this context is:

\[
\max_{\theta} \sum_{i \in \mathcal{L}} \log p(x_i, y_i|\theta) + \lambda \sum_{i \in \mathcal{U}} \log E_{y_i|\theta}\{p(x_i|y_i;\theta)\}, \quad (2.3)
\]

where Nigam et al. (2000) suggested using \( \lambda \) to trade off the relative importance of the labeled and unlabeled data. The EM algorithm applied to this problem leads to a version of self-training with “soft” labels, where on iteration \( l \), the E- and M-step are as follows:
E: Given existing model $p(x, y\|\theta_l)$, compute $p(y_i = c|x_i, \theta_l)$ for all $x_i \in \mathcal{U}$ and all class labels $c$

$$M: \theta_{l+1} = \arg\max_{\theta} \sum_{i \in \mathcal{L}} \log p(x_i, y_i\|\theta) + \lambda \sum_{i \in \mathcal{U}} E_{y_i|x_i, \theta_l} \{ \log p(x_i, y_i|\theta) \}$$

Typically, $\theta_0$ is initialized on $\mathcal{L}$, e.g. by maximizing $\sum_{i \in \mathcal{L}} \log p(x_i, y_i|\theta)$. The exact implementation of the maximization step depends on the model, but note that with $\lambda = 1$, the M-step objective can be written as:

$$\sum_{i \in \mathcal{L}} \sum_{c \in \mathcal{Y}} p(y_i = c|x_i; \theta_l) \log p(x_i, y_i\|\theta),$$

where $p(y_i = c|x_i; \theta_l)$ is an indicator function for those examples with $y_i$ observed (i.e., examples in $\mathcal{L}$). Thus, the maximization step is equivalent to supervised training which includes training examples $(x_i, c)$ with weights $p(y_i = c|x_i; \theta_l)$ for all classes $c$ and all $x_i \in \mathcal{U}$.

We next describe the version using “hard” labels, which is sometimes called Hard EM.

**Classification EM**

The objective of classification EM is to maximize $\log p(x, z|\theta)$ over both the hidden variables $z$ and the parameters $\theta$. Thus the objective is different than in standard EM (McAllester 2007). However, the E and M-steps are very similar; the only difference is that in the E step we compute a hard assignment of hidden variable values, rather than a distribution.

In the domain adaptation context, the objective of hard EM is:

$$\max_{\theta, y_i \in \mathcal{U}} \sum_{i \in \mathcal{L}} \log p(x_i, y_i|\theta) + \sum_{i \in \mathcal{U}} \log p(x_i, y_i|\theta). \quad (2.4)$$

This corresponds to a version of self-training with hard labels on $\mathcal{U}$:

E: Given existing model $p(x, y\|\theta_l)$, compute $\hat{y}_i \equiv \arg\max_c p(y_i = c|x_i; \theta_l)$ for all $x_i \in \mathcal{U}$ and all class labels $c$

$$M: \theta_{l+1} = \arg\max_{\theta} \sum_{i \in \mathcal{L}} \log p(x_i, y_i\|\theta) + \sum_{i \in \mathcal{U}} \log p(x_i, \hat{y}_i|\theta)$$

As can be seen from comparing (2.3) and (2.4) the classification EM objective differs from the standard EM one in the second term computed over target domain examples: classification
EM tries to maximize \( p(x, y) \), whereas standard EM tries to maximize only \( p(x|\theta) \). From the perspective of generative modeling, \( p(x, y) \) appears more appropriate since it asserts that we want to model only “what we know” from the data. However, in most predictive scenarios we ultimately want to assign labels to \( D \) anyway, based on highest likelihood. Note also that only local optima of 2.3 and 2.4 are found, and from the perspective of matching the unseen labels, some local optima are better than others. If the estimated labels or label probabilities do not reflect well the true unseen labels, the optimum found may not reflect the true classes. So both approaches suffer from the problem that poor label or label probability estimates on one round can lead to a model that ultimately gives bad label estimates. Classification EM has the potential to propagate bad label estimates faster than standard EM. For example, if \( x_i \) is really in class 1 and \( p(y_i = 1|x_i; \theta_l) = 0.4 \), standard EM will still include \( y_i = 1 \) as a soft label in the next estimation step, but classification EM will not. However, classification EM is easier to implement with models that do not easily output probabilities or that output only the most likely hypotheses, such as HMMs used in speech recognition.

**EM for Discriminative Models**

We described above the EM algorithm and a hard-decision variant for modeling \( p(x, y|\theta) \) using labeled and unlabeled data. Related algorithms have also been used with discriminative modeling \( p(y|x; \theta) \). Amini and Gallinari (2002) described a hard-decision version for logistic regression which combines labeled and unlabeled data in order to optimize:

\[
\max_{\theta, y_i \in U} \sum_{i \in L} \log p(y_i|x_i; \theta) + \sum_{i \in U} \log p(y_i|x_i; \theta). \tag{2.5}
\]

Their algorithm is essentially the same as the classification EM algorithm above, except that it maximizes posterior label likelihoods \( p(y|x; \theta) \) in the M step.

Grandvalet and Bengio (2005) proposed a semi-supervised learning approach that incorporates unlabeled data into the likelihood objective as a “minimum entropy regularizer.” This results in optimizing:

\[
\max_{\theta} \sum_{i \in L} \log p(y_i|x_i; \theta) + \lambda \sum_{i \in U} \sum_{y_i \in C} p(y_i|x_i; \theta) \log p(y_i|x_i; \theta). \tag{2.6}
\]
The first term is the log likelihood of the labeled data while the second represents the empirical conditional entropy $H(Y|X)$ of the model on the unlabeled data. The motivation is that the unlabeled data should be useful only in cases when the classes are well separated, in which case the correct model should have low conditional entropy. The authors note that this resembles the EM objective (when $\lambda = 1$, it represents a “soft” version of Eqn. 2.5). However, the optimization procedure for Eqn. 2.6 is not described in detail.

In the literature, “bootstrapping”, or “self-training”, refers to variations on the objectives above, which seek to maximize likelihood of both labeled and unlabeled data. These are usually solved iteratively as in the EM algorithm, but may use a fixed number of iterations rather than convergence as the stopping criterion. Also, one might use only a subset of the examples to update the parameters in the maximization step—generally, the ones with the highest $p(\hat{y}_i|x_i; \theta_l)$, since these are the most confidently labeled samples according to the previous round’s model.

Self-training and EM for Domain Adaptation

In the domain adaptation setting, EM or classification EM can be applied with $\mathcal{L} = \mathcal{D}_s$ and $\mathcal{U} = \mathcal{D}_t$. This does not deal explicitly with the fact that $p_s(x, y) \neq p_t(x, y)$, but since it attempts to model both $\mathcal{D}_s$ and $\mathcal{D}_t$ simultaneously, we can hope that it will do a better job of generalizing to the target domain compared with just modeling $\mathcal{D}_s$. Ghahramani and Jordan (1995) discussed using the EM algorithm for learning in cases of missing data, including missing labels; they noted that if the missing data is of the missing-at-random (MAR) type—which describes the labels of the target data under covariate shift—then maximizing the likelihood of all observed data (including the ones with missing labels) can be done with EM and does not require modeling the missing data process. They described methods for training generative mixture model classifiers from labeled and unlabeled data related by MAR. Thus, this method might be useful for the covariate shift scenario, assuming mixtures were shared between domains. However, in other domain adaptation scenarios where these assumptions are not met, we generally do not want to model all the examples from both

---

3“Bootstrapping” as used in this context is not to be confused with the resampling procedure in statistics.
domains together, but only the target domain data. One heuristic for applying EM in this context is to alter the relative contributions of $D_t$ and $D_s$; in the method of Tan et al. (2009), the weight on the target data is increased at each iteration, while in the method of Dai et al. (2007b), the trade-off between the source and target data terms is determined by estimating KL divergence between the source and target distributions, with more weight on the target data as KL divergence increases. (The exact relationship is determined empirically over several source/target pairs). Some adaptation versions drop the $D_s$ term entirely, fitting the model only on $D_t$ but using $D_s$ to select the initial model $\theta_0$. For example, in the method of Saerens et al. (2002), the aim is to fit a generative model $p(x, y|\theta)$, where it is assumed that $p(x|y)$ is the same between source and target domains but that the class proportions $p(y)$ differ. EM is applied on $D_t$ only, where $D_s$ is used to estimate the initial $p_0(y)$, and the M-step updates $p(y)$ based on the (soft) class proportion counts in $D_t$ resulting from the model learned in step $l$. Thus, this method has the ability to adapt $p(y)$ from the source to the target domain. In the method of Pérez and Sánchez-Montañés (2007), EM is performed on the target data but with an additional term penalizing the distance between the new parameters and the source domain parameters. Ling et al. (2008c) applied a method based on the information bottleneck approach (Tishby et al. 1999) to a domain adaptation text classification problem. The goal was to categorize the unlabeled examples in order to maximize certain information theory objectives. In practice, the method has a similar iterative implementation as EM, and results in a generative distribution over features for each category.

One popular approach for language model adaptation for speech recognizers is to use recognized word sequences on the target data to adapt the language model trained initially on the source domain (Bacchiani and Roark 2003). This approach can be considered as the sequence modeling version of Saerens et al. (2002), since the language model is analogous to $p(y)$. Bacchiani and Roark (2003) described two general approaches for adapting the language model using the top-1 recognition hypotheses on the target domain. These approaches, count-merging and interpolation, are actually motivated by the maximum a posteriori (MAP) framework for supervised adaptation. In MAP adaptation, labeled source domain data is used to estimate a prior distribution over the parameters, and then one
chooses parameter values to maximize the posterior distribution over parameters given the target domain data:

\[ \theta^* = \arg\max_{\theta} p((x_i, y_i) \in D_t | \theta)p_s(\theta). \]

When target domain labels are not available, bootstrapped labels can be used, as in the work of Bacchiani and Roark (2003). In this case, MAP adaptation is related to the M-step in the hard version of self-training. For example, if:

\[ p_s(\theta) \propto p((x_i, y_i) \in D_s | \theta), \]

then, assuming \((x_i, y_i)\) represent independent samples (e.g., distinct sentences), we can write the objective as:

\[ \arg\max_{\theta} p((x_i, y_i) \in D_t | \theta)p_s(\theta) = \arg\max_{\theta} \sum_{i \in D_t} \log p(x_i | \theta) + \sum_{i \in D_s} \log p(x_i, y_i | \theta). \]

However, note that count-merging and interpolation represent slightly different computations of \(p_s(\theta)\) based on \(D_s\). The unsupervised MAP approach has also been applied to adapt probabilistic context-free grammar (PCFG) models used for parsing (Roark and Bacchiani 2003). Cache language modeling (Kuhn and De Mori 1990) may be considered a related domain adaptation method used in in speech recognition; it has also been applied to machine translation (Tiedemann 2010). It uses a changing language model which is an interpolation of a base model and a model built from nearby, earlier decoded sentences in the document.

Self-training methods have been applied to domain adaptation on many NLP tasks, including parsing (Roark and Bacchiani 2003; Sagae 2010; McClosky et al. 2006; Sagae and Tsujii 2007); part-of-speech tagging (Jiang and Zhai 2007a); conversation summarization (Sandu et al. 2010); entity recognition (Ciaramita and Chapelle 2010; Jiang and Zhai 2007b,a); sentiment classification (Tan et al. 2008); spam detection (Jiang and Zhai 2007a); cross-language document classification (Shi et al. 2010; Rigutini et al. 2005); and speech act classification (Jeong et al. 2009).

As noted above, self-training is closely related to the idea of entropy regularization (Grandvalet and Bengio 2005), and this idea has been applied to the domain adaptation setting as well. Zhuang et al. (2009) fit an inductive model with labeled source data, then
adjusted it based on certain “regularization” objectives on the target domain: manifold regularization (neighbors should have the same label); expectation regularization (class proportions should be maintained); and entropy minimization. This was applied to a document classification task. The proposed procedure of Rastrow et al. (2010) first learns a model $\theta_{\text{init}}$ on the labeled source data, then adjusts it in order to minimize an objective on the target data which includes entropy minimization and “entropy stability”, defined as the derivative of entropy with respect to $\theta$; it also includes a regularization term encouraging $\theta$ to be close to $\theta_{\text{init}}$. This was applied to estimate the interpolation weight in the language model for ASR.

We mention briefly that HMM models for speech recognition are traditionally trained to maximize likelihood using the EM algorithm for HMMs, which is called Baum-Welch. However, other algorithms, such as “extended Baum-Welch”, have been developed for training with a discriminative criterion rather than maximum likelihood. One such method was described by Huang and Hasegawa-Johnson (2008a), which can be used to fit a model to both labeled and unlabeled data. They applied their method to a simple phonetic classification problem, including a domain adaptation setting where the labeled and unlabeled speakers were different genders.

2.5.2 Co-training

Co-training (Blum and Mitchell 1998) is a semi-supervised learning method based on the idea of multi-view learning. It trains two classifiers based on different “views” (i.e., feature representations). Each classifier is used alternately to label new examples from the unlabeled pool. Confident examples from each classifier are then used to train the opposite classifier on the next round. Co-training has been used for domain adaptation, although, like self-training, it does not assume that $p_s(x, y) \neq p_t(x, y)$. For instance, Wan (2009) used it for cross-language sentiment classification; a machine translation system was used to derive versions of each document in both languages, representing two views. Another example is Wang (2009), who used co-training to adapt parsers trained on newswire to other genres (although a small amount of target data was used in addition to the source
data). Christoudias et al. (2006) proposed “co-adaptation,” which uses an out-of-domain training set to label seed target data used to build the initial model; this model is then improved via co-training. This was applied to two multi-modal (audio/visual) recognition tasks, agreement and phoneme classification, to adapt to new users and noise conditions.

2.5.3 Maximum Likelihood Linear Regression (MLLR)

MLLR is a method for speaker adaptation of acoustic models for speech recognition; it was first proposed by Leggetter and Woodland (1995) and Digalakis et al. (1995). MLLR adapts a Gaussian mixture HMM to new speaker data, which may be initially labeled or unlabeled; if unlabeled, it first uses the model to derive the most likely labels. It makes an explicit assumption about the form of the relationship between the distributions in the two domains: Gaussian mixture components, grouped into “regression classes”, are assumed to be related via linear transformation of the means and variances in each Gaussian. The transformation parameters, shared among all Gaussians in a class, are estimated to maximize likelihood on the new data, using the automatically derived labels if there are no hand labels available. The grouping of states into regression classes depends on the amount of data, so that with more data there can be more regression classes and more free parameters.

2.5.4 Assessment and Implications for Our Work

For unsupervised domain adaptation, an assumption behind both bootstrapping and co-training is that the initial source-trained model can label some target data both accurately and with high confidence. Furthermore, the target distribution needs to be amenable to bootstrapping or co-training from this initial data (which might not be representative of the target distribution). Standard EM (with “soft” training), used in the context of generative modeling, assumes that the source and target distributions share common latent generative variables (typically, clusters) whose class labels can be learned from the source domain. For discriminative self-training, such as Amini and Gallinari (2002) and Grandvalet and Bengio (2005), we believe these approaches work best when the classes in the target domain are well-separated from each other. (This is explicitly the goal of minimum entropy regularization in...
the work of Grandvalet and Bengio (2005).) However, if only the most confident examples are added on each round and if the iterations are stopped before using all the target domain data, then this might not be required.

For our tasks, we do not have knowledge of whether any of these assumptions is better than the others. However, we prefer to use methods with discriminative classifiers, because they are either standard or have been shown to work better than generative classifiers for the tasks we consider. Because we have a natural feature split (acoustic prosodic vs. textual) in Chapters 5 and 7, we experiment with co-training for those tasks. We also experiment with self-training, which provides a reasonable comparison with co-training using only a single classifier/feature view, and which is an extremely widely used, easily implemented adaptation method.

2.6 Methods Based on the Clustering/Manifold Assumption

Here we review domain adaptation approaches based on the clustering/manifold assumption. This assumption states that two data points are likely to have the same label if there is a high density path between them, or more formally, that \( p(x) \) should be low around the decision boundary \( \text{Gao et al. (2008)} \). Within the semi-supervised learning framework there exist many proposed methods based on this assumption. These methods often involve construction of a graph in which the labeled and unlabeled examples form the nodes, with the edge weights between pairs based on their similarity. The work on semi-supervised learning with graphs is too large to review here, but we mention a few classic algorithms. In the algorithm of \( \text{Zhu et al. (2003)} \), the labels from labeled samples propagate to unlabeled nodes based on the weights along the paths between them. In the “min-cut” method of \( \text{Blum and Chawla (2001)} \), binary classification is achieved by finding a cut in the graph that has the lowest cost based on the edge weights. Spectral clustering algorithms (e.g., \( \text{Ng et al. (2002)} \)) use the eigenvectors of the graph affinity matrix to cluster unlabeled data; this approach has been extended to include labeled examples (e.g., \( \text{Kamvar et al. (2003)} \)).

Graph-based semi-supervised methods are potentially very useful for domain adaptation because they use the density pattern of the unlabeled target data along with the labels from the source data. This idea is seen in the methods of \( \text{Xing et al. (2007)} \), \( \text{Ling et al.} \)
Ling et al. (2008a), Ling et al. (2008b), Zhuang et al. (2009), Hein (2009), Gao et al. (2008), and Alexandrescu and Kirchhoff (2009). Xing et al. (2007) proposed a method called “bridged refinement” that transfers class probability estimates from the source data, to the target data, via a “bridge” consisting of a mix of the data. First a generic classifier (such as SVM, Naive Bayes) is trained on the source data and applied to get initial confidence scores for the target data. Then, the confidence scores are adjusted to make them more “consistent” by applying a graph-based SSL algorithm similar to label propagation, whereby close neighbors are encouraged to have similar confidence scores (and assigned labels can change). This is actually done twice: the first pass adjusts the initial confidence scores from the classifier using all the source and target data mixed together, and the second pass adjusts those, using only the target data. Ling et al. (2008b) proposed a related approach that transfers label information among the $k$ nearest neighbors of each data point. Hein (2009) proposed “adaptive graph-based regularization,” which uses the unlabeled test data to construct a regularization term that penalizes prediction changes in dense regions. When the source/target relationship is described by sample selection bias, he proposed combining the regularization term with an instance-weighted empirical loss term on the training data.

The method of Ling et al. (2008a) is a modification of spectral classification (Kamvar et al. 2003) that seeks to find a cut of the graph that optimizes a function that is a combination of three terms: (1) a usual spectral clustering cost function based on the data similarity matrix (all the data together); (2) soft “must-link” constraints for the labeled source data; and (3) a spectral clustering cost function on the test data only. Gao et al. (2008) proposed a method motivated by ensemble learning; they assume there are multiple models trained on an out-of-domain data set or on several such sets. They wish to combine the out-of-domain models, weighting the model predictions “locally” based on how well each model corresponds with local partitions from clustering the target data. The idea is that some models are better suited to certain regions, which can be detected by correspondence with local cluster boundaries in that region. They combine this with a voting method within the clusters. As noted above, Zhuang et al. (2009) proposed learning an inductive model using “hybrid regularization” on the target domain, which included manifold regularization, entropy minimization, and expectation regularization (class proportions should
be maintained). Finally, “topic-bridged” probabilistic latent semantic analysis (Xue et al. 2008) applies an unsupervised learning method, pLSA, which decomposes the documents into latent topics under a maximum likelihood objective. The proposed objective of topic-bridged pLSA combines the pLSA objectives for the unlabeled data in each domain, and adds “must-link” and “cannot-link” terms for the source data based on their known labels. Although this is related to LSA, it is a transductive approach that resembles clustering so we include it in this section rather than in the change-of-feature-representation section.

Several of the domain adaptation papers described in this chapter have applied semi-supervised learning methods such as transductive SVMs (TSVMS) (Joachims 1999), spectral graph transducers (SGT) (Joachims 2003), and spectral classification (SC) (Kamvar et al. 2003) as baselines. These methods also make use of the cluster/manifold structure of the unlabeled target data. Xiang et al. (2010) noted that the assumption made by TSVMs—that the decision boundary lies in a low-density region—can be problematic in a domain adaptation scenario if the source and target data are themselves well separated. To solve this problem they proposed a method for selecting additional unlabeled data from a large collection such as Wikipedia, in order to fill in the “gap” between domains.

One of the challenges of graph-based and clustering methods in general is the construction of the pairwise similarity measure. (Of the above methods, that of Xue et al. (2008) is an exception because it does not compute pairwise similarities or distances.) Similarity measures may be designed for the task. Alexandrescu and Kirchhoff (2009) applied a graph-based method to rank machine translation hypotheses for sentences; the similarity measures for sentences are based on string kernels (Lodhi et al. 2002) or BLEU scores. A graph-based method for unsupervised word-sense disambiguation (to select translation candidates) was proposed by Yang and Kirchhoff (2010); the candidate translations of all content words in a document represent the nodes, with similarity measures computed from counts of common words in glosses obtained from Wikimedia resources. The methods of Xing et al. (2007), Ling et al. (2008a), Ling et al. (2008b), and Zhuang et al. (2009) are applied to document classification, and the standard cosine similarity is used as the document distance measure. In the case of continuous features, Gaussian kernels based on Euclidean distance between feature vectors are a standard choice for similarity measure. Euclidean distance assumes
that the given features are on comparable scales and that they are equally relevant to the task. Poor performance could result if the chosen feature representation contains many features that are irrelevant to the task, particularly if those features contain cluster-like structure.

2.6.1 Relationship between Cluster Methods and EM Methods

The EM methods reviewed in Section 2.5 can be viewed as semi-supervised clustering methods, since they assume clusters defined by the hidden variables. Like semi-supervised graph methods, they learn these clusters in a semi-supervised manner combining labeled and unlabeled data. However, the EM methods are based on parametric models, while the graph-based and clustering methods are based on pair-wise similarities represented in a graph or affinity matrix. In contrast with most of the covariate shift, self-labeling, and feature representation approaches, many of the graph-based methods are transductive approaches. They do not explicitly produce an adapted model that can be applied to new data, but rather derive labels for the test data as part of the adaptation process. (There are some exceptions, such as the method of Zhuang et al. (2009), which produces an inductive model.) In theory, once the labels for a small target set are derived transductively, they could be use to train an inductive model for the target domain, perhaps with training weights proportional to their label probabilities.

Nearest-neighbor methods, like graph-based methods, are non-parametric learning algorithms which use pair-wise similarities between data points. They do not attempt to find a cluster/manifold structure, but only assume that examples should have the same labels as their neighbors. Wang et al. (2008a) proposed that “example-based” learning such as nearest-neighbor and graph-based methods should be more robust to domain distribution differences than “inductive” methods which fit a model on the source domain, because “the training data that bear little similarity to a test query do not contribute much to the decisions.” In their experiments, a nearest-neighbor approach outperformed inductive methods on a text classification task across domains. Again, these methods rely on the availability of a good similarity measure.
2.6.2 Other Methods Based on Graphs and Clustering

The methods of Dai et al. (2009b), Chen et al. (2009), Pan et al. (2010), and He et al. (2009) use graphs with nodes representing features. In the methods of Dai et al. (2009b), Chen et al. (2009), and He et al. (2009), the graphs contain both example and feature nodes, with links between them based on co-occurrence. Another approach by Dai et al. (2007a) uses co-clustering of words and documents. These methods are intended for document classification tasks, and they differ from the graph-based and clustering methods described above. They do not explicitly use the cluster/manifold assumption, but aim to propagate label information via co-occurring features.

2.6.3 Our Work

We do not investigate cluster-based methods in our work; we expect such methods are less likely to work on our tasks, which are inherently very noisy and may contain many “clusters” unrelated to the labels of interest. In addition, most cluster methods require computation of a pairwise affinity matrix, which is sensitive to the affinity metric/feature space, as well as being computationally expensive. In spoken language processing applications, the choice of an affinity metric can be a challenging question in itself, particularly if the feature vector combines language and acoustic features.

2.7 Applications of Domain Adaptation

The unsupervised domain adaptation methods reviewed here have been applied to a wide variety of tasks. Tables 2.2 through 2.9 list a number of real domain adaptation tasks on which published methods have reported success. (It does not include experiments on artificially generated datasets or on real datasets with artificial sampling to create domain differences.) This is not an exhaustive list—in particular, there is a lot more work on the sentiment classification task, and on cross-language document classification—but it gives an idea of the breadth of applications.

Note that the 20Newsgroups/Reuters/SRAA tasks refer to classification of newsgroup postings based on top-level categories, where the source and target domains differ in the
subcategories they contain.

2.8 Conclusions

This chapter surveyed a large number of domain adaptation methods that have been proposed in the literature. In the following chapters, we implement a small selection of these methods for our tasks, including change-of-feature-representation approaches (feature restriction, LSA, and Blitzer et al.’s SCL); covariate shift instance weighting; co-training; and self-training. As described above, we chose these methods for a variety of reasons, including prior beliefs about what is most likely to work, ease of implementation, and interest. Self-labeling approaches are perhaps the most widely used for speech and language processing tasks; instance weighting and change-of-feature-representation approaches have been used less often, which motivates our research into these methods.
<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>part-of-speech tagging</td>
<td>WSJ vs. Medline</td>
<td>Blitzer et al. (2006)</td>
<td>SCL</td>
</tr>
<tr>
<td>part-of-speech tagging</td>
<td>WSJ vs. Medline</td>
<td>Huang and Yates (2009a, 2009)</td>
<td>latent variable HMM; contrastive estimation</td>
</tr>
<tr>
<td>NP chunking</td>
<td>WSJ vs. biochemistry journals</td>
<td>Huang and Yates (2009)</td>
<td>latent variable HMM; word-context-frequency vectors; LSA of word-context-frequency vectors</td>
</tr>
<tr>
<td>phrase chunking</td>
<td>WSJ (different sections)</td>
<td>Son et al. (2009)</td>
<td>instance weighting in SVM</td>
</tr>
<tr>
<td>parsing</td>
<td>WSJ vs. Brown</td>
<td>Roark and Bacchiani (2003); Sagae (2010); McClosky et al. (2006)</td>
<td>self-training</td>
</tr>
<tr>
<td>parsing</td>
<td>WSJ vs. various</td>
<td>Dredze et al. (2007)</td>
<td>parser diversity; instance weighting; feature scaling/restriction (poor results)</td>
</tr>
<tr>
<td>parsing</td>
<td>WSJ vs. various</td>
<td>Sagae and Tsujii (2007)</td>
<td>self-training</td>
</tr>
<tr>
<td>parsing</td>
<td>various</td>
<td>McClosky et al. (2010)</td>
<td>self-training, corpus/source weighting</td>
</tr>
<tr>
<td>parsing</td>
<td>Mandarin newswire vs. broadcast news, conversations</td>
<td>Wang (2009)</td>
<td>co-training (some labeled target)</td>
</tr>
</tbody>
</table>
Table 2.3: Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued).

<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>semantic role labeling</td>
<td>WSJ vs. Brown</td>
<td>Huang and Yates (2010b)</td>
<td>latent sequence variable HMM</td>
</tr>
<tr>
<td>word-sense disambiguation (WSD)</td>
<td>Brown vs. WSJ</td>
<td>Chan and Ng (2006)</td>
<td>EM prior adjustment</td>
</tr>
<tr>
<td>statistical machine translation (SMT)</td>
<td>read vs. spontaneous spoken domains, and read domain</td>
<td>Alexandrescu and Kirchhoff (2009)</td>
<td>graph-based propagation of ranking function over source/translation nodes</td>
</tr>
<tr>
<td>SMT and cross-language IR</td>
<td>different domains</td>
<td>Rogati (2009)</td>
<td>instance or corpus weighting</td>
</tr>
<tr>
<td>WSD for SMT</td>
<td>different domains</td>
<td>Yang and Kirchhoff (2010)</td>
<td>unsupervised graph-based WSD using all content words in target document</td>
</tr>
<tr>
<td>conversation summarization</td>
<td>AMI meetings vs. Enron email</td>
<td>Sandu et al. (2010)</td>
<td>SCL; self-training; supervised methods</td>
</tr>
<tr>
<td>opinion sentence extraction</td>
<td>different domains</td>
<td>Liu and Zhao (2008)</td>
<td>combo instance weighting, semi-supervised, labeled target</td>
</tr>
<tr>
<td>Japanese word segmentation</td>
<td>“daily conversation” example sentences vs. medical reference</td>
<td>Tsuboi et al. (2009)</td>
<td>instance weighting</td>
</tr>
<tr>
<td>ranking for document retrieval</td>
<td>adapt to unranked documents returned by test query</td>
<td>Duh and Kirchhoff (2011)</td>
<td>feature learning, instance weighting</td>
</tr>
</tbody>
</table>
Table 2.4: Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued).

<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>information extraction</td>
<td>employment ads in different industries</td>
<td>Chen et al. (2009)</td>
<td>subspace projection minimizing distribution difference and labeled data risk; TSVM</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups</td>
<td>Chen et al. (2009)</td>
<td>subspace projection minimizing distribution difference and labeled data risk; TSVM</td>
</tr>
<tr>
<td>document classification</td>
<td>Reuters</td>
<td>Pan et al. (2008, 2009)</td>
<td>low-dimensional transformation minimizing distribution difference</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups, SRAA, Reuters</td>
<td>Dai et al. (2007b)</td>
<td>Naive Bayes EM; TSVM</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups, SRAA, Reuters</td>
<td>Xing et al. (2007)</td>
<td>graph-based “refinement” of classifier labels; TSVM</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups, SRAA, Reuters</td>
<td>Dai et al. (2007a)</td>
<td>label propagation via word-document co-clustering</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups</td>
<td>He et al. (2009)</td>
<td>example-feature graph (some labeled target)</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups, Reuters</td>
<td>Gao et al. (2008)</td>
<td>combo ensemble classifiers/clustering</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups, SRAA, Reuters</td>
<td>Ling et al. (2008b)</td>
<td>graph-based label transfer; TSVM</td>
</tr>
<tr>
<td>document classification</td>
<td>20 Newsgroups</td>
<td>Ji et al. (2011)</td>
<td>SCL variants</td>
</tr>
</tbody>
</table>
Table 2.5: Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued).

<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>document classification</td>
<td>20 Newsgroups, SRAA, Reuters</td>
<td>Ling et al. (2008a)</td>
<td>graph-based with multi-term objective; TSVM; SC; SGT; instance weighting with KDE</td>
</tr>
<tr>
<td>web query classification</td>
<td>different domains</td>
<td>Xiang et al. (2010)</td>
<td>TSVM with addition of unlabeled Wikipedia data</td>
</tr>
<tr>
<td>job categorization</td>
<td>job listings vs. queries</td>
<td>Wang et al. (2008a)</td>
<td>nearest-neighbor methods</td>
</tr>
<tr>
<td>research paper topic</td>
<td>different time periods</td>
<td>Bickel et al. (2007)</td>
<td>instance weighting</td>
</tr>
</tbody>
</table>
Table 2.6: Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued).

<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>named-entity recognition</td>
<td>different domains</td>
<td>Guo et al. (2009)</td>
<td>latent semantic assoc. based on latent Dirichlet allocation</td>
</tr>
<tr>
<td>named-entity recognition</td>
<td>Reuters (CoNLL) vs. BBN</td>
<td>Ciaramita and Chapelle (2010)</td>
<td>LSA/SVD; SCL; self-training; prior adjustment</td>
</tr>
<tr>
<td>entity extraction</td>
<td>citation databases, CoNLL</td>
<td>Satpal and Sarawagi (2007)</td>
<td>feature weight penalization based on source/target domain difference</td>
</tr>
<tr>
<td>gene/protein name recognition</td>
<td>documents on different organisms</td>
<td>Jiang and Zhai (2007b)</td>
<td>feature weight penalization based on generalization across multiple source domains, self-training on target</td>
</tr>
<tr>
<td>protein name extraction</td>
<td>different abstract collections</td>
<td>Arnold et al. (2007)</td>
<td>feature scaling to match distribution means; TSVM; EM</td>
</tr>
<tr>
<td>entity-type classification</td>
<td>newswire vs. weblog, CTS</td>
<td>Jiang and Zhai (2007a)</td>
<td>self-training</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Blitzer et al. (2007)</td>
<td>SCL</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Tan and Cheng (2009); Ji et al. (2011)</td>
<td>modified versions of SCL</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Blitzer et al. (2009)</td>
<td>SCL/CCA/subspace projection</td>
</tr>
</tbody>
</table>
Table 2.7: Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued).

<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Aue and Gamon (2005)</td>
<td>restrict features</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Tan et al. (2009)</td>
<td>shared features plus self-training</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Tan et al. (2008)</td>
<td>self-training</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>He et al. (2009)</td>
<td>example-feature graph</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Pan et al. (2010)</td>
<td>LSA/SVD; SCL; feature clustering via feature co-occurrence graph</td>
</tr>
<tr>
<td>sentiment classification</td>
<td>different domains</td>
<td>Xiang et al. (2010)</td>
<td>TSVM with addition of unlabeled Wikipedia data</td>
</tr>
<tr>
<td>cross-language sentiment classification</td>
<td>different languages</td>
<td>Wan (2009)</td>
<td>co-training with MT (different views = different languages)</td>
</tr>
<tr>
<td>cross-language sentiment classification</td>
<td>different languages</td>
<td>Wei and Pal (2010); Prettenhofer and Stein (2010)</td>
<td>SCL with MT/lexicon</td>
</tr>
<tr>
<td>cross-language topic classification</td>
<td>different languages</td>
<td>Shi et al. (2010)</td>
<td>self-training with lexicon</td>
</tr>
<tr>
<td>cross-language topic classification</td>
<td>different languages</td>
<td>Ling et al. (2008c)</td>
<td>information bottleneck with MT</td>
</tr>
<tr>
<td>cross-language topic classification</td>
<td>webpages in English vs. Chinese</td>
<td>Rigutini et al. (2005)</td>
<td>self-training with MT</td>
</tr>
</tbody>
</table>
Table 2.8: Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued).

<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>email spam detection</td>
<td>different times</td>
<td>Kolcz and Teo (2009)</td>
<td>feature scaling</td>
</tr>
<tr>
<td></td>
<td>public pool vs. individual user</td>
<td>Gao et al. (2008)</td>
<td>combo ensemble classifiers/clustering</td>
</tr>
<tr>
<td></td>
<td>public pool vs. individual user</td>
<td>Bickel and Scheffer (2007); Bickel et al. (2007)</td>
<td>instance weighting</td>
</tr>
<tr>
<td>email spam detection</td>
<td>public pool vs. individual user</td>
<td>Jiang and Zhai (2007a)</td>
<td>self-training</td>
</tr>
<tr>
<td>age estimation in face images</td>
<td>different lighting conditions</td>
<td>Ueki et al. (2010)</td>
<td>instance weighting</td>
</tr>
<tr>
<td>TCP intrusion detection</td>
<td>different intrusion types</td>
<td>Gao et al. (2008)</td>
<td>combo ensemble classifiers/clustering; TSVM</td>
</tr>
<tr>
<td>TCP intrusion detection</td>
<td>different intrusion types</td>
<td>He et al. (2009)</td>
<td>example-feature graph</td>
</tr>
<tr>
<td>brain-computer interface signals</td>
<td>different sessions</td>
<td>Sugiyama et al. (2007)</td>
<td>instance weighting</td>
</tr>
<tr>
<td>land mine detection</td>
<td>different regions</td>
<td>Bickel et al. (2007)</td>
<td>instance weighting</td>
</tr>
<tr>
<td>heart disease prediction</td>
<td>different hospitals</td>
<td>Pérez and Sánchez-Montaños (2007)</td>
<td>EM with source parameter distance penalty</td>
</tr>
<tr>
<td>WiFi localization</td>
<td>different times</td>
<td>Pan et al. (2008, 2009)</td>
<td>low-dimensional transformation minimizing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>distribution difference; TSVM</td>
</tr>
</tbody>
</table>
Table 2.9: Summary of tasks used for experiments in published work on unsupervised domain adaptation (continued).

<table>
<thead>
<tr>
<th>task</th>
<th>domains</th>
<th>references</th>
<th>approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>audio-visual agreement and phoneme classification</td>
<td>noisy conditions and/or new speaker</td>
<td>Christoudias et al. (2006)</td>
<td>co-training</td>
</tr>
<tr>
<td>speaker identification</td>
<td>different text, different recording times</td>
<td>Yamada et al. (2010)</td>
<td>instance weighting</td>
</tr>
<tr>
<td>phonetic classification</td>
<td>male vs. female speakers</td>
<td>Huang and Hasegawa-Johnson (2008a)</td>
<td>semi-supervised modeling</td>
</tr>
<tr>
<td>acoustic model for ASR</td>
<td>speaker-independent vs. speaker specific</td>
<td>Leggetter and Woodland (1995), Digalakis et al. (1995)</td>
<td>MLLR</td>
</tr>
<tr>
<td>LM for ASR</td>
<td>voicemail transcription vs. customer care</td>
<td>Bacchiani and Roark (2003)</td>
<td>self-training</td>
</tr>
<tr>
<td>LM for ASR</td>
<td>broadcast news vs. conversations</td>
<td>Wang and Stolcke (2007)</td>
<td>self-training</td>
</tr>
<tr>
<td>LM for ASR</td>
<td>broadcast news vs. lectures</td>
<td>Rastrow et al. (2010)</td>
<td>self-training based on entropy regularization</td>
</tr>
<tr>
<td>LM, translation model for SMT</td>
<td>Europarl vs. EMEA (medical)</td>
<td>Tiedemann (2010)</td>
<td>self-training with cache</td>
</tr>
<tr>
<td>speech act classification</td>
<td>SWBD/MRDA vs. emails/forums</td>
<td>Jeong et al. (2009)</td>
<td>self-training; semi-supervised boosting with sophisticated feature “subtrees”</td>
</tr>
<tr>
<td>prosodic phrase break and accent labeling</td>
<td>different corpora, speakers</td>
<td>Fernandez and Ramabhadradas (2010)</td>
<td>self-training</td>
</tr>
</tbody>
</table>
Chapter 3

EXPERIMENTS WITH FEATURE REPRESENTATION METHODS FOR DOMAIN ADAPTATION

3.1 Introduction

In this chapter we investigate a selection of “feature representation” approaches, including SCL, LSA/PCA, and feature restriction. We provide some analysis of each method and conduct comparative experiments on two synthetic and two real datasets. The synthetic datasets permit us to vary certain properties of the data while controlling others. The real datasets allow us to investigate the extent to which the methods can work on real domain adaptation tasks. These datasets consist of sentiment classification and topic classification tasks, and have been used in previous literature on domain adaptation. In Chapter 4, we investigate instance weighting and conduct additional experiments with that method on the same two datasets.

This chapter is organized as follows. We first analyze feature restriction, PCA/LSA, and SCL. Following that, we conduct comparative experiments among the methods using synthetic datasets. Section 3.6.1 introduces the two document classification datasets that we use in this chapter and in Chapter 4, followed by comparative experiments and results on those datasets. Finally, we consider a method for automatic SCL and LSA feature selection, which is inspired by the error bounds of Ben-David et al. (2007).

3.2 Feature Restriction

One simple adaptation strategy for NLP tasks has been the feature restriction approach, where word or n-gram features are restricted to those found in the target domain (Aue and Gamon 2005; Dredze et al. 2007). We experiment with this approach for dialog act tagging in Chapter 6. The idea behind this strategy is roughly that certain features are “domain-specific,” and that we should not use the value of source-specific features to make
decisions on the target data. In practice this strategy has shown mixed results—in some cases transfer performance improves while in other cases it degrades or has no effect. Aue and Gamon (2005) argued that the feature restriction approach makes sense under the assumption that features appearing in the target domain have the same correlations with class labels in both domains, noting that the strategy “allows the classification algorithm to make the best possible use of the out-of-domain data since it need not take into account features that never appear in the target domain.” In this section we seek to formalize the assumptions behind this approach.

Our starting idea is the existence of a “single model” that can describe both the source and target domains simultaneously. This is a common idea behind many published approaches to domain adaptation. It is sometimes stated as \( p_s(y|x) = p_t(y|x) \) for \( x \) in the region of feature space that is shared by both domains. In addition, for classification problems it is common to assume that \( p_s(y|x) \) and \( p_t(y|x) \) belong to a simple class of functions, specifically those whose decision boundaries can be modeled simultaneously by the chosen classifier. Blitzer et al. (2009) makes the assumption that there exists a single linear function for source and target domain that describes the expected value of \( Y \) given the features \( X \):

\[
E\{Y|X; D=\text{source}\} = E\{Y|X; D=\text{target}\} = \beta \cdot X.
\]

In other words, there exists a \( \beta \) that is “independent” of domain \( D \). That work considered three subspaces of \( X \): the subspace occupied by the source domain feature vectors; that occupied by the target domain features vectors, and their intersection subspace, which is referred to as the “shared subspace.” Our work here is inspired by this assumption from Blitzer et al. (2009), but rather than subspaces, we consider the feature vector \( X \) to be composed of three sets of features: those that are only present in the source domain \( X_s \), only present in the target domain \( X_t \), or present in both \( X_b \). In other words, we assume that \( X_s = 0 \) for all target domain examples and \( X_t = 0 \) for all source domain examples. For example, the features might represent counts of vocabulary words, where the words in \( X_s, X_t \) are domain-specific terms.

We consider a binary classification problem \( Y \in \{-1,1\} \). Here we use an explicit
notation where $X$ represents a vector of random variables and $x$ is a particular value of that random vector. We will write $X = [X_s, X_b, X_t]$ and $x = [x_s, x_b, x_t]$. Like Blitzer et al. (2009), we assume there exists a “domain-general” model $p(Y = 1|X = x)$ that corresponds with $p_s(Y = 1|X = x)$ and $p_t(Y = 1|X = x)$ at any $x$ where these distributions are defined. (This assumption is the same as the covariate shift assumption that $p_s(y|x) = p_t(y|x)$, except we do not assume support of $p_t(y|x)$ is within that of $p_s(y|x)$.)

3.2.1 Feature Restriction for a Linear Model

Consider a linear model $[\beta, \beta_0]$, where:

$$p(Y = 1|X = x) = g(\beta \cdot x + \beta_0)$$

for a monotonic $g(\cdot)$ and some $[\beta, \beta_0]$, such that the decision boundary occurs at $\beta \cdot x + \beta_0 = 0$.

This includes the logistic regression model which we use in many of the later experiments. In this section we ignore the effect of model misspecification and assume a true linear model underlying the data. We assume $p_s(Y|X = x)$ and $p_t(Y|X = x)$ can be described by the same linear model defined by $[\beta, \beta_0]$, and assume first that $X$ is not linearly dependent (so there is a unique $[\beta, \beta_0]$). Since $X = [X_s, X_b, X_t]$, we can write:

$$\beta \cdot x + \beta_0 = \beta_s \cdot x_s + \beta_b \cdot x_b + \beta_t \cdot x_t + \beta_0.$$ 

Consider trying to estimate $\beta = [\beta_s, \beta_b, \beta_t, \beta_0]$ on the source domain. We are unable to estimate $\beta_t$, because $X_t = 0$ for those samples. The only feature that is shared between the source and target domains is $X_b$, so without resorting to a feature-learning method such as SCL, we must use the source data to try to learn a model for $p_t(Y = 1|X_b = x_b)$, i.e., the dependence of $Y$ on $X_b$ in the target domain, when $X_t$ is unobserved. Note that, in general:

$$p_t(Y = 1|X_b = x_b) = \int u g(\beta_b \cdot x_b + \beta_0 + \beta_t \cdot u) f_{X_t}(u|X_b = x_b) du,$$

where $f_{X_t}(u|X_b = x_b)$ is the conditional distribution of $X_t$ given $X_b$ on the target domain.

We consider two models that can be learned on the source data, the “full source-domain model,” trained using all available features:

$$p_s(Y = 1|X_s = x_s, X_b = x_b) = g(\beta_s \cdot x_s + \beta_b \cdot x_b + \beta_0),$$
or the “shared-only source-domain model,” trained using only the shared features $X_b$:

$$p_s(Y = 1|X_b = x_b) = \int_u g(\beta_s \cdot u + \beta_b \cdot x_b + \beta_0) f_{X_s}(u|X_b = x_b) du,$$

where $f_{X_s}(u|X_b = x_b)$ is the conditional distribution of $X_s$ given $X_b$ on the source domain.

When evaluated on the target domain, the full source-domain model gives:

$$p_s(Y = 1|X_s = 0, X_b = x_b) = g(\beta_b \cdot x_b + \beta_0) = p_t(Y = 1|X_b = x_b, X_t = 0).$$

It will give the correct values for $p_t(Y = 1|X_b = x_b)$ only if:

$$p_t(Y = 1|X_b = x_b, X_t = 0) = p_t(Y = 1|X_b = x_b). \quad (3.1)$$

On the other hand, the shared-only source-domain model will be correct only if:

$$p_s(Y = 1|X_b = x_b) = p_t(Y = 1|X_b = x_b),$$

which is true if:

$$\int_u g(\beta_s \cdot u + \beta_b \cdot x_b + \beta_0) f_{X_s}(u|X_b = x_b) du = \int_u g(\beta_b \cdot x_b + \beta_0 + \beta_t \cdot u) f_{X_t}(u|X_b = x_b) du. \quad (3.2)$$

There are a number of conditions under which Eqn. 3.2 will approximately hold, but we focus on one condition where:

$$\beta_s \approx \beta_t \quad \text{and} \quad f_{X_s}(u|X_b = x_b) \approx f_{X_t}(u|X_b = x_b), \forall u, x_b. \quad (3.3)$$

We call this condition “positive correspondence”: the feature $X_t$ in the target domain corresponds to $X_s$ in the source domain in the sense that it has both the same relationship to the shared feature $X_b$ (specifically, the conditional distributions given $X_b$ are the same) and the same relationship to the target variable $Y$ (specifically, $\beta_s = \beta_t$). By contrast, “negative correspondence” can also occur, where $X_t$ and $X_s$ have the same relationships to $X_b$ but $\beta_s = -\beta_t$. In this case, the shared-only source-domain model may give very bad predictions on the target domain.

In most cases we do not actually care if $p_s(Y = 1|X_b = x_b) = p_t(Y = 1|X_b = x_b)$ as much as we care if the decision regions are the same (based on a decision boundary where
\( p(Y = 1|x) = 0.5 \). Note that even if the decision boundary on all features \( X \) is assumed to be linear, the decision boundary on a subset of features might not be. However, to understand the problem better we consider a specific case where the decision boundaries on both the full model and the restricted model are linear. Assume three variables \( X_s, X_b, X_t \), where \( X_s \sim N(0, \sigma_s) \) in the source domain, \( X_t \sim N(0, \sigma_t) \) in the target domain, and:

\[
X_b = \alpha_s X_s + \alpha_t X_t + N
\]

\[
Y = \text{sign}(\beta_s X_s + \beta_b X_b + \beta_t X_t + \beta_0 + \epsilon),
\]

where \( \epsilon \sim N(0, \sigma_\epsilon) \) and \( N \sim N(0, \sigma_N) \); \( \epsilon \) and \( N \) are independent. The decision boundary over all features \( X_s, X_b, X_t \) occurs at \( x_s, x_b, x_t \) such that:

\[
p(\beta_s X_s + \beta_b X_b + \beta_t X_t + \beta_0 + \epsilon > 0|X_s = x_s, X_b = x_b, X_t = x_t) = 0.5
\]

\[
\implies \beta_s x_s + \beta_b x_b + \beta_t x_t + \beta_0 = 0.
\]

For the full source-domain model, the decision boundary occurs at all values of \( x_s, x_b \) that satisfy:

\[
\beta_s x_s + \beta_b x_b + \beta_0 = 0.
\]

For the shared-only source-domain model, the decision boundary occurs at the value of \( x_b \) that satisfies:

\[
p_s(Y = 1|X_b = x_b) = p_s(\beta_s X_s + \beta_b x_b + \beta_0 + \epsilon > 0|X_b = x_b) = 0.5.
\]

\[
(3.4)
\]

The conditional random variable \( X_s|X_b = x_b \) is normally distributed in the source domain, with mean \( \rho x_b \):

\[
\rho_s \equiv \frac{\alpha_s \sigma_s^2}{\alpha_s^2 \sigma_s^2 + \alpha_N^2}.
\]

Therefore, \( \beta_s X_s + \beta_b x_b + \beta_0 + \epsilon \) is a normally distributed random variable with mean \( \beta_s \rho x_b + \beta_b x_b + \beta_0 \) given \( X_b = x_b \), and so Eqn. \( 3.4 \) will be satisfied when:

\[
(\beta_s \rho s + \beta_b) x_b + \beta_0 = 0.
\]

The true shared-only target-domain model has the decision boundary:

\[
(\beta_t \rho t + \beta_b) x_b + \beta_0 = 0,
\]
where:

$$\rho_t \equiv \frac{\alpha_t \sigma_t^2}{\alpha_t^2 \sigma_t^2 + \alpha_t^2 N}.$$ 

Therefore, the shared-only decision boundary will transfer correctly under any condition where $\beta_s \rho_s = \beta_t \rho_t$. For example, under the “positive correspondence” condition, $\rho_s = \rho_t$, and the shared-only decision boundary will transfer correctly if $\beta_s = \beta_t$. It would also transfer correctly if $\beta_s = -\beta_t$ and $\rho_s = -\rho_t$. By contrast, any condition where $\beta_s \rho_s = -\beta_t \rho_t$ would cause the decision boundary to transfer poorly—in the case where $\beta_b = 0$, the source domain coefficient on $x_b$ gives the wrong sign for the target domain.

**Example**

We consider a simple version of the linear model described above with $\rho_t = \rho_s$ and illustrate four cases with different values of $[\beta_s, \beta_b, \beta_t]$ in Figures 3.1 through 3.5. In all cases, we assume there are three features, $X_s, X_b, X_t$, whose distributions differ between domains. On the source domain:

$$X_s \sim N(0, 0.5)$$

$$X_t = 0,$$

while on the target domain:

$$X_s = 0$$

$$X_t \sim N(0, 0.5).$$

On both domains:

$$X_b = X_s + X_t + N$$

$$N \sim N(0, 0.25).$$

Finally, $Y \equiv \text{sign}(\beta_s \cdot X_s + \beta_b \cdot X_b + \beta_t \cdot X_t + \beta_0 + \epsilon)$, where $\epsilon \sim N(0, 0.25)$ and $\beta_0 = -0.5$ in all cases. Note that $\rho_t = \rho_s = 0.8$.

Case 1. No dependence on $X_t$ in the full model, so $\beta_t = 0$ on both domains (Figure 3.1). This occurs when there are no target-domain-only features, or when those features are not informative given the shared features. We would expect the full source-domain model
to do better than the shared-only source-domain model, because the full source-domain model is identical to the full model over both domains. In the example in Figure 3.1, $[\beta_s, \beta_b, \beta_t] = [1, 1, 0]$. The full source-domain model decision boundary (shown as a solid line) occurs at:

$$X_s + X_b - 0.5 = 0,$$

which intersects $X_s = 0$ at $X_b = 0.5$. The shared-only source-domain decision boundary (shown as a dashed line) occurs at $X_b = 0.278$. The full source-domain model performs better on the target data than the source shared-only model; in fact, the full source-domain model is the correct model on the target domain as well, in the sense that Eqn. 3.1 is satisfied.

Case 2. No dependence on $X_s$ in the full model (Figure 3.2). This is the opposite of Case 1; $\beta_s = 0$ on both domains so there is no difference between the full and shared-only source-domain models. In the example in Figure 3.2, $[\beta_s, \beta_b, \beta_t] = [0, 1, 1]$. The single decision boundary for both models occurs at $X_b = 0.5$. (Note, of course, that if we had access to target domain data we could build a more accurate model involving $X_t$.)

Case 3. Positive correspondence between $X_s, X_t$. This is the case where the shared-only source-domain model transfers better than the full source-domain model. We illustrate two versions in Figures 3.3 and 3.4. In Figure 3.3, $[\beta_s, \beta_b, \beta_t] = [1, 0, 1]$. Since there is no dependence on $X_b$ given $X_s$ in the source domain, the decision boundary occurs at $X_s = 0.5$ in the full source-domain model and so all the target domain data (having $X_s = 0$) lies to one side of the decision boundary. For the shared-only source-domain model, the decision boundary occurs at $X_b = 0.625$, which splits the target domain data and gives higher accuracy in this case. In Figure 3.4, $[\beta_s, \beta_b, \beta_t] = [1, 1, 1]$. Like in Case 1, the decision boundary for the shared-only model occurs at $X_b = 0.278$, while for the full model, it occurs at $X_s + X_b = 0.5$. However, unlike Case 1, the decision boundary for the shared-only source-domain model transfers better to the target domain, because $\beta_s = \beta_t$.

Case 4. Negative correspondence between $X_s, X_t$. This is the opposite of Case 3, and is
illustrated in Figure 3.5 where \([\beta_s, \beta_b, \beta_t] = [1, 0, -1]\). In this case neither Eqn. 3.1 nor Eqn. 3.2 is satisfied, but the shared-only source-domain model transfers particularly badly to the target domain because the predictions implied by the decision boundary have the wrong sign.

Table 3.1: Error rates (%) for the full and shared-only models for the source domain evaluated on the target domain, for each of the cases depicted in Figure 3.1 through 3.5

<table>
<thead>
<tr>
<th></th>
<th>full</th>
<th>shared-only</th>
</tr>
</thead>
<tbody>
<tr>
<td>case 1</td>
<td>8.3</td>
<td>11.8</td>
</tr>
<tr>
<td>case 2</td>
<td>13.8</td>
<td>13.8</td>
</tr>
<tr>
<td>case 3a</td>
<td>18.6</td>
<td>13.7</td>
</tr>
<tr>
<td>case 3b</td>
<td>13.8</td>
<td>9.5</td>
</tr>
<tr>
<td>case 4</td>
<td>18.3</td>
<td>32.2</td>
</tr>
</tbody>
</table>

Table 3.1 shows the transfer error (i.e., the error on the target data) for the full and shared-only source-domain models for each of the four cases illustrated. (Decision boundaries are true source domain boundaries rather than estimated ones from the data). As can be seen from this table, only Case 3 (positive correspondence) benefits from the feature restriction approach. Case 4 (negative correspondence) shows a very poor transfer situation where the best we can do is to classify all target domain examples as the same class. Since the scenarios differ only in the values of \([\beta_s, \beta_b, \beta_t]\), it is not possible to know which scenario is occurring based only on unlabeled target data; it requires an additional assumption about the relationship between domains.

Note that the above example used positive/negative correspondence conditions to illustrate cases where the source-domain shared-only model transfers well or badly to the target domain. However, we emphasize that there are other conditions where the same effect would be observed; what matters is really whether \(\beta_s\rho_s = \beta_t\rho_t\). For example, if \(X_b = X_s - X_t + N\), then the shared-only model would transfer well if \([\beta_s, \beta_b, \beta_t] = [1, 0, -1]\) or \([-1, 0, 1]\), for example, and would transfer poorly if \([\beta_s, \beta_b, \beta_t] = [1, 0, 1]\) or \([-1, 0, -1]\).
Figure 3.1: $Y = \text{sign}(X_s + X_b + 0.5 + \epsilon)$, for $X_s, X_b, X_t$ as described above. Magenta and cyan data points represent the two classes in the target domain, corresponding to red and blue in the source domain. Lower plot shows the target distribution for $X_b, X_t$; upper plot shows the source distribution for $X_s, X_b$ as well as the target distribution projected onto that space. The full source domain model (solid decision boundary) does better than the shared-only source domain model (dashed decision boundary).
case 2: no dependence on $X_s$ in the full model

Figure 3.2: $Y = \text{sign}(X_b + X_t + 0.5 + \epsilon)$. There is no difference between the full source domain model and the shared-only source domain model.
case 3a: positive correspondence between $X_s$, $X_t$ (with no dependence on $X_b$).

Figure 3.3: $Y = \text{sign}(X_s + X_t + 0.5 + \epsilon)$. Positive correspondence condition: the shared-only source domain model (dashed) does better than the full source domain model (solid).
Figure 3.4: \[ Y = \text{sign}(X_s + X_b + X_t + 0.5 + \epsilon). \] Positive correspondence condition: the shared-only source domain model (dashed) does better than the full source domain model (solid).
Figure 3.5: $Y = \text{sign}(X_s - X_t + \epsilon)$. Negative correspondence condition: both models do poorly on the target domain, but the full source domain model (solid) does better than the shared-only model (which has the wrong sign for $\beta_b$).
3.2.2 Linear Dependence

In Section 3.2.1 we assumed that the features in $X$ were all linearly independent. However, in a high dimensional space it is likely that this is not the case: we may have some features that are nearly perfectly predicted by others, e.g., $X_3 = k_1 X_1 + k_2 X_2$. In this case, there are an infinite number of linear models that have the same $p(Y = 1|X = x)$. Consider a case where $X_s = A \cdot X_b$, where $A$ is a matrix of coefficients. Then two of the source-domain linear models are given by:

$$p_s(Y = 1|X = x) = g((\beta_s \cdot A + \beta_b) \cdot x_b + \beta_0).$$

The latter model is learned on the shared features only, and would be preferred if it corresponds to the domain-general model. Consider cases 1 and 3b in the example in Section 3.2.1 but let $N = 0$ so $X_s, X_b$ are perfectly correlated on the source domain and $X_t, X_b$ are perfectly correlated on the target domain. On the source domain, the following models are equivalent:

$$p(Y = 1|X = x) = g(X_s + X_b + 0.5)$$
$$p(Y = 1|X = x) = g(2X_b + 0.5), \tag{3.5}$$

but these are not equivalent on the target domain, and an additional assumption is required about which one transfers better. If $\beta_t = 0$ (case 1) then the first model is correct, while if $\beta_t = 1$ (case 3b) then the second model is correct.

3.2.3 Sparse Features

Consider the case where $X_s, X_b, X_t$ represent counts of rare words, such that $p_s(X_s = 0|X_b = x_b)$ is very close to 1. Then:

$$p_s(Y = 1|X_b = x_b) = g(\beta_s + \beta_b \cdot x_b + \beta_0) \cdot p_s(X_s = 1|X_b = x_b)$$
$$+ g(\beta_b \cdot x_b + \beta_0) \cdot p_s(X_s = 0|X_b = x_b)$$
$$\approx g(\beta_b \cdot x_b + \beta_0).$$

Therefore:

$$p_s(Y = 1|X_b = x_b) \approx p_t(Y = 1|X_b = x_b, X_s = 0)$$
$$p_s(Y = 1|X_b = x_b) \approx p_t(Y = 1|X_b = x_b).$$
There is little difference between the full and shared-only source-domain models, and the estimated model should be close to the shared-only target-domain model. Thus we would expect the feature restriction approach would have little effect.

### 3.2.4 Model Fitting and Other Linear Classifiers

In the example above, the true model was assumed to be linear and known for both \( p(Y = 1|X = x) \) and \( p(Y = 1|X_b = x_b) \). However, in practice the true function may not be linear. Even if \( p(Y = 1|X = x) \) is linear, \( p_s(Y = 1|X_b = x_b) \) or \( p_t(Y = 1|X_b = x_b) \) may not be. And in any case, even if the conditional probabilities are not linear in \( x \), the decision boundary could still be. In practice, also, we fit a boundary to minimize an empirical loss, so we may not model \( p(Y = 1|X = x) \) at all. For example, logistic regression uses maximum likelihood loss so it does model \( p(Y = 1|X = x) \), but least-squares regression and SVMs do not model \( p(Y = 1|X = x) \). These procedures can generally produce different linear decision boundaries. However, as long as the coefficients in the model are learned jointly, the coefficients \( [\beta'_b, \beta'_0] \) for a model containing only \( X_b \) is not generally the same as the coefficients \( [\beta''_b, \beta''_0] \) when additional features \( X_s \) are included. In general we expect that \( \beta'_b, \beta'_0 \) will be influenced by the correlation of \( X_b \) and \( X_s \) in the source domain, and by the label dependence on \( X_s \) in the source domain.

Naive Bayes classifiers have a linear decision boundary, but the coefficients on each feature are learned independently and there is no difference between the shared-only and full source-domain models when applied to the target domain. This is true whenever Naive Bayes is used to learn a model, whether or not the model fits the true distribution.

Like Naive Bayes, linear discriminant analysis fits the coefficients for each feature in a linear decision boundary independently. However, feature restriction changes the coefficient \( \beta_0 \) since this includes the sum of means for each feature.

### 3.2.5 Biased Sampling Assumption vs. Corrupted Features Assumption

Ultimately, the feature restriction approach requires an assumption about whether \( p_s(Y|X_s = 0, X_b = x_b) \) vs. \( p_s(Y|X_b = x_b) \) is a better model for \( p_t(Y|X_b = x_b) \). The analysis and illustra-
tion above started from the assumption of a single domain-general linear model \( p(Y|X = x) \), and showed that \( p_s(Y|X_b = x_b) \) is a better model on the target domain under the positive correspondence assumption. However, we can also motivate the analysis from the perspective of the “biased sampling” assumption vs. “corrupted features” assumption.

Under the biased feature assumption, the target domain is sampled in a biased way from the source domain such that \( X_s = 0 \), but \( p_s(Y|X_s = 0, X_b = x_b) \) remains the correct model for \( p_t(Y|X_b = x_b) \). An example of this is case 1 above: the target data is in a subspace of the source domain. Under the biased feature assumption, we prefer to learn on all available features.

Under the corrupted feature assumption, \( X_s \) has been lost or corrupted in the target domain and so \( p_s(Y|X_s = x_s, X_b = x_b) \) no longer gives the correct probability for \( p_t(Y|X_b = x_b) \); but we assume also that the shared features are not corrupted, and that \( p_s(Y|X_b = x_b) = p_t(Y|X_b = x_b) \). Under the corrupted feature assumption, we prefer to exclude the source-only features.

As an illustration, consider the problem of name detection in text using capitalization features. If the target domain contains no capitalized words, the corrupted feature assumption implies that capitalization is missing and should be ignored in training. (Satpal and Sarawagi (2007) considered a similar example where the source domain followed usual capitalization conventions and the target domain was in all caps.) However, the biased sampling assumption just assumes the target domain contains no capitalized words—perhaps it contains very few names in general. In that case it is better to include the capitalization feature in training, since the lack of capitalization on the target domain correctly indicates that most words are not names.

Some papers have proposed more general versions of the feature restriction approach: Tan et al. (2009) proposed using only features that occur frequently in both domains and that have similar probability. Satpal and Sarawagi (2007) proposed penalizing the weights on CRF features that differ in expected value between the source and target domain. This is related to the feature restriction approach since features that do not occur or that are constant in the target domain will be penalized. These methods are motivated from the corrupted feature assumption.
3.3 Latent Semantic Analysis and Principal Component Analysis

LSA and PCA derive a set of vectors \( \{ \bar{u}_1, \ldots, \bar{u}_l \} \) spanning a subspace of the original feature space. Let \( U \) be a matrix containing the vectors as rows and let \( X' = UX \) be the projection of \( X \) onto the space. The assumption behind the use of the projected features for domain transfer is that:

\[
p_s(Y = 1|X' = x') = p_t(Y = 1|X' = x') = p(Y = 1|X' = x'),
\]
i.e., there exists a domain-general model over the projected features \( X' \). It is assumed also that the projected features contain most of the label predictive information found in the original features, so:

\[
p(Y = 1|X' = x') \approx p(Y = 1|X = x).
\]

The vectors \( \bar{u}_1, \ldots, \bar{u}_l \) must be learned from the correlation structure of the source and/or target data; and in order to learn the model correctly on this subspace, the source data must span the entire subspace.

To understand how PCA or LSA might work for domain adaptation, we consider PCA on the simple 3-feature Gaussian example in Section 3.2.1 (PCA is the same as LSA except for mean subtraction, so we consider only PCA here. LSA is intended for sparse word features; we perform experiments with LSA using the simulated and real datasets later.) In the linear model, there exists a one-dimensional subspace that satisfies the assumptions above, namely, the projection onto the vector \( \beta \). However, this vector cannot be derived from PCA on the original data. We consider three possibilities: (a) PCA on the source data only; (b) PCA on the target data only; (c) PCA on the combined source and target data.

In (a), the principle components (from the theoretical covariance matrix of the source domain) are:

\[
\bar{u}_1 = [0.6618, 0.7497, 0]
\]
\[
\bar{u}_2 = [0.7497, -0.6618, 0].
\]

The model learned from source data projected onto both of these dimensions is the same as the full source-domain model on the original features, so there is no difference in transfer
performance. We could consider instead learning a model on just the feature defined by projection onto the first component:

\[ F_1 \equiv 0.6618 X_s + 0.7497 X_b. \]

The decision boundary will consist of a threshold for this feature \( f_1 \) such that:

\[ E\{\beta_s \cdot X_s + \beta_b \cdot X_b + \beta_0 | F_1 = f_1\} = 0. \]

Generally the correct threshold \( f_1 \) on the source domain is not the correct one on the target domain since the distributions of \( X_s, X_b \) given \( F_1 = f_1 \) differ between domains. So there is no obvious benefit to transfer performance by using PCA features from the source domain.

If we use (b), PCA features from the target domain, we can learn a model on the source data that utilizes features occurring only in the target domain. The PCA vectors are:

\[ \bar{u}_1 = [0, 0.7497, 0.6618] \]
\[ \bar{u}_2 = [0, -0.6618, 0.7497], \]

so the PCA features are linear combinations of shared and target-only features. The problem is that the source data does not occupy the full subspace spanned by these vectors, so we cannot hope to learn the correct model on them using the source data. Consider instead learning a model on just the feature defined by the first component:

\[ F_1 = 0.7497 X_b + 0.6618 X_t. \]

On the source domain this feature is perfectly correlated with \( X_b \), so the model learned will be equivalent to the shared-only source-domain model. However, when evaluated on the target domain it is not the same as the shared-only model, because \( F_1 \) includes both \( X_t \) and \( X_b \). When the shared-only model transfers correctly to the target domain, the model here will transfer incorrectly because it puts incorrect weight on the target domain features.

Consider again the positive correspondence cases (3a and 3b) in Section 3.2.1 If we were to construct a feature such as \( X_f = X_s + X_t \) from the unlabeled data, based on the fact that both \( X_s \) and \( X_t \) are positively correlated with \( X_b \), we could then learn a model over \( X_f, X_b \) in the source domain, which would be identical to the correct model on \( X_t, X_b \) in the target domain, and would outperform the feature restriction approach.
The problem is that $X_s, X_t$ are not positively correlated with each other—in fact they never co-occur, and have covariance 0. Here we consider PCA on a combination of the source and target data with 50% weighting of each (i.e., the eigenvectors of the average of the covariance matrices from each domain). The PCA resulting vectors are:

$$\bar{u}_1 = [0.365, 0.857, 0.365]$$
$$\bar{u}_2 = [0.707, 0, -0.707]$$
$$\bar{u}_3 = [0.606, -0.520, 0.606].$$

The feature defined by projection onto $\bar{u}_1$ captures some of the desired correspondence between $X_s, X_t$. A model learned on this feature will be less accurate than the model on $X_f, X_b$, but more accurate than the model on just $X_b$. Unfortunately, the second feature (defined by projection onto $\bar{u}_2$) assigns opposite weights to $X_s$ and $X_t$; using this feature will generally lead to degradation under the positive correspondence scenario.

This illustrates a problem with doing PCA on the source and target data together: the PCA vectors will not only describe variance that is shared across domains, but also variance due to differences between domains. In some cases, the PCA vectors may improve transfer performance, but in other cases they can degrade it. We next consider Structural Correspondence Learning, which learns correspondences between domain-specific features using multiple “auxiliary” prediction tasks.

### 3.4 Structural Correspondence Learning

As noted in Chapter 2, SCL is based on alternating structural optimization (ASO) (Ando et al. 2005). As a semi-supervised learning method, ASO is motivated by the goal of learning a good hypothesis space from unlabeled data; this is done by considering many “auxiliary” prediction problems that are likely to be similar to the real prediction problem. For domain adaptation, Blitzer (2007) motivated SCL by the same idea. In addition, Blitzer motivated SCL as a method to “align” domain-specific features. In SCL the auxiliary prediction tasks are based on predicting a set of domain-shared features, called pivots, and are learned on all the other features (including domain-specific ones). If source-only and target-only features have similar predictive weights across many pivot features, they are said to “correspond”
and will be represented by similar feature values in the new feature space resulting from SVD.

We can formalize the SCL assumption as follows. Assume for simplicity that there are $K$ pivot features $\{X_{b,k}\}_{k=1}^K$ which are identical to the shared features (in practice the pivots will be a subset of the shared features). Let $\bar{X}_s$, $\bar{X}_t$, and $\bar{X}_b$ be vectors representing the source-specific, target-specific, and shared features, and let $\bar{X} = [\bar{X}_s, \bar{X}_t]$, and $\bar{X}_{full} = [\bar{X}_s, \bar{X}_t, \bar{X}_b]$. Then SCL learns a set of pivot predictors $\bar{a}_k$ to predict pivot values according to:

$$X_{b,k} \approx g(\bar{a}_k \cdot \bar{X}), \forall k,$$

where $g(\cdot)$ is a function given by the chosen model family. SCL then derives a set of orthogonal vectors $\{\bar{u}_j\}_{j=1}^L, L \leq K$ that span the pivot predictors $\{\bar{a}_k\}_{k=1}^K$. So we can write:

$$\bar{a}_k = \sum_{j=1}^L \alpha_{k,j} \bar{u}_j,$$

for some scalars $\alpha_{k,j}$ and so:

$$X_{b,k} \approx g\left(\sum_{j=1}^L \alpha_{k,j} (\bar{X} \cdot \bar{u}_j)\right), \forall k.$$

Note that $\bar{X} \cdot \bar{u}_j$ represents SCL feature $j$. SCL assumes a linear decision function for $\hat{Y}$:

$$\hat{Y} = \text{sign}\left(\sum_{j=1}^L \beta_j (\bar{X} \cdot \bar{u}_j) + \bar{c} \cdot \bar{X}_{full} + \beta_0\right),$$

where the terms $\bar{c} \cdot \bar{X}_{full}$ captures dependence on the raw features. (Note that $\beta_j$ and $\bar{c}$ are not necessarily unique.)

Each singular vector $\bar{u}_j$ has a source-specific part $\bar{u}_{j,s}$ and a target-specific part $\bar{u}_{j,t}$ which contain the entries corresponding to the domain-specific features. Therefore, SCL feature $j$ has a source-specific part $\bar{X}_s \cdot \bar{u}_{j,s}$ and a target-specific part $\bar{X}_t \cdot \bar{u}_{j,t}$. These parts “correspond” in the sense that they have the same coefficient $\alpha_{k,j}$ for predicting each pivot feature $X_{b,k}$, as well as the same coefficient $\beta_j$ for predicting the label variable $Y$. This set of conditions resembles the “positive correspondence” justification for feature restriction proposed above. Recall that under positive correspondence, $X_s$ and $X_t$ have the same
relationship to $X_b$ (specifically, the same conditional distributions given $X_b$), and the same relationship to the target variable $Y$. SCL derives corresponding domain-specific features $\bar{X}_s \cdot \bar{u}_{j,s}$ and $\bar{X}_t \cdot \bar{u}_{j,t}$, which have the same relationships to many shared pivot features, and are assumed to have the same relationship to the target variable.

However, there are a couple of important differences between SCL and feature restriction. First, the SCL method actually uses the source-specific and target-specific features; it finds a model using $X_s$ and $X_t$ rather than just a model on $X_b$. In practice, since the SCL features are used together with the raw features, and since they represent linear combinations of the raw features, the model learned is not unique; Blitzer et al. (2006) suggested that the SCL features be scaled up relative to the raw features to encourage their usage when learning a regularized linear model. This scaling trades off a model on the SCL features only (which exclude the non-pivot features but include the target-specific features) with the full source-domain model. While the effect of the scaling has yet to be fully analyzed, we can argue that if the assumptions are satisfied, the inclusion of the target-specific features will often give lower prediction error than just using the shared features—although it can also lead to much greater cross-domain degradation if the assumptions are not satisfied. Thus, SCL is more “powerful” than feature restriction.

A second difference between feature restriction and SCL comes from the fact that SCL does not typically use all singular vectors $\{\bar{u}_j\}_{j=1}^K$; it uses only the top ones. Intuitively, these top vectors describe structure that appears in predicting many $X_{b,k}$ from $\bar{X}$, but does not necessarily include correlations between $\bar{X}$ and all $X_{b,k}$. This is in contrast with LSA. As an example, if a single shared feature is highly positively correlated with both a source-specific and target-specific feature, feature restriction and LSA would assume that the source-specific and target-specific features have the same relationships to the class variable. However, SCL requires that the source-specific and target-specific feature be positively correlated with many shared (pivot) features.

SCL is less likely than LSA to include dimensions that discriminate domains. The reason is that it derives dimensions that span pivot predictors, rather than dimensions that span the raw feature vectors. Since the pivots are assumed to occur frequently in both domains, the pivot predictors are less likely to contain substantial structure due to source and target
“parts.” However, it is still possible for SCL to include such dimensions, which can degrade performance, as we show in later experiments on 20Newsgroups.

3.5 Experiments with Feature Representation Methods on Synthetic Datasets

This section describes experiments with two kinds of synthetic datasets. Both simulate a binary text classification task using bag-of-words features. The first, which we call the SCL Dataset, represents an ideal domain adaptation setting for SCL, which we then alter in order to examine how SCL and other feature representation methods behave when the correspondence assumption is violated. The SCL Dataset follows a linear generating model for $p(Y = 1|X = x)$, like the three-feature dataset in Section 3.2.1 except we use binary features. The second dataset, which we call the Generative Dataset, follows a feature generative model according to $p(X = x|Y = y) = \prod_i p(X_i = x_i|Y = y)$, where $p(X_i = x_i|Y = y)$ is Bernoulli-distributed. On this dataset we compare the performance of the three feature representation methods as the proportion of shared features decreases.

In all experiments we use a logistic regression model based on the LIBLINEAR package (Fan et al. 2008), with binary features representing presence/absence of each word. The words are divided into source-specific, target-specific, and shared features; the source-specific features are zero in the target data and vice versa.

3.5.1 SCL Dataset

In Section 3.2.1 we examined the feature restriction approach in simple scenarios based on linear models over three features. Since SCL assumes multiple domain-specific and shared features, for this section we use a toy example with a high dimensional space. We generate three sets of sparse, binary features: 200 domain-specific for each domain and 400 shared. Let $\{X_{s,i}\}_{i=1}^{200}$ represent the source-specific features, $\{X_{t,i}\}_{i=1}^{200}$ represent the target-specific features, and $\{X_{b,i}\}_{i=1}^{400}$ represent the shared features. We represent each example by an 800-dimensional feature vector:

$$X = [X_{s,1}, \ldots, X_{s,200}, X_{t,1}, \ldots, X_{t,200}, X_{b,1}, \ldots, X_{b,400}].$$
with $X_{t,i} = 0$ for all source examples, and $X_{s,i} = 0$ for all target examples. To generate an ideal case for SCL we construct a correspondence between source and target features, in which each source-specific feature corresponds to one target-specific feature. This is implemented by requiring that:

- $X_{s,i}$ and $X_{t,j}$, for $i = j$, have identical correlation with every shared feature
- $X_{s,i}$ and $X_{t,j}$, for $i = j$, have identical prediction weights for the label

More specifically, for each domain, the domain-specific features are generated independently and then used to generate each shared feature $X_{b,j}$ via a linear model such that:

$$X_{b,j} = 0.5 \times \text{sign}(\bar{\alpha}_j \cdot X_s + \bar{\alpha}_j \cdot X_t + \epsilon_1 - \delta) + 0.5,$$

where $\delta$ is a constant threshold and $\bar{\alpha}_j$ is the true prediction vector for shared feature $X_{b,j}$ based on the source features $X_s$ or the target features $X_t$ (depending on the domain). We then generate the class label as:

$$Y = \text{sign}(\bar{\beta}_d \cdot X_s + \bar{\beta}_d \cdot X_t + \bar{\beta}_b \cdot X_b + \epsilon_2),$$

where $\bar{\beta}_d$ is a prediction vector based on the domain-specific features (which is identical on both domains), and $\bar{\beta}_b$ is a prediction vector based on the shared features. $\epsilon_1$ and $\epsilon_2$ are independent random variables that add noise to the predictions.

We generated $\bar{\beta}_d$, $\bar{\beta}_b$, and $\{\bar{\alpha}_j\}_{j=1}^{400}$ randomly; the entries of each vector are IID and follow $\mathcal{N}(0, 1)$. The domain-specific features are IID and binary with $p(X = 1) = 0.2$. We set $\delta = 4$ (two standard deviations of the zero-mean random variable defined by $\bar{\alpha}_j \cdot X_s$) and set $\epsilon_1, \epsilon_2 \sim \mathcal{N}(0, 0.1)$. We used 10,000 examples for each domain, half for train and half for test.

The baseline is the result of training on the source domain training set with no adaptation. Only features occurring at least 3 times in the training set were included. (By “occur” we mean that the feature has value 1 rather than 0.) The feature restriction method used only those features that occurred at least once in the target domain. The “gold standard” represents the ideal case of training on the target domain, with the same size training set.

\[1\] Note that it is possible to do better than the gold standard with some domain adaptation methods, such as SCL or co-training, which can exploit unlabeled data even in semi-supervised learning settings.
For SCL, pivots were defined as all features occurring at least 5 times in both domains; this usually included all 400 shared features. SCL was performed using all 20,000 source and target samples to learn predictors \( \hat{\alpha}_j \) of the pivot features from the other features, which were included for prediction only if they occurred at least 3 times. The auxiliary prediction tasks were also trained via logistic regression using LIBLINEAR. The resulting 400 auxiliary task prediction vectors were collected into a \( 400 \times 400 \) dimensional matrix on which SVD was performed. This led to at most 400 new SCL features formed by projecting the original feature representation onto each SVD dimension. To train the final classifier based on the source training set, we used all original features that occurred at least 3 times in the source training data, plus a variable number of the SCL features.

In addition to the number of pivots, SCL has several other hyperparameters: the number of dimensions of the SVD (i.e., the number of SCL features); the regularization parameter for the auxiliary learning tasks; the regularization parameter for the final classifier; and the scaling factor for the SCL features relative to the raw features, which was used by Blitzer et al. (2006) to encourage the learner to put weight on the SCL features. For regularization we chose a single value (\( C = 100 \) weighting on the empirical loss term) from tuning on the source domain; this value was used for all feature sets and auxiliary learning tasks. We investigated the dependence on SCL dimensions and feature scaling, which is shown on the left in Figure 3.6. SCL performance does not appear to be very sensitive to the scaling factor although some scaling is helpful (Blitzer et al. (2006) used a factor of 5 in all experiments). However, it is quite sensitive to the number of SCL dimensions in this case, with performance actually dropping below the baseline for too many dimensions. This suggests that the latter dimensions capture spurious correspondences, and that using those dimensions causes the classifier to put inappropriate weights on target domain features. In the absence of knowledge about the target domain, it would be better to err on the side of too few SCL dimensions rather than too many.

We compare SCL to LSA and feature restriction. The LSA features were derived from SVD on the data matrix of all 20,000 samples, consisting of both domains together. Only features occurring at least 3 times were included. The features resulting from projection onto the LSA vectors were scaled and concatenated onto the original feature vector in the
same manner as the SCL features. Figure 3.6 (right) shows the effect of different numbers of LSA features and different scaling factors. LSA appears to be much less effective than SCL for this data set, and less sensitive to the scaling factor. The optimal number of dimensions is also 200, but it does much worse than baseline for 400 dimensions.

In Figure 3.7 we examine what happens when the correspondence decays between the true prediction weights while the correlations between features stay the same. To do this we keep the generation of the domain-specific and shared features identical to the above and use the optimal number of SCL dimensions above, but we generate the class labels from:

$$Y = \text{sign}(\bar{\beta}_d \cdot X_s + \bar{\beta}_{d,\text{tar}} \cdot X_t + \bar{\beta}_b \cdot X_b + \epsilon_2),$$

where:

$$\bar{\beta}_{d,\text{tar}} = \frac{(1 - \gamma)\bar{\beta}_d + \gamma \bar{\beta}_{\text{new}}}{\sqrt{\gamma^2 + (1 - \gamma)^2}}.$$

As $\gamma$ is increased from 0 to 1, the correspondence between the source-specific and target-specific feature weights decreases. (It is necessary to divide by a factor of $\sqrt{\gamma^2 + (1 - \gamma)^2}$ so that the $\bar{\beta}_{d,\text{tar}}$ entries will have the same variance as the $\bar{\beta}_d$ entries.) So while the SCL features remain the same, the correspondences captured by those features become less related to label prediction.

For both SCL and LSA we use 200 dimensions (which is optimal for no correspondence noise) and scale the features by 5 relative to the raw features. As expected, as $\gamma$ increases the performance of SCL drops dramatically. The feature restriction method also drops, which is expected from the analysis in Section 3.2.1 since this method also makes use of assumed correspondences. The SCL method continues to be useful even after the feature restriction approach drops below the baseline, but it does worse than the feature restriction method when $\bar{\beta}_d$ and $\bar{\beta}_{d,\text{tar}}$ are completely independent. On average LSA is only slightly better than the baseline when $\gamma = 0$ (in one case it is worse than the baseline). However, it does not degrade from the baseline as much as SCL and the feature restriction approach when the correspondence decays.

Based on this, one would generally not choose SCL unless knowledge about the problem suggested the existence of predictive correspondences. However, our method of simulating “non-correspondence” may be pessimistic with respect to real datasets—one might argue
Figure 3.6: Performance of SCL (left) and LSA (right) on the SCL Dataset, as a function of the feature scaling factor using different numbers of SCL or LSA dimensions. The “baseline cross-dom” is the baseline transfer result (using the original feature set); “gold-standard” is the in-domain (target-trained) result.
Figure 3.7: Performance of SCL, LSA, feature-restriction, baseline transfer and gold standard on the SCL Dataset as the correspondence decays between the true source and target prediction weights. SCL and LSA used $d = 200$ dimensions and scale factor 5. Shown results are averages from 5 runs, with max and min shown in the error bars.
that it is unusual for features to have high correspondence in the auxiliary prediction tasks but low correspondence in the real task.

We note finally that our dataset generation procedure leads to a rather high variance in cross-domain baseline performance: for example, with $\gamma = 0$ over 100 runs the range was 70.3 to 77.9. By contrast, the gold-standard in-domain performance on these target domains ranged from 92.5 to 94.5.

3.5.2 Generative Dataset

The data is generated as follows: for each example we first generate a label with $p(Y = 1) = 0.5$. Then for each example in class $y$, we generate the values of $k$ binary features, each IID with:

$$p(X_k = 1|Y = y) = p_{k,y}$$
$$p(X_k = 0|Y = y) = 1 - p_{k,y}.$$  

Features 1 through $l$ are source-specific and only generated for the source data (they are set to zero for the target data); features $l + 1$ through $2l$ are target-specific and only generated for the target data (set to zero for the source data). The rest are shared features with the same occurrence probability in each domain. The feature occurrence probability $p_{k,y}$ for each feature $k$ and class $y$ is generated IID according to $\max(0, P)$, where $P \sim \mathcal{N}(0.05, 0.02)$. We experimented with the distribution of $P$ in order to get a sparse corpus with less than perfect performance and some cross-domain baseline degradation. Like the SCL Dataset, we use 10,000 samples in each domain, with half for train and half for test.

In Figure 3.8 we plot the target domain performance of SCL and LSA, on a dataset with 200 domain-specific features for each domain and 400 shared features. We used the same setup as the SCL Dataset; the regularization parameter for LIBLINEAR was set at $C = 0.05$ from tuning on classification of the source test set. It can be seen that, like the SCL Dataset, the performance is sensitive to the number of dimensions; however, the optimal number of dimensions is much smaller than for the SCL Dataset, even though they have the same number of domain-specific and shared features.

Figure 3.9 compares the performance of SCL, LSA, and feature restriction as the pro-
Figure 3.8: Performance of SCL (left) and LSA (right) on the Generative Dataset, as a function of the feature scaling factor using different numbers of SCL or LSA dimensions.
3.6 Experiments with Feature Representation Methods on Real Datasets

3.6.1 Datasets

Blitzer’s Sentiment Classification dataset \cite{blitzer2007domain} consists of Amazon product reviews in four domains: books, DVDs, kitchen appliances, and electronics. The task is to classify the reviews as positive or negative (labels were derived from the number of stars on each review). The original adaptation approach in \cite{blitzer2007domain} used SCL; subsequently the dataset was used in other domain adaptation papers, including \cite{pan2010domain,blitzer2009domain,blitzer2008domain,ji2011domain}, and \cite{mansour2009domain}. In \cite{pan2010domain}, several approaches were compared including SCL, a version of LSA, and other feature co-occurrence methods; \cite{blitzer2009domain} used a subspace projection method based on CCA; and \cite{ji2011domain} proposed a new variant of SCL. \cite{blitzer2008domain} and \cite{mansour2009domain} presented experiments in support of proposed error bounds using weighted combinations of data from multiple source domains or from both source and target domain. The Sentiment Classification dataset represents a case in which SCL is known to work well, but to our knowledge no one has investigated feature restriction or instance weighting on this dataset.

\footnote{http://www.cs.jhu.edu/~mdredze/datasets/sentiment/}
Figure 3.9: Performance of SCL, LSA, feature-restriction, baseline transfer and gold standard on the Generative Dataset as the proportion of domain-specific to shared features increases. Shown results are averages from 5 runs, with max and min shown in the error bars.
Our second dataset, the 20Newsgroups dataset[^3] is a well-known collection of newsgroup postings that has been used in many papers on text classification and clustering. The standard task is to classify the newsgroup of each posting, which does not involve domain adaptation. However, recently this dataset has been used to test domain adaptation methods including: graph-based and clustering methods in Ling et al. (2008b), Gao et al. (2008), Dai et al. (2007a), Xing et al. (2007), Ling et al. (2008a), Dai et al. (2009b), Zhuang et al. (2009) and He et al. (2009); topic-bridged pLSA (Xue et al. 2008); instance weighting (Xue et al. 2008; Ling et al. 2008a); Naive Bayes EM (Dai et al. 2007b); variants of SCL (Ji et al. 2011); and feature representation methods to minimize source/target distribution difference in Chen et al. (2009). Transductive SVMs have also been used as a baseline in many of these works. The most common domain adaptation problem is to classify the top-level newsgroup category when the source and target domains consist of different sub-categories. We used the six binary tasks and sub-category divisions described by Dai et al. (2007a). For example, the “comp-vs-rec” problem consists of classifying the postings into comp.* vs. rec.*. The source domain consists of comp.\{graphics,sys.ibm.pc.hardware, sys.mac.hardware\} and rec.\{motorcycles, sport.hockey\} while the target domain consists of comp.\{os.ms-window-misc, windows.x\}, and rec.\{autos, sport.baseball\}.

3.6.2 Sentiment Classification Results

We use the pre-processed data which consists of unigrams and bigrams extracted from each review. Each feature represents the total count of a unigram or bigram in the review. Unlike Blitzer (2007), we do not normalize the raw feature counts to sum to one for each example. The reasons we do not do this are that: (a) normalization might potentially cause numerical problems, and our experiments indicate that it does not generally improve performance; (b) normalization would change the feature values under feature restriction, and we wish to compare results without this additional effect. We present results for each of the 12 domain adaptation pairs as done in Blitzer et al. (2007). Each domain contains between 5118 and 5901 reviews roughly balanced between positive and negative. In each domain, we randomly

[^3]: http://people.csail.mit.edu/jrennie/20Newsgroups/
selected 1600 examples as the training set and 1000 different examples as the test set. This matches the training set size used by Pan et al. (2010) and Blitzer et al. (2007). For each domain pair, we train on the source domain training set and test on the target domain test set. The entire collection of data in each domain was used as unlabeled data for domain adaptation; thus we included the test set in the unlabeled pool of target data.

We use LIBLINEAR (Fan et al. 2008) as the classifier under logistic regression mode. For all experiments we used a single value of the regularization parameter ($C = 0.1$); this value is close to the optimum found by in-domain tuning with the raw feature set on all domains. For the feature restriction approach we remove features if they do not occur at all in any of the 5000+ examples in the target domain. The number of features in the training sets ranged from 82k (kitchen) to 167k (dvds), while the shared-only set ranged from 30k to 59k. For SCL, we use mutual information to select the pivots out of features occurring at least 5 times in both domains. We use 50 SCL dimensions (same as Pan et al. (2010); Blitzer et al. (2007)) and 1000 pivots. All available data in both domains (5118 to 5901 examples per domain, depending on the domain) are used as unlabeled data to learn the SCL dimensions. Pivot predictors are learned only on unigram and bigram features that occur at least twice and that do not overlap with any of the pivots; the total number of predictor features ranged from 27k to 54k. Even using the published parameters for this dataset, we found we had to tune a few details of the SCL algorithm to get it to work, and this was done on two tasks (DVDs on books; books on electronics). We set the scale factor for the SCL features to 5, and the regularization parameter for learning the pivot predictors to the same value as the classifier. Our baseline and SCL results are generally comparable to those of Blitzer et al. (2007) and Pan et al. (2010), with some differences. Those differences may be due to, for instance, different train and test divisions, different numbers of unlabeled examples, different classifiers, feature normalization, different SCL scale factors, and different pivot and predictor selections. (Note that Ji et al. (2011) also reported results for SCL on this dataset, but they used both a larger training set and a

\[4\text{The domain adaptation methods we discuss in this chapter can be applied in either a transductive or inductive manner, i.e., the unlabeled pool used for adaptation may include the test set or not. Because of the limited amount of data we chose to apply the methods transductively, which may lead to slightly better results than if we applied the model to a held-out set.}\]
smaller number of pivots; their baseline results are generally comparable or better, and their SCL are comparable.)

For LSA, we perform SVD on the data matrix for all unlabeled examples using all features occurring at least twice (this ranged from 108k to 175k features). Although there are slightly different numbers of unlabeled examples in each domain, we ensured 50/50 weighting by scaling the feature vectors in each domain prior to forming the data matrix. If the source and target domains have $n_1$ and $n_2$ unlabeled examples respectively, we scale the source feature vectors by $\sqrt{n_2}$ and the target feature vectors by $\sqrt{n_1}$. This ensures that the sample correlation matrix is an equal average of the correlation matrix for each domain. After computing the SVD, the LSA features resulting from projection are concatenated onto the raw feature vector in the same manner as the SCL features, with the same number of dimensions and scale factor.

Our results validate that SCL works on this dataset, and it is clearly superior to the other two methods, giving the best performance in all cases. Feature restriction usually leads to a small improvement over baseline, but in a few cases gets worse. LSA does especially poorly, generally getting below baseline. The LSA results of Pan et al. (2010) were somewhat more mixed, doing better than baseline in some cases. Their version of LSA (not described in detail) may only use a subset of features selected as having high mutual information with the domain label, although the motivation for that approach is unclear. Our experiments on two tasks indicate that LSA performance may be sensitive to the number of dimensions and scaling factor, and that there is not a single good setting across tasks. In either case, LSA again proves to be a fragile method compared to SCL.

3.6.3 20Newsgroups Results

For the 20Newsgroups dataset, we remove the header of each message (up to the first newline) and use Rainbow’s algorithm (McCallum 1996) to prune messages containing mostly uuencoded blocks. Messages without Latin characters are removed. We apply word normalization, including punctuation and case removal, and replacement of numbers and dates with a standard token. We then use the same features as Xue et al. (2008), which consist
Figure 3.10: Comparison of feature restriction, LSA, and SCL performance for the Sentiment task, along with baseline and gold standard.
of stemmed unigram counts with TF-IDF weighting. We use the Perl module Lingua::Stem for stemming. As in Xue et al. (2008), stopwords (from Rainbow) and rare words (occurring less than 3 times in the training set) were removed. Our experiments showed that TF-IDF weighting performed slightly better than raw counts on in-domain training for the “comp vs. rec” task. Unlike Xue et al. (2008), we did not rescale each feature vector to sum to one, because this step did not seem to improve performance, and it would lead to different feature values under the feature restriction approach. Note that the features for the 20Newsgroups tasks are different from the features for Sentiment Classification in several ways: TF-IDF weighting, unigram features only, frequency thresholding, stemming, and stopword removal. These differences are reasonable since topic and sentiment classification are different problems, and our choices are inspired by previous work on these datasets.

Each domain contains between 3375 and 4899 examples. In Xue et al. (2008), the full source and target domains are used as the labeled training and test sets, and cross-validation on the target domain is used to estimate the gold-standard result from target domain training. We use a similar setup, but with the same size labeled training set (3375) in each case, so that results are more comparable across tasks (each task has the same number of labeled examples, but varying amounts of unlabeled data). The labeled training set of 3375 examples is derived by random subsampling from the full source domain. The gold-standard result is averaged over the held-out set from cross validation using target domain training sets of size 3375. As in the Sentiment Dataset, we use all available data in both domains as the unlabeled data for domain adaptation. We use the labeled training set of 3375 examples to compute IDF weights for each word when that word is used as a raw feature; we use the full unlabeled domain data to compute (domain-specific) IDF weights for the words used in the SCL and LSA features.

As the base classifier, we again use LIBLINEAR, with $C = 100$ (which is close to optimal for in-domain training across all six tasks). The number of features in the source domain training sets ranged from 11k to 13k. For the feature restriction method, we remove features that do not occur at least once in the target domain; the resulting feature set ranged in size from 7k to 8k. For SCL and LSA, we experimented with different settings, using the “comp vs rec” task as a development task. From this we set $C = 1$ in the auxiliary prediction
Figure 3.11: Comparison of feature restriction, LSA, and SCL performance for 20Newsgroups tasks, along with baseline and gold standard. The settings of SCL and LSA were tuned on the “comp-vs-rec” task.
tasks. SCL used only 5 dimensions of the SVD; LSA used 50 dimensions. Both used a scale factor of 5 relative to the raw features. Otherwise, for SCL, we kept the same settings as in the Sentiment Classification: we used 1000 pivots selected using mutual information on the labeled source set, out of words occurring at least 5 times in each domain. Features used in learning the LSA dimensions and the SCL auxiliary tasks were required to occur at least 3 times in the unlabeled data set for the corresponding domain. This resulted in 18k to 20k total LSA features, and 1000 fewer as SCL predictor features. All features were represented as TF-IDF values in learning the features and in projection, although the auxiliary tasks are still binary prediction tasks (presence vs. absence of pivot word).

Results for all methods are shown in Figure 3.11. Our baseline results are generally comparable to or better than those reported in other papers. In Xue et al. (2008), baseline results were presented using both Naive Bayes and SVM classifiers. Their gold standard results were close for both classifiers, but their baseline cross-domain results varied between the two classifiers. Our gold standard results are close to those reported by Xue et al. (2008) (noting that our training set size is smaller for some tasks). Our baseline cross-domain results differ, in some cases, from both results of Xue et al. (2008), but are within or close to the range of those results. A different set of baseline cross-domain results were reported by Gao et al. (2008), using various classifiers (Winnow, logistic regression, SVM). Compared with those, our baseline cross-domain results tend to be better. Possible reasons for these differences include: different settings for the classifier parameters, such as the regularization parameter; different kinds of word normalization; and different methods for computing the IDF value for each word. (In particular, our experiments indicate that cross-domain performance is extraordinarily sensitive to the regularization parameter.) Finally, Ji et al. (2011) also reported baseline results that were consistently worse than ours; however the exact details of feature weighting were not described.

Both SCL and LSA show mixed results. SCL appears less effective for this dataset than for the Sentiment Classification tasks. In four cases it improves transfer performance slightly, and in two cases it degrades performance dramatically. These two cases appear to suffer from "misalignment" of features as discussed by Blitzer et al. (2007). Misalignment occurs when the SCL dimensions distinguish subtopics or domains rather than the labels
of interest. If a misaligned dimension is discriminative enough to be useful in the source domain, it can misclassify target examples. For instance, in rec-vs-talk, the source domain contains rec.{autos, motorcycles} and talk.politics.{guns, misc} while the target domain contains rec.sport.{baseball, hockey} and talk.{politics.mideast, religion.misc}. One of the SCL dimensions assigns high positive weights to words related to autos and motorcycles (bike, break, manual) and high negative weights to words related to team sports (player, team, game), while splitting words related to mideast politics and religion (armenian vs. god). This dimension may result from the fact that some pivots mean different things in the source and target domains. For instance, car, ride, and drive are pivot features because they occur in both domains and have high mutual information with the label in the source domain. But in the target domain, they behave differently: they are not predictive of “rec”— in fact, they are more common in the “talk” postings. They are also much less common overall: most of the occurrences, in both domains, occur in the rec.{autos, motorcycles} section of the source domain. So the SCL dimension distinguishes rec.{autos, motorcycles} from either everything else, or from rec.{baseball, hockey}.

This suggests that some improvement may be possible by setting the frequency threshold higher for selection of the pivots. A better method may be to require pivots to have similar occurrence frequencies in both domains. Doing so may help weed out pivots that occur in both domains but are highly correlated with one subset of one of the domains.

Recently-reported results by Ji et al. (2011) also showed mixed results for SCL on this dataset. The focus of that work was a proposed new version of SCL that separates auxiliary tasks into domain-specific versions. The separately-learned weight vectors form separate rows in the weight matrix prior to SVD. They argued that their method will help eliminate problems with “polysemous” features (that exist in both domains and have different label associations). It seems that their method might help with some misalignment problems arising from pivot features that are polysemous or more common in one domain. However, it might cause difficulty with learning correspondences between domain-specific features, since in their method, source-only and target-only features never have non-zero weights simultaneously in the same weight vector. According to their results, their method does not generally outperform standard SCL. However, a combination version, which includes
both the separately-learned and a jointly-learned weight vector, does frequently lead to
an improvement over standard SCL—although only on one case does it outperform our
cross-domain baselines. Clearly, further analysis is needed to understand their approach.

In contrast to our results here and the results reported by Ji et al. (2011) with feature-
based methods, previous work using different domain adaptation methods (EM- and cluster-
based) have been very successful on these adaptation tasks. For instance, Dai et al. (2007a)
reported target domain accuracies ranging from 87% to 98% on all tasks using their co-
clustering method; Dai et al. (2007b) reported accuracies above 97% on all tasks using their
EM-based Naive Bayes method; Xue et al. (2008) reported accuracies above 95% for their
pLSA method; and Zhuang et al. (2009) reported accuracies above 96% for their mixed
regularization method.

For both SCL and LSA, the optimal number of dimensions appears to be variable across
tasks, so choosing the dimensions using a development task does not work well. In Section
3.7 we explore choosing the dimensions for both LSA and SCL automatically on each task
using the $A$-distance proposed by Ben-David et al. (2007).

### 3.7 Automatically choosing dimensions for LSA and SCL

As noted in the previous chapter, Ben-David et al. (2007) derived a bound on the target do-
main generalization error that includes a distribution difference measure called $A$-distance.
This distance is a function of the minimum error of a hypothesis that discriminates between
the source and target domains. While this cannot be computed exactly (both because the
distributions are not available, and because it is infeasible find the minimum error rate clas-
sifier on a finite sample using 0-1 loss), it can be approximated. In the work of Ben-David
et al. (2007) and Blitzer et al. (2007), a classifier was trained to discriminate source and
target samples using Huber loss, and the authors used the resulting minimum Huber loss as
the proxy for $A$-distance. For the Sentiment Classification tasks, Blitzer et al. (2007) showed
that the approximated $A$-distance of domain pairs using the SCL feature representation is
correlated with the transfer performance.

We trained a logistic regression classifier to discriminate source and target training data,
and approximated $A$-distance as the error rate of this classifier on a held-out set of source
and target data. Our resulting $\mathcal{A}$-distances on the original feature representation led to the same ordering of domain pairs as Blitzer et al. (2007) for the Sentiment Classification task.

We would like to use the idea of $\mathcal{A}$-distance to automatically choose the dimensionality for LSA and SCL. Note that LSA, in particular, showed strong sensitivity to dimensionality in the above experiments; in some cases, the optimal number of dimensions tuned on one domain pair would lead to a large degradation on a related domain pair. One problem appears to be that LSA dimensions can capture differences between domains or variation within a single domain, rather than shared latent features. Thus, we experimented with automatically selecting the dimensionality of the LSA representation for each domain pair, by learning the $\mathcal{A}$-distance for each candidate dimensionality and selecting the value of $d$ with the lowest $\mathcal{A}$-distance. However, as is clear from Figure 3.10 this method does not always work: note that we originally tuned $d = 50$ on the books $\rightarrow$ electronics task, but this dimensionality performs poorly for the reverse direction, which has the same $\mathcal{A}$-distance.

We note also that rather than selecting all of the top $d$ LSA dimensions, we really want to select a subset of the dimensions. It is not practical to compute $\mathcal{A}$-distance for every subset of LSA dimensions, so we implemented a short cut: we tested the distance for each dimension individually (using a 1-dimensional linear classifier) and pruned dimensions with distance above a threshold. We observed anecdotally that this improved performance in some cases where LSA did very poorly, but not always—in a few cases, performance was worse. We observed additionally that some LSA dimensions display reversed label distributions in the source and target domain, even if they do not discriminate the unlabeled domain examples. For instance, the positive classes in the source and target domains may be projected onto opposite sides of an LSA vector. These LSA features can lead to large degradation in target domain performance. In order to detect these LSA vectors and remove them, we must have target domain labels; assuming the domains are not too different, we can get label estimates from the baseline classifier (based on raw word features).

5The analogue of the method of Blitzer et al. (2007) would be to report log loss on the training data, but we found that to be extremely sensitive to $C$. We do not address the issue of the regularizer but keep $C = 0.1$; much larger values caused numerical problems.
We therefore implemented the following method for pruning LSA dimensions from the top $D$ dimensions. For each candidate dimension $d$, let $\{X^d_i\}_{i=1}^n$ represent a set of source data and $\{X^d_i\}_{i=n+1}^{2n}$ an equal set of target data projected onto dimension $d$. Let $\{Z_i\}_{i=1}^{2n}$ be the domain labels (S or T) of each example; and let $\{\hat{Y}_i\}_{i=n+1}^{2n}$ be the labels for the target data estimated from the baseline classifier trained on the source data $\{X_i, Y_i\}_{i=1}^n$ in the original feature space. Let $h(X^d)$ be a linear classifier on dimension $d$ (i.e., a signed boundary) that predicts domain label (S or T). Let $h_s(X^d)$ be a classifier trained to predict class labels on $\{X^d_i, Y_i\}_{i=1}^n$. We keep dimension $d$ only if the following criteria are met:

1. $\min_{d} \frac{1}{2n} \sum_{i=1}^{2n} I(h(X^d_i) \neq Z_i) > m$
2. $\frac{1}{n} \sum_{i=n+1}^{2n} I(h_s(X^d_i) \neq \hat{Y}_i) < M$,

where $I(\cdot)$ is an indicator function. Note that criterion (1) measures the training error rate on $X^d$ for discriminating domains; it aims to keep only dimensions where this error rate is close to chance. In other words, we want to exclude features $X^d$ that have different distributions between domains, and thus have a lower-than-chance error rate for discriminating domains, because such features are bad for domain transfer. Criterion (2) measures the extent to which the target data predictions based on the raw features differ from the predictions using dimension $d$; it aims to roughly remove dimensions for which $p_s(y|x^d) \neq p_t(y|x_t)$, and will also remove some “noise” dimensions, although these are not a major concern since the source domain training should ignore them anyway.

We empirically set $M = 0.5$ and $m = 0.45$ from experiments on the Generative Dataset. Figure 3.12 shows the performance of this method on the synthetic Generative Dataset described in Section 3.5.2. We compare LSA and SCL with and without pruning, starting with the top $D = 50$ dimensions in each case, rather than 10 in Figure 3.9. We used the 5000 example training sets in both domains to implement the pruning procedure. The resulting selected features were added onto the raw feature vector; we kept the LSA and SCL feature scale factor at 2, as tuned in Section 3.5.2. Figure 3.12 compares the performance with and without pruning of the 50 dimensions. The number of LSA/SCL dimensions pruned varied substantially from run to run and for different domain-specific proportions.
Figure 3.12: Performance of automatic dimension pruning for LSA and SCL on the Generative Dataset. Shown results are averages from 5 runs, with max and min shown in the error bars.
Generally, criterion (2) was responsible for removing more dimensions. The plot shows that the pruning method is most useful for LSA, which shows a large drop in performance with the full set of dimensions when the number of domain-specific features is large. Out of 5 runs with different randomly generated datasets, the pruning method dramatically outperformed the full LSA, although one case showed anomalously low performance at the highest proportion of domain-specific features. This may be related to the high error rate in estimating target domain labels. This illustrates that the pruning method is not perfect: it does not remove all the harmful dimensions, and in some cases the classifier may put weight on these dimensions. Nevertheless, it appears to be a useful strategy for LSA on this dataset, on average. By contrast, for SCL the pruning method has very little effect, although many dimensions are removed. On this dataset SCL does not suffer from degradation like LSA. It seems that SCL dimensions are less likely than LSA to describe differences between domains or within one domain or the other, since they are based on predictions of shared features.

Unfortunately, the results of this method on the real datasets are significantly less impressive. Figure 3.13 shows the results on the Sentiment Classification dataset using (a) the full 50 dimensions (tuned on two cases, same as Figure 3.10); (b) full 200 dimensions; (c) pruned 200 dimensions. The pruning method does not generally lead to an improvement over the full set of either 50 or 200 LSA features, although it does so in a couple cases (electronics → books, kitchen → books) where the full sets do especially poorly. For SCL, the pruning method most frequently led to a degradation compared with using all 200 dimensions, and only in one case led to a noticeable improvement; in no case did it outperform the original 50 dimensions. Figure 3.14 shows the same comparison on the 20Newsgroups tasks. (In this case we compare with 5 dimensions for SCL and 50 dimensions for LSA, which was used in Figure 3.11 as tuned on the “comp vs rec” task.) In this case, the pruning method appears slightly more successful for SCL than for LSA, but in no case does the automatic pruning method achieve the performance of the original tuned 5 or 50 dimensions, and in most cases it is below the cross-domain baseline. Our results suggest that while this pruning approach has the potential to help in some cases, it is generally not successful (at least without additional tuning of the parameters \( m \) and \( M \), which defeats the purpose of
Figure 3.13: Results of automatic dimension pruning for LSA and SCL on the Sentiment Classification dataset.
Figure 3.14: Results of automatic dimension pruning for LSA and SCL on the 20Newsgroups dataset.
3.8 Conclusions

In this chapter we analyzed and compared feature restriction, LSA/PCA, and SCL. Assuming a domain-general linear model for generating the label variable $Y$ given the features $X$, we showed that the feature restriction approach, which excludes source-only features from training, can be motivated as a domain adaptation method under a “positive correspondence” assumption. We also described a motivation under a “corrupted feature” assumption. However, we illustrated other cases where excluding source-only features leads to worse performance. We showed that SCL can be justified from an assumption similar to positive correspondence with respect to the source and target parts of the SCL features, but SCL is more powerful than feature restriction since it can use the target-domain-only features. We showed that LSA on a merged collection of source and target examples has the potential to aggregate domain-specific features from both domains based on their “corresponding” correlations with shared features, and that such aggregate features can improve performance in the positive correspondence scenario. However, if the domains are sufficiently different, it is likely that some of the LSA dimensions will describe variance between domains, which can dramatically degrade performance under cross-domain learning. SCL learns correspondences by analyzing weight vectors learned on both domains, rather than analyzing raw feature vectors. Our experiments on the Generative synthetic dataset showed that as the proportion of domain-specific features increased, SCL maintained its performance while LSA degraded dramatically. Although more robust than LSA, SCL has the potential to suffer from similar problems, and was shown to cause substantial degradation on some tasks in the 20Newsgroups dataset.

A persistent challenge with unsupervised domain adaptation methods is the need to select parameters, such as the number of SCL features, without having access to labeled target domain data. For standard machine learning research, a widely-followed experimental framework exists: parameter tuning and model selection are performed on a “development” set, while final results are reported on a held-out test set. However, for unsupervised domain adaptation there is no consistent experimental framework in the literature. In some cases,
researchers may tune parameters on labeled target data; in other cases, results are presented across a range of parameter values. Researchers experimenting on multiple related tasks, such as the datasets used in this chapter, have sometimes used one of the tasks for tuning of parameters, a procedure that we have followed here. However, even when the tasks are closely related, optimal settings might differ between tasks. This appears to be true on the 20Newsgroups tasks, for which LSA and SCL performance is very sensitive to the number of dimensions. We experimented with a method to automatically prune LSA and SCL dimensions that might cause degradation in cross-domain learning. We implemented the method on the real datasets without tuning any parameters (except on the synthetic dataset). However, the results of this approach were quite mixed: on the 20Newsgroups tasks, automatic pruning led to an improvement in the majority of tasks, compared with using the “full” set of LSA or SCL features, but still did worse than baseline in most cases. For the Sentiment tasks, automatic pruning frequently led to degradation compared with using the full set of features, particularly for SCL.
Chapter 4

EXPERIMENTS WITH INSTANCE WEIGHTING FOR DOMAIN ADAPTATION

4.1 Introduction

In Chapter 2, we reviewed work on instance weighting for covariate shift, which is motivated by the assumptions that: (a) $p_s(y|x) = p_t(y|x)$; (b) $p_t(x)$ has support within $p_s(x)$; (c) the model family/hypothesis space does not contain a model that minimizes loss on both domains simultaneously. This chapter seeks to investigate two issues about instance weighting: its effect in combination with regularized learning, and its utility on real domain adaptation tasks that do not necessarily fit the covariate shift assumption. We first analyze instance weighting with regularized learning, then perform an experiment using a simulated ridge regression dataset. We next perform experiments with instance weighting on the Sentiment Classification dataset in a contrived scenario. Finally, we report experiments on the original domain adaptation tasks for Sentiment Classification and 20Newsgroups.

4.2 Instance Weighting and Regularized Learning

In order to analyze the effect of instance weighting in combination with regularized learning for domain adaptation, we first consider the role of regularization. We consider a standard supervised learning scenario with training data sampled from the same distribution $p(x, y)$ as the test domain. The generalization error of a learned model to unseen data can be decomposed into estimation error and approximation error (e.g., Nowak (2009)). We review this decomposition next.

For concreteness, consider learning a linear model $\beta$, where predictions are a function of

---

1Recall that $p(y|x)$ represents a conditional distribution of labels given feature vectors, $p(x)$ represents a distribution over feature vectors, and $p_s(\cdot), p_t(\cdot)$ represent distributions in the source and target domains, respectively. See Table 2.1 for a review of notation.
Let $\beta \cdot x$. Let:

$$R(\beta) \equiv E_{p(x,y)}\{l(\beta \cdot x, y)\}$$

represent theoretical risk as a function of $\beta$, where $l(\beta \cdot x, y)$ is a given loss function. Let

$$\beta^* \equiv \arg\min_{\beta} R(\beta) = \arg\min_{\beta} E_{p(x,y)}\{l(\beta \cdot x, y)\},$$

so $R(\beta^*)$ represents the smallest possible theoretical risk over all linear predictors $\beta$. Let $R^*$ be the risk of the best possible predictor (not necessarily a linear model), i.e., the Bayes risk. Assume $\hat{\beta}$ is learned from a random training sample of size $n$:

$$\hat{\beta} \equiv \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} l(\beta \cdot x_i, y_i).$$

Let $E\{R(\hat{\beta})\}$ be the expected generalization error for $\hat{\beta}$; note that the expectation is with respect to the training sample. The excess risk of this expected error is often decomposed as:

$$E\{R(\hat{\beta})\} - R^* = \left( E\{R(\hat{\beta})\} - R(\beta^*) \right) + \left( R(\beta^*) - R^* \right).$$

The first term is the estimation error and the second is the approximation error. Estimation error is the effect of using a random sample rather than infinite training data, and it is related to the variance of $\hat{\beta}$. Approximation error is the effect of using a constrained model space—in this case, it is the effect of using linear models when the best predictor might not be a function of $\beta \cdot x$ for any $\beta$.

Regularization adds a term $\gamma h(\beta)$ to the objective. For example, $L_2$ regularization uses $h(\beta) = ||\beta||^2$, so we have instead:

$$\hat{\beta}(\gamma) \equiv \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} l(\beta \cdot x_i, y_i) + \gamma ||\beta||^2,$$

for some $\gamma > 0$. The aim of regularization is to reduce variance in $\hat{\beta}$, which should lead to lower estimation error. However, regularization adds bias to $\hat{\beta}(\gamma)$ as an estimate of $\beta^*$. Let:

$$\beta(\gamma)^{\text{best}} \equiv \arg\min_{\beta} E_{p(x,y)}\{l(\beta \cdot x, y)\} + \gamma ||\beta||^2.$$ 

In other words, $\beta(\gamma)^{\text{best}}$ is the best value of $\beta$ that would be learned with regularized loss using infinite data, and $\beta^*$ is the best value without the regularization penalty. We define a
new “estimation error” under regularization, as the expected difference in risk between $\hat{\beta}(\gamma)$ and $\beta(\gamma)_{\text{best}}$. We also define an “approximation error,” as the difference in risk between $\beta(\gamma)_{\text{best}}$ and $\beta^*$. (We do not consider $R(\beta^*) - R^*$ since it does not depend on $\gamma$.) This allows us to write the following “excess risk” expression:

$$E\{R(\hat{\beta})\} - R(\beta^*) = \left( E\{R(\hat{\beta}(\gamma))\} - R(\beta(\gamma)_{\text{best}}) \right) + \left( R(\beta(\gamma)_{\text{best}}) - R(\beta^*) \right)$$

where the first term is the estimation error and the second term is the approximation error.

As a rough argument, we can say that for larger $\gamma$, the empirical loss term in the objective is less important, so the estimation error should be smaller, while the approximation error should be larger. Note that $E\{R(\hat{\beta}(\gamma))\}$ could be less than $R(\beta(\gamma)_{\text{best}})$, making estimation error negative, but this seems unlikely.

We now turn to the domain adaptation scenario, where we assume the training set is sampled from $p_s(x, y)$ and the test domain generalization error is with respect to $p_t(x, y)$. Now let:

$$R_t(\beta) \equiv E_{p_t(x,y)}\{l(\beta \cdot x, y)\},$$

and:

$$\beta_s(\gamma)_{\text{best}} \equiv \arg\min_{\beta} E_{p_s(x,y)}\{l(\beta \cdot x, y)\} + \gamma||\beta||^2$$

$$\beta_t(\gamma)_{\text{best}} \equiv \arg\min_{\beta} E_{p_t(x,y)}\{l(\beta \cdot x, y)\} + \gamma||\beta||^2$$

$$\beta^*_t \equiv \arg\min_{\beta} E_{p_t(x,y)}\{l(\beta \cdot x, y)\}.$$.

We now write the following “excess risk” expression:

$$E\{R_t(\hat{\beta}(\gamma))\} - R_t(\beta^*_t) = \left( E\{R_t(\hat{\beta}(\gamma))\} - R_t(\beta_s(\gamma)_{\text{best}}) \right)$$

$$+ \left( R_t(\beta_s(\gamma)_{\text{best}}) - R_t(\beta_t(\gamma)_{\text{best}}) \right)$$

$$+ \left( R_t(\beta_t(\gamma)_{\text{best}}) - R_t(\beta^*_t) \right),$$

where $E\{R_t(\hat{\beta}(\gamma))\}$ is taken with respect to the distribution of the source domain training sample. The first term represents estimation error, the second we call “domain mismatch error,” and the third represents approximation error. This bears some similarity to a target
domain error bound derived in Dredze et al. (2006), which has four terms: one related to the Bayes error rate, one in terms of $VC$-dimension, one related to domain distribution differences, and one related to expected performance differences of the best hypotheses.

Note that, as in Eqn. 4.1, it is possible to have negative estimation error, if $E\{R_t(\hat{\beta}(\gamma))\} < R_t(\beta_s(\gamma)_{best})$, but this seems unlikely. It is also possible to have negative domain mismatch error, if $R_t(\beta_s(\gamma)_{best}) < R_t(\beta_t(\gamma)_{best})$, which means that the value of $\beta$ that optimizes the regularized objective based on the source distribution gets lower target domain error than the value optimized using the target distribution. However, the sum of the domain mismatch and approximation error terms is never negative since $R_t(\beta_s(\gamma)_{best}) \geq R_t(\beta_*^t)$ by definition of $\beta_*^t$. (The total excess risk is also never negative.)

Now consider learning under covariate shift assumptions with instance weighting. Let $p(y|x)$ be the shared conditional label distribution in both domains, so $p_s(x, y) = p(y|x)p_s(x)$ and $p_t(x, y) = p(y|x)p_t(x)$. Assume that there exists a single unknown linear model $\beta^*$ that minimizes expected error $E_{p(y|x)p(x)}\{l(\beta \cdot x, y)\}$ for the given $p(y|x)$ and all distributions $p(x)$; therefore, $\beta_*^t = \beta^*$. Assume we know or have estimates of $w(x_i) = \frac{p_t(x_i)}{p_s(x_i)}$ for the samples $(x_i, y_i) \sim p_s(x, y)$. Adopting the adaptive parameter $\lambda$ from Shimodaira (2000), define:

$$\hat{\beta}(\gamma, \lambda) \equiv \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} w(x_i)^\lambda l(\beta \cdot x_i, y_i) + \gamma ||\beta||^2,$$

with $0 \leq \lambda \leq 1$. Now let:

$$\beta_s(\gamma, \lambda)_{best} \equiv \arg\min_{\beta} E_{p_s(x,y)}\{w(x)^\lambda l(\beta \cdot x, y)\} + \gamma ||\beta||^2,$$

and consider the excess risk $E\{R_t(\hat{\beta}(\gamma, \lambda))\} - R_t(\beta^*)$, which is a sum of three terms that we can define as:

- **Estimation error:** $E\{R_t(\hat{\beta}(\gamma, \lambda))\} - R_t(\beta_s(\gamma, \lambda)_{best})$ (4.2)

- **Domain mismatch error:** $R_t(\beta_s(\gamma, \lambda)_{best}) - R_t(\beta_t(\gamma)_{best})$ (4.3)

- **Approximation error:** $R_t(\beta_t(\gamma)_{best}) - R_t(\beta^*)$. (4.4)

We consider the impact of $\gamma$ and $\lambda$ on the target domain error. In Sugiyama et al. (2007), the use of a regularization term $\gamma$ or an “adaptive” parameter $\lambda$ were described as ways to
trade off “consistency” and “stability”, but they did not consider using both simultaneously. In Shimodaira (2000) for (unregularized) maximum likelihood fitting, the role of increasing λ is to decrease “asymptotic bias” (the difference between $\beta_{s}^{\text{best}}$ and $\beta_{t}^{\text{best}}$) at the cost of increasing the variance of $\hat{\beta}$. (Note that $E\{\hat{\beta}\} \neq \beta_{s}^{\text{best}}$ necessarily but it does converge to it for large $n$..) In terms of our definitions above, note that $\lambda = 1$ means $\beta_{s}(\gamma, \lambda)^{\text{best}} = \beta_{t}(\gamma)^{\text{best}}$, so we can say roughly that increasing $\lambda$ drives the domain mismatch error to zero. At the same time, increasing $\lambda$ increases estimation error, and does not affect approximation error. The role of $\gamma$ is to decrease estimation error at the cost of increasing approximation error, but its effect on domain mismatch error is more complicated. If $\gamma = 0$ then $\beta_{t}(\gamma)^{\text{best}} = \beta^{*}$, so approximation error is zero; furthermore, $\beta_{s}(\gamma, \lambda)^{\text{best}} = \beta^{*}$ for all $\lambda$, because of the assumption of a single linear model, so domain mismatch error is also zero, while estimation error can be large. If $\gamma = \infty$, then $\hat{\beta}(\gamma, \lambda) = \beta_{s}(\gamma, \lambda)^{\text{best}} = \beta_{t}(\gamma)^{\text{best}} = 0$ so estimation error and domain mismatch error are both zero, but approximation error is large.

As mentioned in Chapter 2, there exist several theoretical machine learning approaches to the domain adaptation problem that derive bounds on the target domain risk in terms of the empirical risk computed on the source domain examples (Cortes et al. 2010; Mansour et al. 2009a; Ben-David et al. 2007). The work of Cortes et al. (2010) appears to be especially relevant to the instance weighting scenario; it includes bounds on target domain risk in terms of instance-weighted empirical source risk and also in terms of statistics of the weights and complexity measures of the hypothesis space. These bounds suggest strategies for choosing weighting functions, and perhaps for choosing the hypothesis space or regularization weight. However, they only derive error bounds, and they do not appear to directly address the performance of a linear model $\hat{\beta}(\gamma, \lambda)$ chosen to minimize weighted, $L_{2}$ regularized risk, as a function of $\gamma$ and $\lambda$. In the following section, we experiment with a synthetic dataset by varying these parameters.

---

2This includes the possibility that domain mismatch error is negative, in which case setting $\lambda = 1$ increases it to 0.
4.3 Experiment with Instance Weighting on a Synthetic Dataset

We investigate the effects of $\lambda$ and $\gamma$ on a simple data set illustrating sample selection bias under the covariate shift assumption. Squared loss is used on a linear regression model. In order to design a scenario where regularization is useful, we encourage a high estimation error by using many noisy irrelevant features and a small training set. The $x$ variables are vectors of 102 dimensions. The $y$ variable is generated from the first two dimensions only:

$$y = 10x_1 + 10x_2 + \epsilon,$$

where $\epsilon \sim N(0, 10)$. The distribution of $x$ differs between the source and target domains. In the source domain, 90% of the $x$ vectors are generated with:

- $x_1 \sim N(0, 1)$
- $x_2 = 0$
- $x_3, \ldots, x_{102} \sim N(0, 1)$.

The other 10% follow:

- $x_1 = 0$
- $x_2 \sim N(0, 1)$
- $x_3, \ldots, x_{102} \sim N(0, 1)$.

In the target domain, 5% of the samples follow the first distribution and 95% follow the second. Therefore:

$$w(x) = \begin{cases} 
\frac{5}{90}, & x_1 \neq 0 \\
\frac{95}{10}, & x_2 \neq 0.
\end{cases}$$

The training set consists of 200 examples from the source domain; 1000 examples from the target domain are used to test. Although linear regression typically includes an intercept, for simplicity we assume the intercept is known to be 0 and consider learning only $\beta_{102}^{102}$.

Figure 4.1 shows the average error on the target domain over 100 runs with different training and test data, as a function of $\gamma$ and $\lambda$. The error function appears to be complicated (and non-convex). The plot shows that different values of $\lambda$ have different optimal values of $\gamma$. 
Figure 4.1: Empirical average error on target domain for a simulated ridge regression example, as a function of $\gamma$ and $\lambda$. Black dot shows the minimum error point.
Tuning both $\gamma$ and $\lambda$ appears to be useful: the lowest average error is 190 which occurs at $(\log(\gamma), \lambda) = (-0.1, 0.4)$. Without instance weighting, the lowest error is 197 which occurs at $\log(\gamma) = 0.5$. For larger $\gamma$, the optimal $\lambda$ is higher, indicating that more instance weighting is useful with high regularization. When $\gamma$ is very small (large estimation error), using instance weighting is not useful at all and merely increases the average error. We also note that, even without instance weighting, the optimal setting of $\gamma$ for minimizing target error differs from the optimal setting for minimizing source error—the latter is $\log(\gamma) = -0.3$.

We consider the effect of $\gamma$ and $\lambda$ on the target excess risk (defined as the sum of terms in Eqn. 4.2). We can compute theoretically the value of $\beta_{s}^{\text{best}}$ and $\beta_{t}^{\text{best}}$ (i.e., the values fit with infinite source or infinite target data). With $n$ training examples, the empirical estimate $\hat{\beta}$ for weighted ridge regression satisfies:

$$(X^T \cdot W \cdot X + \gamma I) \cdot \hat{\beta} = X^T \cdot W \cdot Y,$$

where $X$ is the $n \times 102$ data matrix (containing one training example $x_i$ per row); $Y$ is a length $n$ column vector containing the corresponding $y_i$ values for each row of $X$; $W$ is a $n \times n$ diagonal matrix with $W_{i,i} = \frac{1}{n} w(x_i)$; and $I$ is a $102 \times 102$ identity matrix. With infinite source samples, we can view the solution $\beta_{s}^{\text{best}}$ as solving:

$$A \cdot \beta_{s}^{\text{best}} = b,$$

where $A$ is a $102 \times 102$ matrix with $A_{l,k} = E_{p_s(x,y)}\{w(x)^\lambda x_l x_k\}$ (here $x$ is viewed as a random vector and $x_l, x_k$ are components of that vector). Using the known distributions of $x$ and the known value of $w(x)$, this results in:

$$A_{1,1} = 0.9 \cdot \left(\frac{5}{90}\right)^\lambda + \gamma$$

$$A_{2,2} = 0.1 \cdot \left(\frac{95}{10}\right)^\lambda + \gamma$$

$$A_{l,k} = 1 + \gamma, l = k, l \neq 1, 2$$

$$A_{l,k} = 0, otherwise.$$
Similarly $b$ is a length 102 vector with $b_i = E_{p_s(x,y)} \{ w(x)^\lambda x_i y \}$. So:

$$b_1 = 10 \cdot 0.9 \cdot \left( \frac{5}{90} \right)^\lambda$$
$$b_2 = 10 \cdot 0.1 \cdot \left( \frac{95}{10} \right)^\lambda$$
$$b_l = 0, \text{ otherwise.}$$

Therefore:

$$\beta_{s,1}^{\text{best}} = \frac{10 \cdot 0.9 \cdot \left( \frac{5}{90} \right)^\lambda}{0.9 \cdot \left( \frac{5}{90} \right)^\lambda + \gamma}$$
$$\beta_{s,2}^{\text{best}} = \frac{10 \cdot 0.1 \cdot \left( \frac{95}{10} \right)^\lambda}{0.1 \cdot \left( \frac{95}{10} \right)^\lambda + \gamma}$$
$$\beta_{s,i}^{\text{best}} = 0, i = 3 \ldots 102.$$

With $\lambda = 1$, we get the value of $\beta_t^{\text{best}}$:

$$\beta_{t,1}^{\text{best}} = \frac{10 \cdot 0.05}{0.05 + \gamma}$$
$$\beta_{t,2}^{\text{best}} = \frac{10 \cdot 0.95}{0.95 + \gamma}$$
$$\beta_{t,i}^{\text{best}} = 0, i = 3 \ldots 102.$$

Using the known distribution of $p_t(x,y)$ we can compute the expected squared loss expressions $R_t(\beta_s(\gamma, \lambda)^{\text{best}})$ (the expression is a ratio of polynomials in $\gamma$ with exponentials in $\lambda$) and $R_t(\beta_t(\gamma)^{\text{best}})$. Note that $R_t(\beta_s^*) = R_t(\beta_t^*) = E \{ \epsilon^2 \} = 100$. Using these computed values and the estimated total error from the simulation, we plot the estimation error, domain mismatch error, and approximation error in Figure 4.2 along with the total excess error from the simulation. As expected, as $\lambda$ increases the estimation error increases, and the domain mismatch error decreases (approximation error does not depend on $\lambda$). As $\gamma$ increases the estimation error decreases, and the approximation error increases, while the domain mismatch error is more complex: it increases and then decreases. Even for this relatively simple example, the total excess error appears to be a fairly complicated function of $\gamma$ and $\lambda$.

---

$^3$We used the symbolic math capabilities of Maxima: [http://maxima.sourceforge.net/](http://maxima.sourceforge.net/)
Figure 4.2: Estimation error, domain mismatch error, approximation error, and sum (target domain excess error, as defined in Eqn. 4.2) as a function of $\gamma$ for different values of $\lambda$. Estimation error has been estimated from the simulation; domain mismatch and approximation error are theoretical values.
One difficulty is that the optimal $\lambda$ and $\gamma$ for the target domain are different from the optimal values on the source domain. When no labeled target data is available for tuning, this presents a challenge for selecting the parameters. As discussed in Chapter 2, Sugiyama et al. (2007) proposed importance weighted cross-validation to select such parameters, while Shimodaira (2000) proposed a version of the information criterion evaluated on weighted source examples. The information criterion is computed in terms of the gradient of the loss, and it is theoretically justified as an estimate of the true generalization error.

4.4 Experiments with Instance Weighting on Real Datasets

4.4.1 Weight Estimation

As noted in Chapter 2, there have been several proposed methods for estimating the covariate weights when the densities are not known. Kernel density estimation (KDE) is a nonparametric method for estimating both $p_s(x)$ and $p_t(x)$, which is challenging in high dimensional spaces. The Kullback-Leibler importance estimation procedure (KLIEP) (Sugiyama et al. 2008; Tsuboi et al. 2009; Yamada and Sugiyama 2009) estimates weights $w(x)$ directly so as to minimize sample KL divergence between $p_t(x)$ and $w(x)p_s(x)$, using linear or log-linear combinations of kernels centered at examples, or Gaussian mixture models. A related approach uses an objective based on least-squares estimation rather than Kullback-Leibler divergence (Kanamori et al. 2009), and Sugiyama et al. (2010) estimates weights in a lower-dimensional subspace. In the discriminative modeling method of Bickel et al. (2007), the domain membership of each example is treated as a random variable $\sigma \in \{S, T\}$ and a kernel logistic regression model is learned to predict this variable. Then the weights for an example $x$ are given by:

$$w(x) = \frac{p(\sigma = T|x)p(\sigma = S)}{p(\sigma = S|x)p(\sigma = T)}.$$ 

Tsuboi et al. (2009) observe that the functional form of the weights is the same in their work and that of and Bickel et al. (2007), but only the optimization differs—in the former, the optimization is over just the train or test examples, while in the latter it is over all unlabeled train and test examples. Kernel Mean Matching (KMM) (Huang et al. 2007) is a nonparametric method for estimating the weights; a problem with this method, as pointed
out in Tsuboi et al. (2009), is that it has several tuning parameters but derives the weights in a transductive manner, so it could not be applied to held out data, e.g., for weighted cross validation (Sugiyama et al. 2007). The optimization procedure also requires computation of a kernel matrix and solution of a quadratic program, which may be an obstacle on large training sets.

We do not investigate all proposed weight estimation methods here, since our primary focus is not to compare different weight estimation methods but to see whether instance weighting might be useful on these datasets. We therefore selected two methods: (a) modeling of $p_t(x)$ and $p_s(x)$ using language models trained on each domain; and (b) discriminative modeling from Bickel et al. (2007). These methods are easily implemented and fast to run using out-of-the-box toolkits. We used the SRI Language Modeling Toolkit (Stolcke 2002) to estimate the language models and LIBLINEAR for the logistic regression model. A few previous papers have used language model ratios to estimate instance weights in NLP tasks. Liu and Zhao (2008) used unigram language models for domain adaptation on an opinion analysis task, along with other methods, but did not evaluate the effect of instance weighting alone. Plank and Sima’an (2008) used language-model-based instance weights for building subdomain parsers for improving parsing performance, but did not evaluate instance weighting for domain adaptation. By contrast, Xue et al. (2008) and Ling et al. (2008a) briefly reported instance weighting via kernel density estimation (KDE) (rather than language modeling) as a baseline for the 20Newsgroups (and other text classification) problems. We did not use that method here, since we believed that language modeling and discriminative prediction were more promising than KDE for this high-dimensional task.

4.4.2 Oracle Sentiment Classification Problem

We first investigate instance weighting in a contrived covariate shift scenario based on the “books→electronics” task. We take 999 reviews from the electronics domain and add them to the training set, so the “source” domain training set consists of 5501 book reviews and 999 electronics reviews; the target domain consists of the remaining electronics reviews, including 1000 test samples. This allows us to compare the two estimated weight methods
with “oracle” weighting: we assume that, for any electronics review \( x \):

\[
p_t(x) = p(x | \text{electronics}) \cdot p_t(\text{electronics})
\]

\[
p_s(x) = p(x | \text{electronics}) \cdot p_s(\text{electronics}),
\]

and similarly for book reviews. Since \( p_t(\text{electronics}) = 1 \) and \( p_s(\text{electronics}) = \frac{999}{6500} \), the oracle weights on the electronics training examples are:

\[
w(x) = \frac{6500}{999}, \ x \in \text{electronics}
\]

\[
= 0, \ x \in \text{books}.
\]

We compare performance of the oracle weighting with estimated weightings using language modeling and discriminative training. Ideally the estimated weight for each source training example should not come from a model learned on that example. Because of the limited amount of source data, we adopted a round-robin method in which the source examples were divided into batches; on each round, one batch was held out, and the weights for that batch were estimated on the remaining source data (i.e., by estimating a new source domain language model, or a new logistic regression model). For the language modeling method, we used bigram language models with Kneser-Ney discounting. The language models were built from, and evaluated on, the preprocessed counts composing the original corpus. (We derived sentence boundary tags from the unigram and bigram counts.) Note that we are trying to model \( p(x) \), but language models give probabilities of sequences, which do not necessarily represent mutually exclusive events (since one sequence can be contained in another). We therefore included a final symbol \( \text{EndOfReview} \) that occurs only at the end of the word sequence representing a review.

For the discriminative method, we did not experiment with the regularization parameter but kept \( C = 0.1 \) from the Sentiment Classification experiments in Chapter 3. In training the discriminative model we enforce equal fractions of source and target data to ensure that the prior \( p(\sigma = S) = 0.5 \); thus the weights are given by the ratio of the posterior probabilities for domain membership.

\[\text{We abuse notation in this section by using } x_i \text{ to represent both the feature vector seen by the classifier (unordered “bag-of-words” counts of unigrams and bigrams) and the word sequence in document } i. \text{ However, note that the language model does not use the full word sequence, but only counts of bigrams, so except for the } \text{EndOfReview} \text{ tag, it can be viewed as using a subset of the features in } x_i.\]
In both estimated methods, the range of the weights can be from 0 to infinity, so we set a maximum value of 10000; we then mapped the weights according to \( w(x)^\lambda \). The language modeling method produced many more weights with extremely large values, so for that method, we considered an alternative mapping based on \( w(x)^{1/L_i} \), where \( L_i \) is the number of words in review \( i \). Note that we can interpret this as an alternative computation of \( p(x) \).

Consider document \( i \) as a word sequence \( \{w_{i,0}, \ldots, w_{i,L_i}\} \). Since we are using a bigram language model, the method computes:

\[
\begin{align*}
    w(x_i) &= \left( \frac{\prod_{k=1}^{L_i} P_t(w_{i,k}|w_{i,k-1})}{\prod_{k=1}^{L_i} P_s(w_{i,k}|w_{i,k-1})} \right)^{\frac{1}{L_i}} \\
    &= \left( \frac{\prod_{k=1}^{L_i} P_t(w_{i,k}|w_{i,k-1})}{\prod_{k=1}^{L_i} P_s(w_{i,k}|w_{i,k-1})} \right)^{\frac{1}{L_i}}.
\end{align*}
\]

This is equivalent to using \( w(x_i) = \frac{p_t(x_i)}{p_s(x_i)} \) where \( p(x_i) \) is computed as the geometric average over n-grams in the document. This averaging may give a more stable model across different reviews \( i \).

In all cases, the final weights were used to train the sentiment classifier using weighted logistic regression training in LIBLINEAR. Figure 4.3 shows the performance of the different weighting methods on the target domain test set as a function of \( \gamma \) (note that \( \gamma \) was computed from LIBLINEAR’s parameter \( C \) using \( \gamma = \frac{1}{nC} \)). For each method, we show only the \( \lambda \) with the highest achieved accuracy and \( \lambda = 1 \), but note that we considered several values between 0 and 1. For this dataset, the baseline and oracle performance do not appear to be very sensitive to \( \gamma \) unless \( \gamma \) is too large. Both methods achieve the same performance at the optimal \( \gamma \), so overall there is no benefit to oracle weighting for this dataset. However, when \( \gamma \) is very large, training with oracle weighting does better than unweighted training on the full set, which can be explained by domain mismatch error. For small \( \gamma \) the performance using the full unweighted training set is close to the oracle weighting; the oracle weighting may do slightly worse because it has higher estimation error.

For the estimated instance weights, performance varies for the different methods. Most get worse than baseline at small \( \gamma \), presumably due to estimation error, and beat the baseline at higher values of \( \gamma \). For the discriminative method, we observe that the best value of \( \gamma \) with instance weighting is less than the best value for the unweighted baseline (which was also observed in the simulated scenario above). By contrast, the language modeling method
Figure 4.3: Instance weighting performance on the contrived covariate shift task in which the source domain contains both books and electronics reviews while the target domain contains only electronics reviews. Performance shown for baseline (no weighting); weighting based on discriminative modeling method for $w(x)$ using weight $w(x_i)^\lambda$ for example $x_i$; weighting based on language modeling (LM) method for $w(x)$ using $w(x_i)^\lambda$ or $w(x_i)^{1/L_i}$; and “oracle” weighting which includes only electronics examples weighted inversely to their proportion in the training set. The best value of $\lambda$ (in terms of best achieved accuracy over all $\gamma$) is shown for each method, as well as $\lambda = 1$. 
with large $\lambda$ has larger optimal $\gamma$ than the unweighted baseline.

Although oracle weighting at its best setting of $\gamma$ does not lead to any better result than the unweighted baseline, the estimated instance weights do, for their best settings of $\lambda$ and $\gamma$. This makes sense in light of our simulation experiments, which showed the benefits of both regularization and the weight adaptive parameter $\lambda$. In this case, the oracle weights cannot be tuned with $\lambda$ since they are either zero or nonzero. By contrast, the estimated weights are not theoretically “correct,” but they do vary continuously from 0 to $\infty$, so they permit balance between weighting and no-weighting. It is also possible that some book reviews are more like electronics reviews than others, which might be captured by the estimated instance weights.

Table 4.1 lists some statistics about the estimated weights on the training data (which includes both books and electronics reviews). Domain prediction accuracy is the percentage of book reviews with $w(x) < 1$ and electronics reviews with $w(x) > 1$. This is comparably high for both methods, indicating that the weights are emphasizing the right reviews. The effective sample size, from Shimodaira (2000), is the perplexity of the weights as a distribution—i.e., the normalized weights. (It ranges from 1 to 6500 in this case, and is equal to 999 for the oracle weights.) The “percentage at maximum ceiling” gives the percentage of weights that were mapped down to the maximum value (10000). It is quite high for the language model weights with $\lambda = 1$; these weights have a higher perplexity, as a consequence of so many large weights being set to the same value. The average value gives the empirical mean of the weights. Note that $E_{p_s(x)}\left\{\frac{p_t(x)}{p_s(x)}\right\} = 1$ if $p_t(x)$ has support within that of $p_s(x)$ (or less than 1 otherwise), so for $\lambda = 1$, theoretically we would expect the weights to have an empirical mean less than or equal to 1. In fact it is much larger, particularly for the LM method. This may explain why the language modeling with large $\lambda$ has a larger optimal value of $\gamma$ than the baseline. With very large weights the loss term in the objective can be made much larger than the regularization term, since $\gamma$ is fixed. (This would change if we normalized the weights by their sum.)

It is not completely understood why the average value for the LM weights is so large. Clearly there is a large amount of variation in the log likelihood of each review, which leads to some large differences between source and target log likelihoods. A few very small source
Table 4.1: Statistics on the weights estimated using the different methods.

<table>
<thead>
<tr>
<th></th>
<th>LM ($\lambda = 1$)</th>
<th>LM ($\lambda_i = \frac{1}{L_i}$)</th>
<th>Discriminative ($\lambda = 1$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain prediction accuracy (%)</td>
<td>98.9</td>
<td>98.9</td>
<td>96.9</td>
</tr>
<tr>
<td>Percent at maximum ceiling (%)</td>
<td>14.6</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>Effective sample size (out of 6500)</td>
<td>970 (15%)</td>
<td>4989 (77%)</td>
<td>55 (0.8%)</td>
</tr>
<tr>
<td>Average value</td>
<td>1475</td>
<td>0.62</td>
<td>24.9</td>
</tr>
</tbody>
</table>

likelihoods can cause the mean to increase, since $w(x)$ blows up for those examples. These issues may be a consequence of estimating probabilities over a high dimensional space. Overall, the language model method with $\lambda = \frac{1}{L_i}$ appears slightly superior to the other methods, because the best achieved accuracy is slightly higher, and because the weights are more stable and do not require tuning of $\lambda$. (We tried scaling with $\frac{1}{L_i}$ for $\lambda \in [0, 1]$, but it appears this additional scaling is not useful since the weights are already quite uniform.)

4.4.3 Performance of Instance Weighting on Real Tasks

We present results with instance weighting on the original Sentiment Classification and 20Newsgroups domain adaptation tasks. For these results we use the language modeling method to estimate $w(x)$ with $\lambda = \frac{1}{L_i}$. As noted above, there have been a few brief investigations of instance weighting for the 20Newsgroups data using KDE (Xue et al. 2008; Ling et al. 2008a); the reported results showed that it generally had little effect on the cross-domain performance. For the Sentiment Classification dataset, to our knowledge there have been no reported results using instance weighting in this setting, although Blitzer et al. (2008) and Mansour et al. (2009b) theoretically and experimentally investigated weighted combinations of data from multiple source domains or from both source and target domain.

Figure 4.4 shows the results of instance weighting on the 12 Sentiment Classification domain adaptation tasks, using the same data divisions as in Section 3.6.2. We tuned $C$ (note $C = \frac{1}{n^\gamma}$) on the books $\rightarrow$ electronics task. We present two sets of results: the first uses $C = 0.1$, the same value used in Section 3.6.2 (selected by tuning on same-domain
training); the second uses \( C = 10 \), the value chosen by tuning on the cross-domain books \( \rightarrow \) electronics task. For comparison, we also show the SCL results from Section 3.6.2.

In most cases, instance weighting has little or no effect on cross-domain performance. Using McNemar’s test with a threshold of \( p = 0.05 \), none of the instance weighting results is significantly different from the corresponding unweighted result with the same \( C \) value, except for one case (DVDs \( \rightarrow \) electronics with \( C = 0.1 \)) where it is significantly worse. The failure of instance weighting to improve performance may not be surprising since we do not expect that these domains satisfy the covariate shift assumption of shared support—for example, we do not expect to see any electronics reviews mixed into the book reviews. We observe that the weights are fairly uniformly small on the training data; the effective sample size ranges from 91% to 96% of the full training set size (in contrast with 77% in the oracle problem). On the books \( \rightarrow \) electronics task, the most highly weighted examples include seller feedback and reviews referring to the physical quality of books (such as textbooks, journals, or picture books) or their price. It also includes a few reviews that refer to computer applications, operating systems, or “instructions.” On the DVDs \( \rightarrow \) books task, the most highly weighted examples include reviews of documentaries, instructional videos, movie adaptations of books, and some possible (miscategorized) actual book reviews. On the DVDs \( \rightarrow \) kitchen task, the most highly weighted examples also include a lot of seller/company feedback (complaints about delivery, etc.), and some that are mostly agnostic about the product (along the lines of “I got it as a gift for someone and they liked it.”) While some of these examples appear relevant for the target domain, they are not generally representative of most target domain examples; in addition they are sometimes skewed toward one class or the other. The only effect of instance weighting on the DVDs \( \rightarrow \) kitchen task for \( C = 0.1 \) is that some more positive examples get misclassified as negative.

In general, for \( C = 0.1 \), instance weighting usually leads to a small performance drop. In fact, changing \( C \) has a larger effect than weighting in several cases. We observe that in some domain pairs, the classifier with \( C = 10 \) does better than the one with \( C = 0.1 \), while for other pairs it does worse. This confirms that the optimal \( C \) for the target domain error is not the same as for the source domain error; it also suggests that the optimal target domain \( C \) differs between the different domain adaptation pairs, which would make it difficult to
Figure 4.4: Performance of instance weighting on the original Sentiment Classification tasks. The value of $C$ was tuned to $C = 10$ on the B $\rightarrow$ E task. For comparison with the results in Section 3.6.2, we also present weighted and baseline results using $C = 0.1$, and the SCL results (which also used $C = 0.1$).
set the optimal $C$ using a development pair of domains. With a couple of exceptions, the improvement due to SCL is much larger than any improvement due to $C = 10$ (note that we did not tune $C$ for SCL).

Figure 4.5 shows the results of instance weighting on the 20Newsgroups tasks. For these tasks, since the features consist of unigrams with TF-IDF weighting, the language models were unigram models built from fractional counts—we multiplied the total count of each word by its IDF weight. We used Witten-Bell discounting rather than Kneser-Ney, in order to incorporate the fractional counts. We tuned $C$ on the “comp-vs-rec” task. As for the Sentiment data, we present weighted and unweighted results for $C = 100$, the value used in Section 3.6.3 from same-domain tuning, and $C = 10^5$, the value found by tuning the weighted training on comp-vs-rec. Our conclusions for 20Newsgroups are similar to Sentiment Classification. Using instance weights and increasing $C$ both have inconsistent, mostly small effects across tasks. Under McNemar’s test with a significance level of $p = 0.05$, several of the instance weighting results are significantly worse than their corresponding unweighted results, but only in one case—“rec-vs-sci” with $C = 10^5$—is the instance weighting result significantly better.

Target domain performance appears sensitive to $C$, but experiments again indicate that the best value cannot be selected using a “development” task since it varies among the tasks. We investigated the weighted cross-validation method in Sugiyama et al. (2007) as a way to choose $C$ for each target domain without having labeled target domain data. Using a held-out set from the source domain, we weighted the error for example $x_i$ according to $w(x_i)$, which was estimated by the language model method with $\lambda = \frac{1}{L_i}$. However, the weighted error value on the source domain held-out set was close to the unweighted value and did not appear to reflect the target domain error. The setting of $C$ therefore appears to be an open question; we note the complicated relationship between $\gamma$ and the target domain generalization error even in the ideal covariate shift scenario in Figure 4.2.

4.5 Conclusions

In this chapter we investigated instance weighting for training a regularized linear classifier. We first analyzed instance weighting combined with regularization, and considered the effect
Figure 4.5: Performance of instance weighting on the 20Newsgroups tasks. The value of $C$ was tuned to $C = 10^5$ on the “comp-vs-rec” task. For comparison with the results in Section 3.6.3, we also present weighted and baseline results using $C = 100$, and the SCL results (which also used $C = 100$).
of changing the adaptive weight parameter $\lambda$ and regularization weight $\gamma$ on the target domain excess risk. As is well known in the literature, under model mismatch conditions, setting $\lambda = 1$ leads to a decrease in target domain error with infinite training data, but increases estimation error due to finite training data. On the other hand, the regularization term weighted by $\gamma$ aims to decrease estimation error, at a cost of increased approximation error. Despite the apparently opposing goals of instance weighting and regularization, we illustrated a toy linear regression example where $\lambda$ and $\gamma$ can be jointly tuned to achieve lower expected target domain generalization error than possible by fixing $\lambda$ at 0 or 1 while tuning $\gamma$, or fixing $\gamma$ at 0 while tuning $\lambda$.

We next investigated methods for estimating the instance weights on a text classification task, including a discriminative approach based on Bickel et al. (2007) and a generative language modeling approach. In one experiment, weights derived from the language modeling approach with adaptive parameter $\lambda$ set by the total document length were more uniform and achieved slightly higher accuracy than other methods. However, on the real domain adaptation tasks (Sentiment Classification and 20Newsgroups), instance weighting based on this method had little effect. We observed that these tasks likely do not satisfy conditions under which instance weighting is expected to be useful; in particular, the training data is unlikely to contain many examples that are representative of the target domain. In comparison with the feature learning methods in Chapter 3 (SCL and LSA), instance weighting did not show either as large improvements or as large degradations in cross-domain performance. This suggests that feature learning methods are more powerful than instance weighting for these tasks, with greater potential to improve performance as well as to degrade it.

Finally, we observed that the effect of changing $\gamma$ is often larger than the effect of instance weighting. Since the optimal setting for $\gamma$ on the target domain is often different from the optimal setting on the source domain, it would be difficult to set this parameter without a labeled development set from the target domain. However, many “unsupervised” domain adaptation methods have tunable parameters as well, and would benefit from a labeled development set. If such a set were to be used, one might consider tuning the regularization parameter rather than more complicated adaptation approaches.
Chapter 5
CROSS-GENRE TRAINING FOR AUTOMATIC PROSODY CLASSIFICATION

5.1 Introduction

Prosody is the pattern of intonation and rhythm that accompanies words in speech. In English it functions to group words together into phrases, mark the boundaries of sentences, distinguish intent (i.e., question vs. statement) and convey emphasis, contrast, corrections, emotion, and uncertainty. The ability to automatically recognize prosodic phenomena is potentially useful for a number of speech processing applications, including automatic summarization, topic and sentence segmentation, machine translation, text-to-speech synthesis, and dialog act tagging. Significant research has been done on automatic annotation of prosody in speech, e.g. Wightman and Ostendorf (1994); Hasegawa-Johnson et al. (2005); Ananthakrishnan and Narayanan (2008). Most of the proposed methods rely on supervised learning, which requires a training set that has been hand-labeled for the classes of interest; the labeling process is time-consuming and expensive, requiring the efforts of a trained linguist. As a result, most researchers make use of one of a few available labeled corpora, such as the Boston University Radio News Corpus (Ostendorf et al. 1995). However, it is known that prosodic characteristics can vary by genre and style, for instance, between professionally read new speech and spontaneous conversational speech (Yuan et al. 2005; Cuendet et al. 2007a,b; Shriberg et al. 2000, 2009; Kolár et al. 2006a). Conversational speech is generally faster, with more backchannels and disfluencies; it has been shown that read speech has a higher proportion of words with pitch accents (Yuan et al. 2005), and that there are different acoustic and lexical indicators of sentence/dialog act boundaries between the genres (Cuendet et al. 2007b; Shriberg et al. 2000, 2009). In supervised learning to predict prosody, there is additionally the potential for variation between corpora due to different labelers and acoustic conditions.
In this chapter we investigate automatic prosody classification using training data with a different style than the test data. For our experiments we use two corpora: the Boston University Radio News Corpus (BU-RNC) (Ostendorf et al. 1995) and a section of the Switchboard Corpus (SWBD) (Godfrey et al. 1992) that has been labeled with accents and prosodic phrase boundaries (Calhoun et al. 2010). The BU-RNC consists of read news stories by professional radio announcers, and has been widely used for research into automatic prosody classification. The SWBD Corpus consists of spontaneous telephone conversations between strangers on assigned topics. We focus on two word-level binary classification tasks: presence vs. absence of a pitch accent on a word, and presence vs. absence of a major phrase boundary (break) after a word. We use a large standard set of acoustic-prosodic features extracted from the speech waveform and transcripts, as well as a small set of textual features, such as part-of-speech, extracted from the transcripts only. We compare cross-genre classifier performance with acoustic and textual features, then conduct domain adaptation experiments based on feature normalization, class proportion adjustment, instance weighting, co-training, and self-training. Although Chapter 3 showed that feature learning methods (SCL and LSA) can be effective for some kinds of domain adaptation, in this case the feature space is relatively low-dimensional, consisting mainly of continuous features and a few categorical features like parts-of-speech, which are assumed to occur in both domains. Therefore, those methods are unlikely to be useful here, where domain differences that occur should not be due to important features that are missing in the source domain, but rather to other factors like different class proportions or shifted feature distributions.

5.2 Background

Prosody includes a range of phenomena, and different standards have been developed for annotation. Our work is based on corpora annotated with the ToBI (Tones and Break Indices) standard for English (Silverman et al. 1992), which is designed to mark two different “tiers” of phenomena: tones, composed of boundary tones and pitch accents, and prosodic breaks. Pitch accents are defined by unusual stress or prominence, and can be associated

---

1This chapter is partly based on Margolis et al. (2010a).
with a word or syllable. In a multisyllabic word, they occur on a lexically-stressed syllable and are distinguished by energy, pitch contour, and duration lengthening (Kompe 1997). Some debate exists about whether accentuation exists in a continuum or in discrete categories. Also, a distinction has been made between contrastive (or emphatic) accents and non-contrastive (or normal) accents. The ToBI annotation allows for marking the presence or absence of pitch accents and for distinguishing a handful of accent categories characterized by patterns of high and low fundamental frequency (F0). (It does not distinguish between contrastive and non-contrastive accents). Prosodic break indices mark the “degree of decoupling between each pair of words” (Ostendorf et al. 1995), realized acoustically. This coupling is based on the grouping of words into prosodic “phrases”, which are believed to be related to syntactic structure and also, possibly, to distinct conventions related to the use of rhythm, stress, and accents (Wightman et al. 1992). In the ToBI system, full intonational phrase boundaries are marked with break index 4, and often correspond with longer pauses, breaths and/or syntactic boundaries such as periods or commas (Wightman et al. 1992). Break index 3 marks boundaries of “intermediate phrases,” smaller subunits which make up an intonational phrase; index 2 represents a smaller boundary, whereas index 1 represents the default coupling between words within an intermediate phrase, and index 0 marks no boundary. Each intermediate and intonational phrase boundary is accompanied by a boundary tone marked on the tones tier. Finally, disfluencies (such as hesitation and self-correction), are marked with ‘p’ on the break tier. These are rare in read speech, but common in spontaneous conversations.

The BU-RNC is one of the most widely used corpora for English prosody research, having been annotated with a version of ToBI. Figure 5.1 gives an example showing the annotation along both tiers. In this section of read speech, the words “proposition”, “two”, and “half” contain pitch accents of type high (H*) or downstepped high (!H*) marked roughly at the tone targets. An intermediate boundary occurs between “proposition” and “two”, and an intonational phrase boundary occurs at the end of “half” (which in this case corresponds to a sentence boundary). The symbols !H- and L-L% mark boundary tone types at the end of the phrase.

In our work, we focus on the task of recognizing the presence or absence of pitch accents
Figure 5.1: Example of the ToBI annotations for tones and breaks from BU-RNC. Top shows F0 and energy contours (from Praat [Boersma and Weenink 2007]); second level shows tones tier; third level shows breaks tier.
on words by collapsing all pitch accent types (such as H*, L*, !H*) to a single “accent” category, as is a common approach in the literature. For breaks, we also deal with a word-level binary classification problem by mapping indices 4 and above (major phrase breaks) to a “break” class.

Several studies have addressed annotation reliability for the ToBI system, including Silverman et al. (1992); Pitrelli et al. (1994); Ostendorf et al. (1995); in general, the reliability of coarse categories (such as pitch accent presence/absence and breaks) tends to be relatively high, but agreement on the pitch accent and boundary tone types tends to be lower. On BU-RNC, Ostendorf et al. (1995) reported agreement on pitch accent presence/absence in 91% of words, and on break index values in 95% of words (merging uncertain cases), in subsets of the corpus annotated by two labelers. On mixed speaking styles, Pitrelli et al. (1994) and Silverman et al. (1992) found lower rates of agreement (80% and 86-88% respectively for pitch accent, and 92% and 95% for agreement on break index within ± 1 level), where Pitrelli et al. included some newly trained labelers.

5.3 Related Work

5.3.1 Work on Automatic Prosody Labeling based on Supervised Learning

There has been a large amount of research on automatic detection and classification of prosodic events in English, such as those marked by the ToBI scheme. A comprehensive review of this work is beyond our scope, but we mention some examples to illustrate the range of approaches. Some work has focused on integrating prosody classification with ASR, in order to improve ASR performance (Hasegawa-Johnson et al. 2005; Ananthakrishnan and Narayanan 2009; Jeon et al. 2011); these approaches typically rely on a joint model of language, prosodic class, and prosodic acoustic features (such as pitch, energy, word, sub-word and pause duration). Others predict prosody from text only, for text-to-speech synthesis (Hirschberg 1993; Ross and Ostendorf 1996; Pan and McKeown 1999); these approaches use text-based features such as syntax, part-of-speech, n-gram and n-gram frequency, or discourse features. (Note that prediction of prosody classes from acoustic and textual features is also useful for training speech synthesizers.) Finally, some approaches treat prosody
recognition as a post-ASR step, for e.g., sentence/dialog act segmentation, detection of focus, or dialog act tagging. Related work in this realm typically uses a combination of acoustic-prosodic and text-based features. A wide variety of approaches have been used. Sequence modeling methods include HMMs (Wightman and Ostendorf 1994; Kompe 1997), conditional random fields (Gregory and Altun 2004; Levow 2008), and joint sequence models for prosodic events and lexical symbols (Liu et al. 2006; Ananthakrishnan and Narayanan 2008; Chen et al. 2004). Other work has performed independent classification of events at the word or syllable level, often with contextual features based on the surrounding words or syllables. Many different classifiers have been used in these works, including ensemble methods (Sun 2002; Rosenberg 2010), SVMs (Levow 2005; Rosenberg 2010; Jeon and Liu 2009a), logistic regression (Rosenberg 2010), neural networks (Kompe 1997; Jeon and Liu 2009a), decision trees (Wightman and Ostendorf 1994; CART (Wang and Hirschberg 1992), maximum entropy models (Rangarajan Sridhar et al. 2008), and Gaussian mixture models (Chen et al. 2004; Ananthakrishnan and Narayanan 2008).

Different labeling tasks have been addressed in these works, including binary pitch accent detection, pitch accent type classification, boundary tone detection and classification, and break classification. For pitch accent experiments, some work has performed classification on the syllable level and other work on the word level. On the widely-used BU-RNC, published work has used different subsets, which may include one or multiple speakers. Previous work on tasks related to ours has been summarized in tables by Rosenberg (2009) (for pitch accent detection) and Rangarajan Sridhar et al. (2008) (for pitch accent, boundary tone, and break detection). The best performing results for word-level pitch accent detection on speaker-independent settings of the BU-RNC are around 85-86% (Rangarajan Sridhar et al. 2008; Rosenberg 2009; Levow 2008). On subsets of Switchboard, Gregory and Altun (2004) and Nenkova et al. (2007) each reported performance around 76%. For detection of intonational phrase boundaries, on BU-RNC, the best reported results are around 93% (Chen et al. 2004; Levow 2008). A similar result was reported by Rangarajan Sridhar et al. (2008) when detecting combined intonational/intermediate boundaries. We note also that segmenting speech into sentence units or dialog acts is a related task to detection of symbolic prosody events; in particular, intonational phrase boundaries should usually include sentence
unit boundaries. These related tasks can be performed using similar features and models as symbolic prosody classification (Liu et al. 2006; Shriberg et al. 2000), including pitch, energy, and other prosodic acoustic features. See Chapter 7 for a review of related work on segmentation and punctuation annotation.

5.3.2 Work on Automatic Prosody Labeling in Different Genres

There has been some work comparing the utility of different feature sets for prosodic event and sentence boundary detection in different genres. Yuan et al. (2005) explored the use of different textual features for predicting pitch accent in two corpora each of read speech (including BU-RNC) and spontaneous speech (including Switchboard); they considered textual features such as accent ratio, part-of-speech (POS), and unigram/bigram probabilities. They found that content/function word status was a better predictor in read speech, with content words more reliably accented and function words not accented, and they noted major domain differences in the proportion of accented words in certain POS categories. In experiments with feature selection for machine learning, they compared the best feature subsets for each corpus, finding that the best sets (for in-genre training) were slightly different but that a good universal set could be found that did almost as well. On the task of sentence and dialog act boundary detection, several papers by the group of Shriberg, Cuendet and colleagues have compared feature utility and distributions across read news, meeting, and conversational speech genres. Cuendet et al. (2007a) compared the relative usefulness of language and acoustic feature types for prediction in different genres, with and without the presence of speech recognition errors. In the work of Shriberg et al. (2000), acoustic-prosodic models were combined with language models for sentence segmentation, and their relative usefulness was compared on broadcast news and Switchboard. Speaker variability studies on pitch accents have also been conducted by Yuan et al. (2005); Yoon (2007); Jeon and Liu (2010), and on boundary detection by Kolár et al. (2006a).

In our work, we do not compare utility of features for training and testing a classifier within each genre. Rather, we are interested in the problem of cross-genre training, where the training set is from a different spoken language domain than the test set. Some
experiments along these lines have been conducted by Rosenberg (2009) for pitch accent detection between BU-RNC and the Boston Directions spontaneous and read speech corpus, and Cuendet (2006) for boundary detection, between broadcast news and meetings corpora. While training on labeled data from a different corpus, we would like to use unlabeled data from the target corpus for adaptation. Our work is inspired partly by Shriberg et al. (2009), who focused on the task of dialog act segmentation in multiple genres, using a similar feature set to ours. They included a detailed comparison of acoustic feature distributions in a meeting corpus and a broadcast news corpus, observing many similarities in the shapes of the distributions, and proposing that these similarities could be used for cross-genre training and adaptation; for instance, they suggested the possibility of feature normalization or automatic adjustment of class proportions.

5.3.3 Work on Semi-supervised Learning and Adaptation

Recent efforts to use unlabeled data for prosodic event detection or sentence/dialog act segmentation have been described in several works (Jeon and Liu 2009b; Prahallad et al. 2010; Chen et al. 2010; Huang and Hasegawa-Johnson 2008b; Levow 2006; Fernandez and Ramabhadran 2010; Ananthakrishnan and Narayanan 2009, 2006; Guz et al. 2007). Levow (2006) applied sophisticated clustering and semi-supervised methods (based on spectral clustering and manifold regularization) to classify pitch accent in English and tone in Mandarin, using acoustic features. The clustering (based on assigning each cluster to the most frequent label) achieved results close to supervised performance on a single speaker of the BU-RNC corpus. A different set of clustering experiments were conducted by Ananthakrishnan and Narayanan (2006), for binary accent and boundary detection. In an approach reminiscent of co-training, they use clusters based on acoustic features to refine Naive Bayes models based on lexical/syntactic features, and vice versa. This approach leads to good performance (although worse than supervised performance) on both tasks using a multiple-speaker set of the BU-RNC. Co-training for sentence segmentation was investigated by Guz et al. (2007), using lexical and acoustic features as the two different “views.” On the Meeting Recorder Dialog Act corpus (Shriberg et al. 2004) (MRDA), their method led to an improvement,
particularly when the amount of initial labeled data was small. It also compared favorably with self-training using either feature set (which gave very little improvement). A modified version of this co-training approach was investigated by Tur (2009), using model adaptation to incorporate new data on each step; this gave superior performance on MRDA dialog act segmentation. Co-training for prosodic event detection (ToBI break classes and pitch accents on BU-RNC) was investigated by Jeon and Liu (2009b). They proposed a new example selection method that takes into account both label confidence and differences in predictions between the classifiers, to try to ensure that the examples are informative. Their method led to F measure performance approaching supervised baselines, while using only 3% of the supervised labeled data. It also outperformed supervised training using equivalent amounts of labeled examples during early iterations.

In other work using unlabeled data, Huang and Hasegawa-Johnson (2008b) described a bootstrapping/knowledge-based approach for building a prosodic break detector on Mandarin syllables without explicitly labeled data. Initially, all syllable boundaries within short words are taken as examples of the non-break class, and syllable boundaries with silent pauses are taken as examples of the break class. These are combined with unlabeled data via Gaussian mixture models fit with EM, leading to performance approaching a supervised baseline. They also consider English break detection on BU-RNC, using an initial labeled set combined with unlabeled data, which leads to performance close to a supervised result. Related work was described in Prahallad et al. (2010). They first hypothesized an initial set of phrase breaks using pause regions, then bootstrapped the initial system to incorporate more unlabeled data. The detected phrase breaks were incorporated into a speech synthesis system for the Telugu language, with improved performance compared with baselines which did not use phrase breaks or used only text-based punctuation marks. (They also reported that cross-lingual bootstrapping for phrase break detection from English to Telugu did not perform well.)

There has been a smaller amount of work focusing on unsupervised domain adaptation for prosodic event detection. Fernandez and Ramabhadran (2010) utilized self-training to adapt a CRF based on labeled utterances from a single speaker outside the BU-RNC corpus to a multi-speaker set of the BU-RNC corpus. On the pitch accent detection and break
detection tasks, this led to an improvement when only small amounts of labeled data were available, but with larger amounts it degraded performance. Chen et al. (2010) described a method for adapting a text-feature-based phrase break prediction system by incorporating domain-matched web data, using agreement between the predicted phrase breaks and the punctuation in the web data. The system was used to adapt a speech synthesis system for Mandarin news data to a new time period. In the work of Ananthakrishnan and Narayanan (2009), a prosodically-labeled seed corpus (BU-RNC) was used to bootstrap data from a second non-labeled corpus in order to augment a prosodic event language model and a prosodic acoustic model, for detecting pitch accent. However, note that the goal there is to improve performance on the same domain as the seed set, not to perform well on a new domain for which no labeled data exists. Jeon et al. (2011) used co-training with separate lexical and acoustic models as an adaptation strategy for pitch accent modeling, with the labeled training set from BU-RNC and the unlabeled adaptation set from both BU-RNC and a different broadcast news corpus. Their work proposed a method for improving ASR performance using prosody modeling, and they showed that co-training adaptation did decrease word error rate on both corpora using their proposed method.

Supervised adaptation approaches for sentence/dialog act boundary detection have also been described (Cuendet et al., 2006; Cuendet, 2006; Kolář et al., 2010). Cuendet et al. (2006) and Cuendet (2006) showed success with supervised model adaptation between Switchboard and a meetings corpus with a variety of methods, including model combination and using predictions of the cross-domain model as a feature. Kolář et al. (2010) described similar methods for adaptation between different speakers within the same corpus.

In our work, we look at unsupervised adaptation between a multi-speaker conversational corpus and a multi-speaker news corpus. This differs from Fernandez and Ramabhadran (2010); Chen et al., (2010), who did not consider conversational speech. We also consider domain adaptation methods not used in these works, such as instance weighting, co-training, prior adjustment, and normalization.
5.4 Methods

5.4.1 Data

The BU-RNC contains 7 speakers altogether (4 male, 3 female), with stories from each speaker divided into “radio” and “labnews” sections. The “labnews” portion of the corpus, intended to study inter-speaker variation, consists of the same stories read by multiple speakers. The corpus comes with reference transcripts and word times. Labels from the tone tier are associated with tone target times; there can be zero or multiple tone categories within a word. Labels from the breaks tier are associated with word end times. As mentioned above, we follow a common approach of collapsing all pitch accent categories to a single “accent” label, resulting in a binary word classification problem (presence/absence of pitch accent). Similarly, for breaks, we map indices 4 and above (major phrase breaks) to a “break” class. For both the accent and break detection tasks, we perform binary classification of each word, where, in the case of breaks, the class we are detecting is the break status of the boundary after the word.

The conversational corpus that we use is from Switchboard (SWBD). The Switchboard-I telephone speech corpus consists of short 2-sided telephone conversations, each on an assigned topic. The corpus comes with reference words and times, and is segmented into speakers and utterances. A portion has been annotated for prosody using a slightly different scheme from the one in BU-RNC. There are three pitch accent categories (“full”, “weak”, “none”). We mapped both “full” and “weak” accent labels to the pitch accent category. In addition, SWBD has separate break annotations for disfluencies and backchannels. It might be desirable to use separate disfluency and backchannel classes, but since we are interesting in being able to transfer a classifier trained on BU-RNC, which has no disfluencies or backchannels, it is desirable to use a common set of classes between the corpora. In the results presented here, we mapped the backchannels to the break class. Disfluencies were mapped according to their accompanying break label.

---

2The break annotations that we use came originally from University of Washington; the accent annotations that we use were based on annotations from University of Washington that were converted to match the format and standards of University of Edinburgh.
(i.e., the disfluency marks were ignored).

We derived training sets, development sets, and test sets for each corpus, which were used in the following experiments. The BU-RNC training set for both tasks consists of annotated portions of the “radio” stories from three speakers (named f2b, f1a, m2b). The development set consists of “labnews” stories from speaker f2b (about 2.1k words). The test set shares no speakers with the training or development sets, and consists of “labnews” stories for speakers f3a, m1b, and m3b. The SWBD training sets are drawn from 7 conversations. The SWBD development set is drawn from a held-out set of 4 conversations, and the test set is drawn from a different held-out set of 4 conversations. The test set includes 7 distinct speakers and shares no speakers with the training or development set. After the work of Margolis et al. (2010a) we discovered that some of the SWBD files were not entirely annotated for breaks. After excluding the apparently-unannotated segments, the SWBD training set had 9k words, which was smaller than the BU-RNC training set of 13k words. In order to have training sets that were better matched in size, we included only a portion of the stories (s01-s18 and part of s19) for speaker f2b in the BU-RNC training set (this also has the effect of making the BU-RNC training set more balanced among the three speakers).

Characteristics of the datasets are shown in Table 5.1. Note that there exists a significant amount of variation in proportions of the positive class across data sets in the same corpus, although BU-RNC on the whole has a higher proportion of accented words than SWBD.

5.4.2 Features

We use 64 “acoustic” features extracted using the waveforms, based on measurements of duration, energy, and F0 values. The features are based on those described in Shriberg et al. (2000); Liu et al. (2006). They were derived from a phone-level forced alignment of the transcripts with the waveforms, and are associated with a word or the post-word boundary, with some representing differences between this word and the next, or in a window around the boundary. Duration features include word and pause durations as well as subword durations such as average/maximum/last vowel, last rhyme, and average/maximum phone. The same set of acoustic features was used for both the pitch accent and phrase break experiments.
Table 5.1: Characteristics of the train, development, and test sets used in this chapter. BU-RNC is divided into news stories; SWBD is divided into conversations between two people. The number of speakers is the number of distinct speakers in the set (in SWBD, speakers often participate in multiple conversations).

<table>
<thead>
<tr>
<th>dataset</th>
<th># stories or convers.</th>
<th># speakers</th>
<th># words</th>
<th>% accents</th>
<th>% breaks</th>
</tr>
</thead>
<tbody>
<tr>
<td>BU-RNC train</td>
<td>33</td>
<td>3</td>
<td>9657</td>
<td>57.2</td>
<td>21.4</td>
</tr>
<tr>
<td>BU-RNC dev</td>
<td>4</td>
<td>1</td>
<td>2112</td>
<td>49.6</td>
<td>19.8</td>
</tr>
<tr>
<td>BU-RNC test</td>
<td>12</td>
<td>3</td>
<td>4799</td>
<td>54.9</td>
<td>14.9</td>
</tr>
<tr>
<td>SWBD train</td>
<td>7</td>
<td>13</td>
<td>9677</td>
<td>40.5</td>
<td>18.6</td>
</tr>
<tr>
<td>SWBD dev</td>
<td>4</td>
<td>8</td>
<td>3920</td>
<td>45.4</td>
<td>20.5</td>
</tr>
<tr>
<td>SWBD test</td>
<td>4</td>
<td>7</td>
<td>5570</td>
<td>46.9</td>
<td>23.1</td>
</tr>
</tbody>
</table>

Although some previous work has included features derived from lexical stress information, this requires a stress dictionary which we did not have available for the SWBD data, so we did not use those features here. The features are normalized in various ways, based on parameters computed at the story level for BU-RNC and at the conversation-side level for SWBD. Most F0 and energy features can take missing values due to issues with computation of pitch and energy contours (for example, zero or negative values result in undefined log features). Duration features over rhyme or vowel units can also occasionally be missing (or, more precisely, undefined) when there are no vowel phones (e.g., in some pronunciations of backchannel or filled pause words). The handling of these missing/undefined values depends on the classifier, and is described more below (we do not distinguish between missing and undefined). The BU-RNC stories are contained in audio file segments of several sentences each; exact timing between these files is unknown but we assumed a value of 5 seconds. We excluded segments marked as noisy. The SWBD data was processed in segments corresponding to utterances as defined in the MS State transcripts; we excluded segments that
were less than 1 second in duration, which are mostly words like “uhhuh” and “right.” All of the acoustic features are numeric, except for two categorical features representing pitch and energy contour patterns.

The “textual” features were extracted from the transcripts including marked sentence-unit boundaries. We used the BU-RNC “.txt” transcripts and the MS State SWBD transcripts, which include this information. The textual features include part-of-speech (POS) labels for the current, previous, and next word, derived from the MXPOST POS tagger (Ratnaparkhi 1996). Each sentence unit was input to the tagger separately, and special “start” and “final” tokens were added as previous-word POS and next-word POS for the first and last word in each. No tokenization was applied to these sentences (doing so might cause problems with matching tagger output with original words), and they were applied with all capitalized words and no punctuation, to match the POS model. The tagger outputs around 40 distinct tags (based on the Penn-Treebank tagset (Santorini 1990)).

For the pitch accent task, additional textual features include log unigram, bigram, and backwards bigram probability derived from a separate language model trained from broadcast news and talk shows, with out-of-vocabulary words given value $-100$. We also used the accent ratio feature from Nenkova et al. (2007), which represents the fraction of time a word occurs accented in the training set. (As in Nenkova et al. (2007), only words with significant evidence of accent probability different from 0 are given the computed accent ratio; all others and out-of-vocabulary words are given the default value of 0.5.) Bigram and backwards bigram probabilities are computed using sentence boundary tokens at the edges of marked sentence units. For the break task, the textual feature set includes the three POS features plus two novel features: break ratio defined analogously to accent ratio, and POS bigram break ratio, defined analogously for the POS bigram formed from the current and next words. The “ratio” features require a labeled training set, so in the cross-corpus training experiments described below they were computed for the test set based on the ratios in the cross-corpus training set. This more accurately reflects the situation in which

---

3The MS State transcripts were downloaded from [http://www.isip.piconepress.com/projects/switchboard/](http://www.isip.piconepress.com/projects/switchboard/).

4We used a version of the tagger, trained on Treebank-3, including Switchboard and Wall St. Journal data.
no labeled training data is available in the test corpus.

5.4.3 Classifiers

We use two classifier packages in our experiments: icsiboost (Favre et al. 2007), an implementation of BoosTexter (Schapire and Singer 2000), and LIBLINEAR (Fan et al. 2008) in logistic regression mode. BoosTexter is based on the AdaBoost algorithm with single-feature decision stumps as the base classifiers; it easily incorporates both continuous and categorical features, and can handle missing feature values. Each decision stump asks a question about a particular feature—for categorical features, it checks if the feature is equal to a particular value, not equal to that value, or missing; for continuous features it checks if a value is over or under a threshold or missing. During training, the decision stumps are learned iteratively in order to minimize costs, where the weights on the current round reflect errors in the previous round. At test time the discriminant value is determined by a weighted sum of values from the decision stumps.

For the logistic regression classifiers, we represent categorical features (part-of-speech and the two acoustic categorical features) with “dummy coding” using \( k - 1 \) binary variables for a feature that has \( k \) possible values. We also first transformed the part-of-speech categories to a smaller set of nine (such as noun, verb, interjection, preposition/conjunction). The logistic regression approach can perform badly if the features have very different ranges, so all features are scaled using max/min values computed from the training set, such that each feature has range \([0, 1]\) in the training set. Missing features were set to 0. With acoustic and textual features together, BoosTexter sees 71 features for accent classification and 69 for break classification; logistic regression (using the dummy coding) sees 114 features for accent detection and 112 features for break detection.

Both classifiers have training parameters—the number of boosting rounds for BoosTexter, the training error cost \( C \) for logistic regression—that are typically set by cross validation or on a held-out development set. In our case, we would like to do cross-corpus training experiments without assuming the presence of any labeled data in the target corpus, even a development set. In addition, we would like to be able to make some comparisons about
what feature types transfer better across domains, which would be difficult if different training parameters were used for different feature sets. Therefore, for these experiments we fix $C = 10$ and the number of BoosTexter iterations at 2000 for all experiments. These settings were found to be optimal or close to optimal, when tuning on the development sets from the same corpus as the training set, using all features. They are not necessarily optimal when using subsets of features, which should be kept in mind when interpreting the results below.

For logistic regression, we also found it necessary to set a smaller termination tolerance for training ($\epsilon = 0.0001$ rather than the default 0.01), as our initial set of experiments indicated some problems with convergence using the larger value.

5.5 **Cross-Corpus Training**

We first compare the performance of models trained on the same-corpus training set with models trained on the opposite corpus, using different feature sets (acoustic/textual) and models (BoosTexter/logistic regression). We also consider the performance of models trained on a concatenated version of both training sets; this allows us to investigate whether it is possible to model both corpora simultaneously, thus giving some indication of the nature of domain mismatch. When training on both training sets, the “ratio” features for each example were computed only from the same-corpus part of the training set. These results are shown in Figure 5.2.

We first observe that for both tasks, error is almost always much worse for SWBD than for BU-RNC, indicating that the tasks are more challenging on the SWBD corpus. However, overall the cross-corpus classifiers do quite well—using the full set of features, in all cases the cross-corpus error rate is no more than 5 percentage points worse than the in-corpus error rate.

For accents, we observe that in both directions and with all feature sets there is an increase in error due to cross-corpus training compared with in-corpus training, but the

---

5This could be considered cheating because the problem was noticed as a result of unstable performance on the test set, but we emphasize that we did not set this value to optimize performance, and in principle it could have been discovered on a development set. This is a parameter used for numeric optimization; it does not affect the function being optimized.
greatest increase in error occurs with the textual feature set only. This seems to be due to a variety of factors: in some cases the POS features suffer substantial degradation, while in other cases the accent ratio feature does not transfer well across corpora. This is consistent with observed differences by Yuan et al. (2005) between read and spontaneous speech, in terms of the relationship between accented words and POS categories, and in terms of accent ratio distributions. For accent detection, acoustic features are more useful for than textual features for the in-corpus classifiers. Although the in-corpus classifiers benefit slightly from the addition of textual features, the cross-corpus classifiers do not—their performance with acoustic and textual features is similar to their performance with just acoustic features.

For breaks, on SWBD, the BoosTexter cross-corpus classifier with acoustic features actually outperforms the in-corpus classifiers. The fact that BU-RNC can outperform SWBD on held-out SWBD conversations in some cases might be due to noisiness of the SWBD annotations or lower quality of acoustic features in the SWBD utterances—perhaps BU-RNC provides a cleaner set of training utterances. It might also be due to differences between the SWBD training and test conversations, with the BU-RNC training set possibly a better match to the held-out SWBD conversations. (Note in particular that the BU-RNC training set has a larger proportion of breaks than the SWBD training set, which better matches the proportion in the SWBD test set and might permit the classifier to learn the break class better.) In the opposite direction, the SWBD-trained classifiers outperform the BU-RNC-trained classifiers on the BU-RNC test set using textual features, although we note that they detect fewer breaks. In all cases except BoosTexter on BU-RNC, the cross-domain performance on breaks is better with both feature sets than with only one set.

There is some variation in the performance of the two classifiers. On the accent task, logistic regression generally does better with acoustic or acoustic+textual features, while BoosTexter does better with textual features alone. On the break task, their performance is more mixed. One factor for BoosTexter's better performance in some cases is that it uses a more fine-grained set of POS categories. We expected that BoosTexter might be superior to logistic regression for modeling both corpora simultaneously, since it is capable of fitting a more complex (non-linear) decision boundary. As discussed in Chapter 2, model families that are capable of minimizing loss on both domains simultaneously will not suffer
from sample selection bias (see Section 2.3). However, in general this was not observed. On the accent task, we observed that performance of both classifiers in the domain-general setting (i.e., training on both corpora) is comparable to their performance in the domain-specific setting (training on only the corpus-matched training set). We used McNemar’s test with a threshold of $p = 0.05$ to measure statistical significance between the domain-specific and domain-general classifiers. For accents, the domain-general classifier was never significantly worse than the domain-specific classifier, except for one case (BoosTexter with textual features on BU-RNC). This suggests that, for the accent task, both classifiers can generally model the decision boundaries simultaneously on both corpora. On the break task, this was not always true. On BU-RNC using acoustic features or acoustic+textual features, the domain-general classifiers were significantly worse than the domain-specific classifiers, for both BoosTexter and logistic regression. This suggests neither classifier can completely model both decision boundaries simultaneously in these cases. (On SWBD, the domain-general classifier was significantly worse in one case—BoosTexter using textual features.)

We note that Rosenberg (2009) compared cross-corpus performance between BU-RNC and a different spontaneous speech corpus (Boston Directions Corpus) for accent detection. He reported error rates on BU-RNC around 14% for in-domain training, and around 18% when training on the spontaneous corpus (his Table 3.16), using different training and test sets, and different classifiers and features. He also considered a read version of the Boston Directions spontaneous corpus (consisting of the same speakers reading a transcript of their spontaneous speech), and found that the Boston Directions read and spontaneous domains were more similar to each other than either was to the BU-RNC; the cross-corpus training degradation appeared to be due to corpus or speaker differences (Boston Directions vs. BU-RNC) rather than speaking style (read vs. spontaneous). Our experiments here cannot determine whether the observed cross-corpus degradation could be attributed to differences in speaking “style” that would generalize to other read and spontaneous corpora, since there are other possible sources of corpus difference (e.g., annotation and processing differences).

Note that, in addition to excluding the unannotated data and using smaller training sets, as described above, our approach here differs from our approach in Margolis et al.
Figure 5.2: Baseline classification error rates for different feature sets (“A”=acoustic, “T”=textual features); training on either BU-RNC, SWBD, or both sets concatenated; training a BoostTexter (“Boost”) or logistic regression (LR) model.
in the following respects: (i) we use 2000 rounds of boosting, rather than 1000 rounds as in that work (where the value was not tuned); (ii) in Margolis et al. (2010a) we set pause duration features greater than 5 seconds to be missing, while here they are kept; (iii) we compute the bigram log likelihoods differently for words at the edges of sentences; (iv) we performed an additional normalization step on some BU-RNC words. Our work in Margolis et al. (2010a) also sampled the SWBD training set from a much larger number of conversations, which may give better performance for the same size training set, but we did not fully explore this effect.

5.6 Adaptation Approaches

The goal of unsupervised domain adaptation is to use unlabeled test domain data along with labeled cross-domain training data to build a classifier tuned to the test domain. As discussed in previous chapters, several approaches have been suggested in the machine learning literature, which make various assumptions about the nature of the domain mismatch. One possible assumption is that of sampling selection bias: the domains have the same posterior label distribution \( p(y|x) \) for label \( y \) given the features \( x \), but the training domain examples represent a biased sample from the unconditional distribution \( p(x) \); this suggests approaches such as instance weighting (Shimodaira 2000). A different assumption is that domains share class generative distributions \( p(x|y) \) but differ in class proportions \( p(y) \); this suggests that automatic adjustment of the class priors may help, as in Saerens et al. (2002). Another assumption is that features have been scaled or shifted between domains, which might be corrected by normalization. Finally, self-labeling approaches like co-training and self-training might be effective if the source and target distributions are similar enough that the source-trained model can reliably label some target data (with high confidence), and if the target distribution is amenable to self-labeling from these samples. A goal of our experiments is to investigate whether unsupervised domain adaptation approaches based on these assumptions might be effective for accent and phrase boundary classification across corpora.

For the adaptation experiments described below, we used both feature sets (acoustic and textual). In general, we used the training set from the source domain as the labeled data
Table 5.2: Corpus-level feature normalization results (classification error rate %) for the BoosTexter classifier using acoustic+textual features; “norm. all” normalizes all numeric features, “norm. acoustic” normalizes only the acoustic numeric features. Normalization generally harmed cross-corpus training performance.

<table>
<thead>
<tr>
<th></th>
<th>Accents</th>
<th></th>
<th>Breaks</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>train</td>
<td>test</td>
<td>train</td>
<td>test</td>
</tr>
<tr>
<td></td>
<td>SWBD</td>
<td>BU-RNC</td>
<td>SWBD</td>
<td>BU-RNC</td>
</tr>
<tr>
<td>baseline</td>
<td>16.0</td>
<td>26.0</td>
<td>10.0</td>
<td>14.2</td>
</tr>
<tr>
<td>norm. all</td>
<td>29.0</td>
<td>30.5</td>
<td>10.6</td>
<td>14.2</td>
</tr>
<tr>
<td>norm. acoustic</td>
<td>17.1</td>
<td>27.0</td>
<td>10.9</td>
<td>13.8</td>
</tr>
</tbody>
</table>

and the training set from the target domain as the unlabeled adaptation data, reporting performance on the test set from the target domain. (The exception is the class proportion adjustment experiments, for which we used the test from the target domain as the unlabeled adaptation data.)

5.6.1 Feature Normalization

Corpus-level feature normalization is perhaps the simplest adaptation approach. We tested $z$—normalization of the continuous features by subtracting the mean and dividing by the standard deviation in the corresponding corpus’ training set. This might help in the case that the train/test mismatch were mainly due to some constant shifting of the features, for instance, if the lower speaking rate in BU-RNC shifted duration features, making them all higher than those in SWBD. Table 5.2 presents results for the BoosTexter classifier, using both acoustic and textual features; we considered normalization of all numeric features (including the “ratio” features and n-gram log likelihoods), or only of the acoustic numeric features. This kind of corpus-specific normalization appears to worsen cross-corpus training performance in most cases, although there is a slight improvement using acoustic feature normalization for BU-RNC on SWBD breaks.
It is not surprising that corpus-level z-normalization worsens performance in some cases—we would expect that application of different affine transformations to the train and test set would generally worsen performance of a classifier except in very specific situations where the transformations were known to be appropriate. We note that various kinds of normalization are already “built in” as part of the feature design process. Some features are normalized within speakers or stories, and some duration features are normalized relative to expected phone durations, where the expected values are genre specific (for broadcast news or conversational speech.) These kinds of normalization are likely more useful since they use knowledge about the task and about sources of genre and speaker variation in individual features.

5.6.2 Class Proportion Adjustment

We expected some class frequency differences between read and spontaneous speech. For instance, Yuan et al. (2005) showed that read speech has a higher proportion of words with pitch accents. As shown in Table 5.1 our BU-RNC data does overall have a higher proportion of accented words than our SWBD data, although there is also a fair amount of variation within different sections of each corpus. We did not observe an overall difference in proportion of breaks between corpora—the differences happened to be larger between train and test sets from the same corpus.

We tested whether changing the class prior $p(y)$ in the cross-corpus model could improve performance. If the class distributions $p(x|y)$ are the same across domains but only the class prior $p(y)$ differs, we can find $p_t(y|x)$ from $p_s(y|x)$ (for a particular $x$), as long as we know both $p_s(y)$ and $p_t(y)$. For binary classes $\{0, 1\}$:

$$
\begin{align*}
p_t(y = 1|x) &= \frac{p_s(y = 1|x)}{Z(x)} \cdot \frac{p_t(y = 1)}{p_s(y = 1)} \\
p_t(y = 0|x) &= \frac{p_s(y = 0|x)}{Z(x)} \cdot \frac{p_t(y = 0)}{p_s(y = 0)},
\end{align*}
$$

(5.1)

where $Z(x)$ is a normalization factor to make the two probabilities sum to 1. Even if one does not know $p_s(y)$ and $p_t(y)$, it might be possible to apply an iterative adjustment approach as described in Saerens et al. (2002), and in fact this is suggested for automatic adaptation for boundary detection in Shriberg et al. (2009).
Table 5.3: Results of class proportion adjustment (classification error rate %) for the logistic regression classifier using acoustic+textual features. “Cheating” uses labels in the test set to compute the class proportions; “unsupervised” uses the EM method described in Saerens et al. (2002).

<table>
<thead>
<tr>
<th></th>
<th>Accents</th>
<th></th>
<th>Breaks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>train SWBD</td>
<td>train BU-RNC</td>
<td>train SWBD</td>
</tr>
<tr>
<td></td>
<td>test BU-RNC</td>
<td>test SWBD</td>
<td>test BU-RNC</td>
</tr>
<tr>
<td>baseline</td>
<td>14.4</td>
<td>24.4</td>
<td>8.3</td>
</tr>
<tr>
<td>cheating (known</td>
<td>13.5</td>
<td>24.2</td>
<td>8.3</td>
</tr>
<tr>
<td>proportions)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>unsupervised</td>
<td>13.7</td>
<td>25.2</td>
<td>8.2</td>
</tr>
</tbody>
</table>

For these experiments we used logistic regression classifiers, since they directly output probabilities $p(y|x)$. In Table 5.3, we show the results of “cheating” experiments using known $p_t(y)$, as well as the results of unsupervised iterative experiments, following the procedure described in Saerens et al. (2002). In the cheating experiments, we applied Eqn. 5.1 to each test example $x$, using $p_s(y = 1)$ defined as the positive class proportion in the training set, and $p_t(y = 1)$ defined as the positive class proportion in the test set. In the unsupervised adjustment experiments, we started with $p_s(y = 1)$ in the training set, then estimated $p_{new}(y = 1)$ from the posterior probability estimates on the test set. We then computed $p_{new}(y = 1|x)$ for each example according to Eqn. 5.1 and repeated the procedure until the value of $p_{new}(y = 1)$ did not change. As shown in Saerens et al. (2002), this represents an instance of the EM algorithm. It finds values of $[p(y = 0), p(y = 1)]$ to locally maximize the likelihood function $\sum_{i \in D_t} \log p(x_i)$ where $p(x_i|y)$ is assumed fixed.

For the accent task, we observed an improvement in accuracy for the SWBD-trained classifier on BU-RNC. This improvement held for both the cheating and unsupervised approaches. However, the unsupervised approach degraded performance in the opposite direction. (It initially detected the correct proportion, and then adjusted the prior to be too
low.) Note that differences between the automatically-adjusted and unadjusted classifiers are not quite statistically significant for either direction, according to McNemar’s test with a threshold of \( p = 0.05 \). For the break task, where the prior differences are smaller, the prior adjustment approaches did not generally help.

Note that in Margolis et al. (2010a), we used an unsupervised adjustment procedure that differed slightly, using the classification proportions to estimate \( p(y) \) on each round (hard assignment). (We also used a different training parameter for logistic regression, and differences in feature computations noted above.) In that work we noted that in one case, the unsupervised procedure caused the prior to be adjusted down to 0%. That did not occur here, but as noted above we did observe cases where the prior was adjusted to be much too small.

### 5.6.3 Instance Weighting

We performed experiments with instance weighting to match the test set distribution, as described in, e.g., Shimodaira (2000); see Chapter 2 for a review of this technique. Given a source domain density \( p_s(x) \) and target domain density \( p_t(x) \), this method uses weighted training, where the cost associated with example \( \{x_i\} \) is weighted by \( w(x_i) = \frac{p_t(x_i)}{p_s(x_i)} \). In typical machine learning settings such as ours, where the densities are unknown, the first step involves estimation of the weights \( w(x_i) \) using a set of source domain examples \( D_s = \{x_i\}_{i=1}^{n_s} \) and a set of target domain examples \( D_t = \{x_i\}_{i=1}^{n_t} \). Estimating the densities from samples is difficult in our setting, since the feature vector \( x \) is relatively high-dimensional with both continuous and categorical features. Therefore, we investigated two published methods for estimating the instance weights without requiring the densities: (a) the discriminative method of Bickel et al. (2007), and (b) the Kullback-Leibler Importance Estimation Procedure (KLIEP) of Sugiyama et al. (2008).

In the discriminative method, a logistic regression model is learned to predict a binary variable \( \sigma \) that specifies whether a given sample was generated by the source domain \( S \) or target domain \( T \). The application of this model to a training example \( x_i \) is used to derive
the weight, according to:

\[ w(x_i) = \frac{p(\sigma = T|x_i)p(\sigma = S)}{p(\sigma = S|x_i)p(\sigma = T)}. \]

We used the same implementation as in Chapter 4, estimating the logistic regression model for \( p(\sigma = T|x) \) using equal fractions of source and target data, so that the weights on an example are just the odds of it belonging to the target domain. Also as in Chapter 4, we used a round-robin approach, so that the weight for a training example \( w(x_i) \) was derived from a model trained on a portion of the source domain excluding that example. We used \( C = 100 \) for learning the logistic regression models, and a ceiling of 10000 for the weights. Finally, we used the adaptive parameter \( \lambda \in (0, 1] \) to trade off the costs of bias and variance, weighting examples according to \( w(x_i)^\lambda \).

To learn \( w(x_i) \), we used the training sets from each domain as \( D_s \) and \( D_t \), including the full SWBD training set (which was slightly larger than the training set used in the baseline experiments above). We tuned \( \lambda \) to maximize weighted accuracy on the development set from the source domain, as suggested in Sugiyama et al. (2007). In particular, for weighted training with weights \( w(x)^\lambda \), we evaluated total accuracy on the development set with examples weighted by \( w(x) \). The optimal values of \( \lambda \) according to this evaluation were: \( \lambda = 0.3 \) for SWBD on BU for accents; \( \lambda = 0.8 \) for SWBD on BU for breaks; and \( \lambda = 0 \) (no weighting) for BU on SWBD, both tasks.

In KLIEP, the weight function \( w(x) \) is taken to be a linear combination of Gaussian kernels centered around target domain samples. Learning the weight function consists of learning a particular linear combination of these kernels in order to maximize an objective, which is based on the goal of minimizing Kullback-Leibler divergence between \( p_t(x) \) and \( w(x)p_s(x) \). The objective actually maximizes \( \sum_{x_i \in D_t} \log w(x_i) \) subject to \( \frac{1}{n_s} \sum_{x_i \in D_s} w(x_i) = 1 \). (See Chapter 2). We used the code available from the author.

The implementation uses cross-validation on the target data to set the Gaussian kernel width parameter, based on the sum of \( \log w(x_i) \) on the held-out target data. We trained the KLIEP model using the training sets from each domain as \( D_s \) and \( D_t \). We used 1000 Gaussian kernels.

All experiments with instance weighting were conducted using the logistic regression

\[ \text{http://sugiyama-www.cs.titech.ac.jp/~sugi/software/KLIEP/index.html} \]
Table 5.4: Results of instance weighting (classification error rate %) for the logistic regression classifier using acoustic+textual features. “Discriminative” uses the method based on Bickel et al. (2007), as described above; “KLIEP” uses the method from Sugiyama et al. (2008).

<table>
<thead>
<tr>
<th></th>
<th>Accents</th>
<th>Breaks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>train SWBD</td>
<td>train BU-RNC</td>
</tr>
<tr>
<td></td>
<td>test BU-RNC</td>
<td>test SWBD</td>
</tr>
<tr>
<td>baseline</td>
<td>14.4</td>
<td>24.4</td>
</tr>
<tr>
<td>discriminative</td>
<td>14.5</td>
<td>24.4</td>
</tr>
<tr>
<td>KLIEP</td>
<td>14.4</td>
<td>24.4</td>
</tr>
</tbody>
</table>

classifier from the LIBLINEAR package, as above. We used the version of LIBLINEAR that accepts weights on training instances.

Table 5.4 shows the results for both methods, which showed no effect in general, except for degradation in one case (the discriminative method on breaks, which had a higher value of $\lambda$). Note that the instance weighting approach assumes that the source domain distribution overlaps the target domain distribution (in particular, there must be examples in the source data that represent the important regions of the target domain). However, our belief is that the domains do not completely overlap. Table 5.5 shows performance of a classifier trained to discriminate BU-RNC from SWBD words, and evaluated on the test sets from both domains. These results show that the logistic regression classifier does a fairly good job at separating the domains, particularly when all features are available (which is the case in the above instance weighting experiments). The error rate of the majority class classifier is 46%.

To examine whether this “domain separability” contributes to error rate, we show in Table 5.6 the error rates of classifiers for the accent and break tasks on test examples that are correctly vs. incorrectly discriminated in row 1 of Table 5.5. In other words, we divided the test data into “discriminated” and “undiscriminated” based on whether the domain discriminator (using all features) correctly identified their domain. The idea is
Table 5.5: Domain discrimination performance (error rate) using different feature sets. Parentheses indicate which textual feature set was used (textual features differed between the accent and break tasks).

<table>
<thead>
<tr>
<th>feature set</th>
<th>error rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>acoustic+textual (accents)</td>
<td>24.5</td>
</tr>
<tr>
<td>acoustic+textual (breaks)</td>
<td>24.0</td>
</tr>
<tr>
<td>acoustic</td>
<td>29.1</td>
</tr>
<tr>
<td>textual (accents)</td>
<td>29.5</td>
</tr>
<tr>
<td>textual (breaks)</td>
<td>29.1</td>
</tr>
</tbody>
</table>

that discriminated examples can be considered to be less similar to the source domain than undiscriminated examples, and we would like to see whether greater cross-domain degradation occurs on those examples.

We show results for the in-corpus-trained classifier as well as the cross-corpus-trained classifier. We first note that the in-corpus-trained classifier does substantially worse on discriminated examples in two cases. This surprising result could be due to “outlier” clusters that are unique to one domain but not well modeled by the break/accent classifier trained on that domain. (Note also that even examples that look nothing like the training data can still receive high scores in discriminative classifiers; one might argue that a better approach to this analysis might be to use a generative classifier.) However, in all cases there is a larger increase in error from in-corpus to cross-corpus training on the discriminated examples than the undiscriminated ones. This suggests that a larger portion of cross-domain degradation error comes from examples that look more like the target domain than the source domain. Instance weighting could help with this, but only if such examples have similar representatives in the source domain. Furthermore, as discussed in Chapter 2, instance weighting can only help if the chosen model family cannot fit the decision boundaries of both domains simultaneously. Experiments in Section 5.5 showed at most small degradation from the domain-general classifiers compared with the in-domain-only
Table 5.6: In-corpus and cross-corpus error rates of logistic regression classifiers on subsets of the test data: domain-discriminated and undiscriminated examples.

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>Discriminated Error</th>
<th>Undiscriminated Error</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accents</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>on BU-RNC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In-corpus</td>
<td></td>
<td>12.1</td>
<td>12.3</td>
</tr>
<tr>
<td>Cross-corpus</td>
<td></td>
<td>14.5</td>
<td>14.2</td>
</tr>
<tr>
<td></td>
<td>on SWBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In-corpus</td>
<td></td>
<td>24.0</td>
<td>17.7</td>
</tr>
<tr>
<td>Cross-corpus</td>
<td></td>
<td>26.1</td>
<td>18.2</td>
</tr>
<tr>
<td><strong>Breaks</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>on BU-RNC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In-corpus</td>
<td></td>
<td>6.3</td>
<td>3.6</td>
</tr>
<tr>
<td>Cross-corpus</td>
<td></td>
<td>9.3</td>
<td>5.7</td>
</tr>
<tr>
<td></td>
<td>on SWBD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In-corpus</td>
<td></td>
<td>11.1</td>
<td>11.8</td>
</tr>
<tr>
<td>Cross-corpus</td>
<td></td>
<td>14.6</td>
<td>12.9</td>
</tr>
</tbody>
</table>
classifiers, suggesting that, by contrast, both model families do an adequate job of modeling the decision boundaries simultaneously.

5.6.4 Co-training and Self-training

We investigated co-training using the textual and acoustic feature sets as two views. As noted above, co-training was previously used for semi-supervised learning of prosodic events in Jeon and Liu (2009b) using BU-RNC, and the same approach was recently used to adapt BU-RNC to a different broadcast news corpus in Jeon et al. (2011). However, Jeon et al. (2011) did not report prosodic event classification performance on the news corpus since the goal was improved ASR performance. We investigated co-training for adaptation between BU-RNC and SWBD, which represent different styles of speech. We used the implementation of co-training from the original paper (Blum and Mitchell 1998), with only one labeled training set \( L \) used for both classifiers, letting each classifier select its most confident examples on each round. Logistic regression was used for both classifiers and for the combined-view classifier using all features. Note that our approach contrasts with some of the implementation details in Jeon and Liu (2009b); Jeon et al. (2011). For instance, they require label agreement of both classifiers for selected examples.

We set the size of the unlabeled pool (called \( u \) in Blum and Mitchell (1998)) to 200. On each iteration, each classifier selects the top \( p \) positively labeled examples and top \( n \) negatively labeled examples, and adds these to the training set using the assigned labels. For accents, we used \( n = p = 1 \), since the ratio of accents to non-accents is roughly 1. For breaks, we used \( p = 1 \) and \( n = 4 \). (We experimented with some of these settings by co-training within the BU-RNC set.)

In Figure [5.3] we plot performance on the target domain test set vs. the number of iterations. These plots represent performance of the classifiers trained on the labeled training set \( L \) at that round. In three cases, co-training was ineffective as a domain adaptation method, leading to substantial degradation in the case of BU-RNC on SWBD accents. In the case of BU-RNC on SWBD breaks, by contrast, co-training was effective at improving error rate: the classifier with all features went from 14.2% error rate to 12.7% after 600 iterations. F
As a comparison with co-training, we implemented self-training with hard assignment of labels. We trained the initial classifier using all acoustic and textual features on the source domain training set, then repeatedly selected the top $p$ positively labeled examples and $n$ negatively labeled examples from the adaptation set and added them to the training set. We used the same values of $p$ and $n$ as for co-training.

Performance of self-training on the target domain test set is also shown in Figure 5.3. Self-training was generally ineffective, and it led to degradation in one case (SWBD on BU...
breaks, which is also the case where the lexical view of co-training degraded).

We attempted to compare our implementation of co-training with the one in Jeon and Liu (2009b) for semi-supervised learning within the BU-RNC corpus. They did syllable-level accent detection, so our accent results are not comparable. On the break task, we found that our baseline results and fully-supervised results were generally better, perhaps due to different feature sets. Out of three runs with different initial labeled sets, we found that co-training was effective at improving our baseline results in two of them, but on the third the initial baseline result was very high and co-training led to a degradation. We conclude from this that our method can be effective at semi-supervised learning, but it depends on the baseline performance, and it is possible that their method is slightly better.

As noted above, self-training was used in Fernandez and Ramabhadran (2010) to adapt prosodic event detectors to the BU-RNC corpus from an out-of-domain professional speaker. They found it gave a small improvement on pitch accent and break detection using smaller amounts of labeled data; with larger amounts, it degraded performance. We did not explore the sensitivity of our methods to the amount of source data, but clearly the performance of some methods (in particular self-training and co-training) will be sensitive to this, so that should be considered in future work.

5.7 Feature Analysis

We examined the individual features in order to better understand the nature of the domain differences. We ranked all features based on the training accuracy for single-feature, single-threshold classifiers. Table 5.7 shows the top 10 features for each corpus and task. We observe that the top feature (word duration for accents, pause duration for breaks) is the same in both corpora, but the distribution of this feature varies between domains. For word duration, the non-accent class has roughly the same distribution, as shown in Figure 5.4, but the accent class distributions are shifted between domains—the accent class is more separated from the non-accent class in BU-RNC than in SWBD. In addition, for accents we observed that certain F0 difference features have more exaggerated class differences in BU-RNC than SWBD. This is illustrated in Figure 5.4 for the “F0K_WRD_DIFF_HIHI_N” feature, which represents the log ratio of the maximum F0 values in the current vs. next
word, and is in the top 10 for BU-RNC accents but not for SWBD. These observed differences between BU-RNC and SWBD for accents might be related to a “cleaner” prosodic marking style in the broadcast news domain compared with conversational speech. For breaks, we observed that BU-RNC lacks the long pauses present in conversational speech (such as during the other speaker’s turn), except for 500ms pauses that we inserted at boundaries of wavefiles (see above). We also observed that some differences depended on normalization, as shown for the “LAST_RHYME_NORM_DUR_PH_bin” and “LAST_RHYME_DUR_PH_bin” features in Figure 5.4. The first version of this feature uses normalized phone durations, while the second does not; the normalized version appears to show a more exaggerated difference between the break/no-break classes for BU-RNC and SWBD. We do not know the reason for this.

For accents, the n-gram probability features are more useful for BU-RNC, apparently because this corpus contains a larger number of accented, low frequency and out-of-vocabulary words, which have low log-probability values.

Some differences occur in the break task because many breaks in SWBD occur after backchannels and filled pause words (“uh”, ’uhhuh’, “oh”, “yeah”, etc.) In the SWBD training set, 20% of breaks are words marked with the backchannel symbol (the rest are break level 4 or 4-). This explains the importance of the interjection POS tag for SWBD. It also might explain why the “break ratio” feature is more highly ranked in SWBD: these words are both common and have a high proportion of breaks (recall that break ratio of a word is set to 0.5 if the empirical proportion of breaks is not significantly different from 0.5). BU-RNC does not have as many high-break-ratio words. The importance of backchannel breaks in SWBD also helps explain the improvement due to co-training in the BU-RNC-to-SWBD direction. Break ratio values for backchannel words cannot be learned from co-training (we use only the initial training set to compute these) but backchannels may be learned through other features, such as POS tags. Analysis shows that co-training increases detection of backchannels substantially (and also increases detection of break level 4 words).

In summary, our analysis illustrates that different factors contribute to cross-corpus degradation, including: “cleaner” prosodic marking for professionally read news compared
with conversational speech, resulting in exaggerated class differences for some acoustic features in the accent task; greater numbers of accented, low-frequency words in BU-RNC; different pause duration distributions, due to differences between read news and conversational speech as well as to differences in processing; and the presence of backchannels, which have their own vocabulary, break ratios, and POS tags. Cross-corpus differences vary by feature, and the impact of these differences varies by task (accents vs. breaks) and direction of adaptation. This suggests it may be difficult to find an out-of-the-box adaptation method that works with both tasks and directions. However, knowledge-based approaches that are informed about the task and nature of the domain differences may be more successful. As noted above, many of the acoustic features already contain normalization to account for known sources of variation, and this may contribute to the good “baseline” performance that we observe in this chapter.

5.8 Conclusions

This chapter explored the use of annotated training data from another corpus and speaking style for classifying prosodic events. When using a combination of acoustic and textual features, we found that models trained on data from the mismatched corpus increased classification error rates by no more than 5 percentage points compared with models trained on data from the same corpus. This held for both corpora, both tasks, and both models that we explored. Perhaps in part because the baseline cross-corpus performance was so good, it was difficult to improve this performance using unsupervised domain adaptation methods. We found that corpus-level acoustic feature z-normalization gave a small improvement on the break task in one direction (BU-RNC on SWBD), but degradation in all other cases. Class proportion adjustment, both with known proportions or automatically, gave a small improvement on the accent task in one direction (SWBD on BU-RNC); the automatic method caused a degradation in the opposite direction, and otherwise had little effect. Instance weighting had essentially no effect, except the discriminative method degraded performance in one case. Finally, co-training and self-training improved performance on the break task in one direction (BU-RNC on SWBD), but had no effect or led to degradation in the other cases. Our analysis of the domains suggests that a mix of factors contribute to
Table 5.7: Top ranked features (in decreasing order) for training accuracy on each training set, using single-feature, single-threshold classifiers.

<table>
<thead>
<tr>
<th></th>
<th>BU-RNC</th>
<th>SWBD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accents</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BU-RNC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WORD_DUR</td>
<td></td>
<td>WORD_DUR</td>
</tr>
<tr>
<td>LOG_UNIGRAM_PROB</td>
<td></td>
<td>MAX_PHONE_DUR_NSP</td>
</tr>
<tr>
<td>LOG_BIGRAM_PROB</td>
<td></td>
<td>NORM_WORD_DUR</td>
</tr>
<tr>
<td>LOG_BACKWARDS_BIGRAM_PROB</td>
<td></td>
<td>LOG_UNIGRAM_PROB</td>
</tr>
<tr>
<td>MAX_PHONE_DUR_NSP</td>
<td></td>
<td>MAX_VOWEL_DUR_NSP</td>
</tr>
<tr>
<td>F0K_WRD_DIFF_HIHI_N</td>
<td></td>
<td>AVG_PHONE_DUR_NSP</td>
</tr>
<tr>
<td>ACCENT_RATIO</td>
<td></td>
<td>ENERGY_WRD_DIFF_HIHI_N</td>
</tr>
<tr>
<td>POS (prep./conj.)</td>
<td></td>
<td>LAST_RHYME_DUR_PH_bin</td>
</tr>
<tr>
<td>F0K_MAXK_MODE_N</td>
<td></td>
<td>LAST_RHYME_NORM_DUR_PH_bin</td>
</tr>
<tr>
<td>ENERGY_WRD_DIFF_HIHI_N</td>
<td></td>
<td>LAST_RHYME_DUR_PH_ND_bin</td>
</tr>
<tr>
<td><strong>Breaks</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BU-RNC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAU_DUR</td>
<td></td>
<td>PAU_DUR</td>
</tr>
<tr>
<td>LAST_RHYME_NORM_DUR_PH_bin</td>
<td></td>
<td>POS_BIGRAM_BREAK_RATIO</td>
</tr>
<tr>
<td>F0K_WRD_DIFF_ENDBEG</td>
<td></td>
<td>BREAK_RATIO</td>
</tr>
<tr>
<td>LAST_RHYME_NORM_DUR_PH_ND_bin</td>
<td></td>
<td>next-POS (punc.)</td>
</tr>
<tr>
<td>F0K_WIN_DIFF_HILO_N</td>
<td></td>
<td>POS (interject.)</td>
</tr>
<tr>
<td>next-POS (punc.)</td>
<td></td>
<td>LAST_RHYME_DUR_PH_bin</td>
</tr>
<tr>
<td>WORD_DUR</td>
<td></td>
<td>LAST_RHYME_DUR_PH_ND_bin</td>
</tr>
<tr>
<td>POS_BIGRAM_BREAK_RATIO</td>
<td></td>
<td>WORD_DUR</td>
</tr>
<tr>
<td>LAST_RHYME_DUR_PH_ND_bin</td>
<td></td>
<td>PREV_PAU_DUR</td>
</tr>
<tr>
<td>LAST_RHYME_DUR_PH_bin</td>
<td></td>
<td>F0K_WRD_DIFF_ENDBEG</td>
</tr>
</tbody>
</table>
Figure 5.4: Some empirical feature distributions in the training sets for each domain.
cross-corpus-training degradation, and that different features are affected in different ways. Therefore, cross-domain improvements might be better sought through “knowledge-based” approaches to feature design that normalize for known sources of variation, although the already-good baseline cross-domain performance may indicate that the existing features are already well designed.
Chapter 6

DOMAIN ADAPTATION WITH UNLABELED DATA FOR DIALOG ACT TAGGING

6.1 Introduction

Dialog act (or speech act) tagging aims to label abstract functions of utterances in conversations, such as “request,” “floorgrab,” or “statement.” Potential applications include automatic conversation analysis, punctuation transcription, and human-computer dialog systems. Although some applications require domain-specific tag sets, it is often useful to label utterances based on generic tags, and several tag sets have been developed for this purpose, e.g., Dialog Act Markup in Several Layers (DAMSL) \cite{Core97}. Many approaches to automatic dialog act (DA) tagging assume hand-labeled training data. However, when building a new system it may be difficult to find a labeled corpus that matches the target domain, or even the language. Even within the same language, speech from different domains can differ linguistically, and the same DA categories might be characterized by different cues. The domain characteristics (face-to-face vs. telephone, two-party vs. multi-party, informal vs. agenda-driven, familiar vs. stranger) can influence both the distribution of tags and word choice.

In this chapter, we consider cross-domain training for dialog act classification\footnote{This chapter is based on Margolis et al. \cite{Margolis10b}} conducting experiments among three conversational speech corpora: the Meeting Recorder Dialog Act corpus (MRDA) \cite{Shriberg04}, the Switchboard DAMSL corpus (Swbd) \cite{Jurafsky97}, and the Spanish Callhome dialog act corpus (SpCH) \cite{Levin98}. The first is multi-party, face-to-face meeting speech; the second is topic-prompted telephone speech between strangers; and the third is informal telephone speech between friends and family members. The first two are in English, while the third is in Spanish. When the source and target domains differ in language, we apply machine translation to the
target domain to convert it to the language of the source domain. In all cases, we classify pre-segmented utterances based on their transcripts, and we consider only four high-level classes: statement, question, backchannel, and incomplete. We compare two feature-based domain adaptation approaches that were investigated in Chapter 3: the feature restriction approach, which forces the classifier to learn only on “shared” features that appear in both domains, and Structural Correspondence Learning (SCL) from Blitzer et al. (2007), which learns new features from many auxiliary tasks. The feature restriction approach has been investigated for adaptation in other tasks, e.g. sentiment classification (Aue and Gamon 2005) and parsing (Dredze et al. 2007). SCL has been used successfully for a number of NLP tasks, including sentiment classification (Blitzer et al. 2007), part-of-speech tagging (Blitzer et al. 2006), entity recognition (Ciaramita and Chapelle 2010), and conversation summarization (Sandu et al. 2010); here we investigate its applicability to the DA classification task, using a multi-view implementation as suggested by Blitzer et al. (2006) and Blitzer et al. (2009). In addition to analyzing these two methods on a novel task, we show an interesting comparison between them: in this setting, both methods turn out to have a similar effect, caused by the correlation between backchannel words and utterance length.

6.2 Related Work

There exists a substantial amount of work on dialog or speech act tagging using various knowledge sources, modeling approaches, and tag sets. Approaches include generative utterance (language) models combined with prosodic feature decision trees (Shriberg et al. 1998) and DA sequence models (Stolcke et al. 2000; Venkataraman et al. 2003); dynamic Bayesian networks (Dielmann and Renals 2008; Ji and Bilmes 2005); boosted decision trees (Boakye et al. 2009; Guz et al. 2010b; Jeon and Liu 2009b); maximum entropy models (Moniz et al. 2011; Ang et al. 2005; Jeon and Liu 2009b); SVMs (Surendran and Levow 2006; Liu 2006); decision trees (Yuan and Jurafsky 2005; Fernandez and Picard 2002; Hastie et al. 2002); HMM models (Wright 1998); and a cue-phrase approach (Webb and Liu 2008). Dialog act tagging has been combined with dialog act segmentation in sequential (Ang et al. 2005; Mast et al. 1996; Guz et al. 2010b) and joint approaches (Warnke et al. 1997; Quarteroni et al. 2011).
Automatic DA tagging across domain has been investigated by a handful of researchers. \cite{WebbLiu:2008} investigated cross-corpus training between Swbd and another corpus consisting of task-oriented calls, although no adaptation was attempted. Similarly, Rosset et al. \cite{Rosset:2008} reported on recognition of detailed DA tags across domain and language (French to English) by using utterances that had been pre-processed to extract linguistic and task-specific entities (whose tag sets were the same across languages). \cite{Tur:2005} applied supervised model adaptation to intent classification across customer dialog systems, and \cite{Guz:2010b} applied supervised model adaptation methods for DA segmentation and classification on MRDA using labeled data from both MRDA and Swbd. Most similar to our work is that of \cite{Jeong:2009}, who compared two methods for adaptation, using Swbd/MRDA as the source training set and email or forums corpora as the target domains. Both methods were based on incorporating unlabeled target domain examples into training. Success has also been reported for self-training approaches on same-domain semi-supervised learning for dialog act or call classification \cite{Venkataraman:2003, Tur:2005, TurHakkaniTur:2003}. In our work, we are more interested in exploring feature representation approaches to this problem, which have not been explored for this task and have received less attention generally compared with self-training.

We are not aware of prior work on cross-lingual DA tagging via machine translation. However, as reviewed in Chapter 2, there has been significant work on domain adaptation for cross-language text classification problems, often involving machine translation and discovery of new feature representations. Recently \cite{PrettenhoferStein:2010} and \cite{WeiPal:2010} used SCL combined with machine translation for cross-lingual text classification. (Their work was presented at the same time as ours.)

### 6.3 Methods

Our four-class DA problem is similar to problems studied in other work, such as \cite{Tur:2007}, who used five classes (ours plus floorgrab/hold). When defining a mapping from each corpus’ tag set to the four high-level classes, our goal was to try to make the classes similarly defined across corpora. Note that the incomplete category is defined in Swbd-DAMSL to include only utterances too short to determine their DA label (e.g., just a filler
word). Thus, for our work the MRDA incomplete category excludes utterances also tagged as statement or question; it includes those consisting of just a floor-grab, hold or filler word.

For classification we used an SVM with linear kernel, with $L_2$ regularization and $L_1$ loss as implemented in the Liblinear package (Fan et al. 2008), which uses the one-vs.-rest configuration for multiclass classification. Features are derived from the hand transcripts, which are hand-segmented into DA units. Punctuation and capitalization are removed so that our setting corresponds to classification based on (perfect) speech recognition output. The features are counts of unigrams, bigrams, and trigrams that occur at least twice in the train set, including beginning/end-of-utterance tags ($\langle s \rangle$, $\langle /s \rangle$), and a length feature (total number of words, z-normalized across the training set).

As noted above, some previous work on DA tagging has used contextual features from surrounding utterances, or Markov models for the DA sequence. In addition, some work has used prosodic or other acoustic features. Stolcke et al. (2000) found benefits to using Markov sequence models and prosodic features in addition to word features, but those benefits were relatively small, so for simplicity our experiments here use only word features and classify utterances in isolation. Furthermore, Zimmermann et al. (2006) compared different models with lexical-only features for classifying a small set of dialog acts in MRDA, and showed that maximum entropy models and boosting outperformed generative language modeling. Our setup with linear SVMs is similar to the maximum entropy approach, except we use all unigrams, bigrams and trigrams rather than n-grams at the utterance edges, and we use counts rather than binary features. Overall our tagging approach is close to that of Surendran and Levow (2006).

We used Google Translate to derive English translations of the Spanish SpCH utterances, and to derive Spanish translations of the English Swbd and MRDA utterances. Of course, translations are far from perfect; DA classification performance could likely be improved by using a translation system trained on spoken dialog. For instance, Google Translate often failed on certain words like “i” that are usually capitalized in text. Even so, when training and testing on translated utterances, the results with the generic system are surprisingly good.

The results reported in this chapter used the standard train/test splits provided with
the corpora: MRDA had 51 train meetings/11 test; Swbd had 1115 train conversations/19 test; SpCH had 80 train conversations/20 test. The SpCH train set is the smallest at 29k utterances. To avoid issues of differing train set size when comparing performance of different models, we reduced the Swbd and MRDA train sets to the same size as SpCH using randomly selected examples from the full train sets. For each adaptation experiment, we used the target domain training set as the unlabeled data, and report performance on the target domain test set. The test sets contain 4525, 15180, and 3715 utterances for Swbd, MRDA, and SpCH respectively.

6.4 Results

Table 6.1 shows the class proportions in the training sets for each domain. MRDA has fewer backchannels than the others, which is expected since the meetings are face-to-face. SpCH has fewer incompletes and more questions than the others; the reasons for this are unclear. Backchannels have the shortest mean length (less than 2 words) in all domains. Incompletes are also short, while statements have the longest mean length. The mean lengths of statements and questions are similar in the English corpora, but are shorter in SpCH. (This may point to differences in how the utterances were segmented; for instance Swbd utterances can span multiple turns, although 90% are only one turn long.)

Table 6.1: Proportion of utterances in each DA category in each domain’s training set.

<table>
<thead>
<tr>
<th></th>
<th>incomplete</th>
<th>statement</th>
<th>question</th>
<th>backchannel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swbd</td>
<td>8.1%</td>
<td>67.1%</td>
<td>5.8%</td>
<td>19.1%</td>
</tr>
<tr>
<td>MRDA</td>
<td>10.7%</td>
<td>67.9%</td>
<td>7.5%</td>
<td>14.0%</td>
</tr>
<tr>
<td>SpCH</td>
<td>5.7%</td>
<td>60.6%</td>
<td>12.1%</td>
<td>21.7%</td>
</tr>
</tbody>
</table>

Because of the high class skew, we consider two different schemes for training the classifiers, and report different performance measures for each. To optimize overall accuracy, we use basic unweighted training. To optimize average per-class recall (weighted equally across all classes), we use weighted training, where each training example is weighted inversely
to its class proportion. We optimize the regularization parameter using a source domain development set corresponding to each training set. Since the optimum values are close for all three domains, we choose a single value for all the accuracy classifiers and a single value for all the per-class recall classifiers. (Different values are chosen for different feature types corresponding to the different adaptation methods.)

Table 6.2 gives baseline performance for all train-test pairs, using translated versions of the test set when the train set differs in language. It also lists the in-domain results using translated (train and test) data, and results using the adaptation methods (which we discuss below). Figure 6.1 shows details of the contribution of each class to the average per-class recall; bar height corresponds to the second column in Table 6.2.

6.4.1 Baseline performance and analysis

We observe first that translation does not have a large effect on in-domain performance; degradation occurs primarily in incompletes and questions, which depend most on word order and therefore might be most sensitive to ordering differences in the translations. We conclude that it is possible to perform well on the translated test sets when the training data is well matched. However, cross-domain performance degradation is much worse between pairs that differ in language than between the two English corpora.

We now describe three kinds of issues contributing to cross-domain degradation, which we observed anecdotally. First, some highly important words in one domain are sometimes missing entirely from another domain. This issue appears to have a dramatic effect on backchannel detection across languages: when optimizing for average per-class recall, the English-trained classifiers detect about 20% of the Spanish translated backchannels and the Spanish classifier detects a little over half of the English ones, while they each detect more than 80% in their own domain. The reason for the cross-domain drop is that many backchannel words in the English corpora (uhhuh, right, yeah) do not overlap with those in the Spanish corpora (mmm, sí, ya) even after translation—for example, “ya” becomes “already”, “sí” becomes “yes”, “right” becomes “derecho”, and “uhhuh”, “mmm” are un-
Table 6.2: Overall accuracy and average per-class recall on each test set, using in-domain, in-domain translated, and cross-domain training. Starred results under the accuracy column are significantly different from the corresponding cross-domain baseline under McNemar’s test ($p < 0.05$). (Significance is not calculated for the average per-class recall column.) “Majority” classifies everything as statement.

<table>
<thead>
<tr>
<th>train set</th>
<th>Acc (%)</th>
<th>Avg. Rec. (%)</th>
<th>train set</th>
<th>Acc (%)</th>
<th>Avg. Rec. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test on Swbd</strong></td>
<td></td>
<td></td>
<td><strong>Test on MRDA</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swbd</td>
<td>89.2</td>
<td>84.9</td>
<td>MRDA</td>
<td>83.8</td>
<td>80.5</td>
</tr>
<tr>
<td>Swbd translated</td>
<td>86.7</td>
<td>80.4</td>
<td>MRDA translated</td>
<td>80.5</td>
<td>74.7</td>
</tr>
<tr>
<td>MRDA baseline</td>
<td><strong>86.4</strong></td>
<td><strong>78.0</strong></td>
<td>Swbd baseline</td>
<td><strong>81.0</strong></td>
<td>71.6</td>
</tr>
<tr>
<td>MRDA feat. restriction</td>
<td>85.7*</td>
<td>77.7</td>
<td>Swbd feat. restriction</td>
<td>80.1*</td>
<td><strong>72.1</strong></td>
</tr>
<tr>
<td>MRDA SCL</td>
<td>81.8*</td>
<td>69.6</td>
<td>Swbd SCL</td>
<td>75.6*</td>
<td>68.1</td>
</tr>
<tr>
<td>MRDA length only</td>
<td>78.3*</td>
<td>51.4</td>
<td>Swbd length only</td>
<td>68.6*</td>
<td>44.9</td>
</tr>
<tr>
<td>SpCH baseline</td>
<td>74.5</td>
<td>57.2</td>
<td>SpCH baseline</td>
<td>66.9</td>
<td>50.5</td>
</tr>
<tr>
<td>SpCH feat. restriction</td>
<td>77.4*</td>
<td>64.2</td>
<td>SpCH feat. restriction</td>
<td>66.8</td>
<td>52.1</td>
</tr>
<tr>
<td>SpCH SCL</td>
<td>76.8*</td>
<td><strong>64.8</strong></td>
<td>SpCH SCL</td>
<td>66.1*</td>
<td><strong>58.4</strong></td>
</tr>
<tr>
<td>SpCH length only</td>
<td><strong>77.7</strong>*</td>
<td>48.2</td>
<td>SpCH length only</td>
<td><strong>68.3</strong>*</td>
<td>44.6</td>
</tr>
<tr>
<td>majority</td>
<td>67.7</td>
<td>25.0</td>
<td>majority</td>
<td>65.2</td>
<td>25.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>train set</th>
<th>Acc (%)</th>
<th>Avg. Rec. (%)</th>
<th>train set</th>
<th>Acc (%)</th>
<th>Avg. Rec. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test on SpCH</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SpCH</td>
<td>83.1</td>
<td>72.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SpCH translated</td>
<td>82.4</td>
<td>71.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swbd baseline</td>
<td>63.8</td>
<td>41.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swbd feat. restriction</td>
<td>66.2*</td>
<td><strong>50.9</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swbd SCL</td>
<td>68.2*</td>
<td>47.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swbd length only</td>
<td><strong>72.6</strong>*</td>
<td>43.6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MRDA baseline</td>
<td>65.1</td>
<td>42.9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MRDA feat. restriction</td>
<td>65.5</td>
<td><strong>51.2</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MRDA SCL</td>
<td>67.6*</td>
<td>50.9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MRDA length only</td>
<td><strong>72.6</strong>*</td>
<td>44.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>majority</td>
<td>65.3</td>
<td>25.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 6.1: Per-class recall of weighted classifiers in column 2 of Table 6.2. Bar height represents average per-class recall; colors indicate contribution of each class: I=incomplete, S=statement, Q=question, B=backchannel. (Maximum possible bar height is 100%, each color 25%).
A second issue has to do with different kinds of utterances found in each domain, which sometimes lead to different relationships between features and class label. This is sometimes caused by the translation system; for example, utterances starting with “es que . . .” are usually statements in SpCH, but without capitalization the translator often gives “is that . . .”. Since “⟨s⟩–is–that” is a cue feature for questions in English, these utterances are usually labeled as question by the English domain classifiers. The existence of different types of utterances can result in sets of features that are more highly correlated in one domain than the other. In both Swbd and translated SpCH, utterances containing the trigram “⟨s⟩–but–⟨/s⟩” are most likely to be in the incomplete class. In Swbd, the bigram “but–⟨/s⟩” rarely occurs outside of that trigram, but in SpCH it sometimes occurs at the end of long (syntactically-incomplete) statements, so it corresponds to much lower likelihood for the incomplete class.

The last issue concerns utterances whose true label probabilities given the word sequence are not the same across domains. We distinguish two such kinds of utterances. The first are due to class definition differences across domains and annotators, e.g., long statements or questions that are also incomplete are more often labeled incomplete in SpCH and Swbd than in MRDA. The second kind are utterances whose class labels are not completely determined by their word sequence. To minimize error rate the classifier should label an utterance with its most frequent class, but that may differ across domains. For example, “yes” can be either a statement or a backchannel; in the English corpora, it is most likely to be a statement (“yeah” is more commonly used for backchannels). However, “sí” is most likely to be a backchannel in SpCH. To measure the effect of differing label probabilities across domains, we trained “domain-general” classifiers using concatenated training sets for each pair of domains. We found that they performed about the same or only slightly worse than domain-specific models, so we conclude that this issue is likely only a minor effect.

\[^2\text{This issue might be handled better by using multiple alternative translations rather than just the top translation. We have not yet explored this sort of approach.}\]
6.4.2 Adaptation using feature restriction

In the cross-language domain pairs, some discriminative features in one domain are missing in the other. By removing all features from the source domain training utterances that are not observed (twice) in the target domain training data, we force the classifier to learn only on features that are present in both domains. As seen in Figure 6.1, this had the effect of improving recall of backchannels in the four cross-language cases. Backchannels are the second-most frequent class after statements, and are typically short in all domains. Many typical backchannel words are domain-specific; by removing them from the source data, we force the classifier to attempt to detect backchannels based on length alone. The resulting classifier has a better chance of recognizing target domain backchannels that lack the source-only backchannel words. At the same time, it mistakes many other short utterances for backchannels, and does particularly worse on incompletes, for which length is also a strong cue. Although average per-class recall improved in all four cross-language cases, total accuracy only improved significantly in two of those cases, and for the Swbd/MRDA pair, accuracy got significantly worse. The effect on the one-vs.-rest component classifiers was mixed: for some (statement and some backchannel classifiers in the cross-language cases), accuracy improved, while in other cases it decreased.

In Chapter 3, we investigated the feature restriction approach and showed how it can be justified by a “positive correspondence” assumption. We also illustrated cases where it can lead to degradation compared with using all source-domain features. In this multi-class problem, we believe that a mix of these cases occurs. As an example of positive correspondence, consider the discriminative source and target features “uhluh” and “mmm,” which are both similarly correlated with a shared, noisier, feature (length). Forcing the model to learn only on the shared, noisy feature incorporates correlation information about “uhluh”, which is similar to that of “mmm”. Thus, the reduced model is potentially more useful on the target domain, compared to the full source domain model which might not put weight on the noisy feature. This kind of scenario was illustrated in Figure 3.3. On the other hand, when the target domain utterances actually represent samples from a subspace of the source domain, the absence of features is informative: the fact that an utterance does not
contain “⟨s⟩–verdad–⟨/s⟩”, for instance, might mean that it is less likely to be a question, even if none of the target domain utterances contain this feature. This kind of scenario was illustrated in Figure 3.1.

6.4.3 Adaptation using SCL

When training auxiliary tasks, Blitzer (2007) avoided using any feature to predict itself. However, with unigram and bigram features, there are generally multiple features in each utterance that are highly correlated with a given pivots, due to overlapping n-grams (i-love vs. love). This creates a challenge for selecting features to use in the predictions. Therefore, we adopted the multi-view learning approach suggested by Blitzer et al. (2006) and Blitzer et al. (2009), where the feature set is divided into “views,” and features in one view are used to predict features in another view. We defined two views by splitting the utterances into two non-overlapping parts; pivot features in the first part were predicted with all the features in the second, and vice versa. This avoids “trivial” prediction tasks based on overlapping n-grams. We experimented with splitting the utterances in the middle, but found that since the number of words in the first part (nearly) predicts the number in the second part, all of the features in the first part were positively predictive of pivots in the second part so the main dimension learned was length. In the results presented here, the first part consists of the first word only, and the second part is the rest of the utterance. (All utterances in our experiments have at least one word.) Pivot features are selected in each part and predicted using a least-squares linear regression on all features in the other part.

We used the SCL-MI method of Blitzer et al. (2007) to select pivot features, which requires that they be common in both domains and have high mutual information (MI) with the class (according to the source labels). We selected features that occurred at least 10 times in each domain and were in the top 500 ranked MI features for any of the four classes; this resulted in 78-99 first-part pivots and 787-910 second-part pivots (depending on the source-target pair). Unlike Blitzer’s work, which set the negative weights to 0, we keep all weight entries. We performed SVD on the learned weight matrix for each part separately, and the top (at most) 100 dimensions were used to project utterances on each
In all train-test pairs, the first dimension of the first part appeared to distinguish short utterance words from long ones. Such short-utterance words included backchannels from both domains, in addition to acknowledgments, exclamations, swear words and greetings. An analogous dimension existed in the second part, which captured words correlated with short utterances longer than one word (right, really, interesting). The other dimensions of both domains were difficult to interpret.

We experimented with using the SCL features together with the raw features (n-grams and length), as suggested by Blitzer et al. (2006). As in Blitzer et al. (2006), we found it necessary to scale up the SCL features to increase their utilization in the presence of the raw features; however, it was difficult to guess the optimal scaling factor without having access to labeled target data. The results here use SCL features only, which also allows us to more clearly investigate the behavior of those features. Note that the versions of SCL implemented in other chapters appended the learned features onto the raw features, and used either “development tasks” (Chapter 3) or a development set in the target domain (Chapter 7) to set the scaling factor.

The most notable effect was an improvement in backchannel recall, which occurred under both weighted and unweighted training. In addition, there was high confusability between statements and the other classes, and more false detections of backchannels. When optimizing for accuracy, SCL led to an improvement in accuracy in three of the four cross-language cases. When optimizing for average per-class recall, it led to improvement in all cross-language cases; however, recall of statements went down dramatically in all cases. In addition, while there was no clear benefit of the SCL vs. the feature restriction method on the cross-language cases, the SCL approach did much worse on the Swbd/MRDA pair, causing large degradation from the baseline.

As we have noted, utterance length appears to underlie the improvement seen in the cross-language performance for both the SCL and feature-restriction approaches. Therefore, we include results for a classifier based only on the length feature. Optimizing for accuracy, this method achieves the highest accuracy of all methods in the cross-language pairs. (It does so by classifying everything as statement or backchannel, although with weighted training,
as shown in Figure 6.1, it gets some incompletes.) However, under weighted class training, the average per-class recall of this method is much worse than the feature restriction and SCL approaches.

Note that in Chapter 3, we argued that although SCL and feature restriction make similar assumptions, SCL is more powerful since it incorporates target-domain-only features. So it may seem surprising that SCL does not generally outperform feature restriction or the length-only approach in detection of backchannels. Note first that there is an important difference in our implementation of SCL from that analyzed in Chapter 3, because here we include only the SCL features, which means that only the n-gram features included in the prediction tasks get included in the SCL model. Thus, although SCL here includes many target-specific features, it excludes many useful features used in the feature restriction approach, such as the length feature. Furthermore, the SCL features are limited to describing latent variation found in the pivots, and there are many fewer pivots than shared features. This is a consequence partly of how we defined the pivots—excluding features that span the first and rest of the utterance—and partly a consequence of the requirement that pivots be common in both domains. The latter requirement helps to exclude latent dimensions that describe variation between domains, but it can also exclude latent dimensions found in the shared features that are useful for discriminating labels.

In principle, the SCL dimension related to length could give better cross-domain performance than the actual length feature. The value of this SCL feature does not describe the actual length of an utterance—rather, it captures something that might be described as “propensity of the first word in the utterance to occur in a short utterance.” If this feature more clearly segregated backchannel words, it might do better, but as mentioned above it also captures other kinds of short-utterance words, making it fairly noisy.

Comparison with other SCL tasks

Although we basically take a text classification approach to the problem of dialog act tagging, our problem differs in several ways from the sentiment classification task in Blitzer et al. (2007). In particular, utterances are much shorter than documents, and we use posi-
tion information via the start/end-of-sentence tags. Some important DA cue features (such as the value of the first word) are mutually exclusive rather than correlated. In this way our problem resembles the part-of-speech tagging task in Blitzer et al. (2006), where the category of each word is predicted using values of the left, right, and current word token. That work used a kind of multi-view learning for the SCL projection, with three views corresponding to the three word categories. However, our problem essentially uses a mix of bag-of-words and position-based features, since some n-gram features include position information (via start/end-utterance tags), but most do not. This poses a greater challenge since there is no natural multi-view split: the bag-of-words features can come from any part of the utterance, while at the same time overlapping the position-based features. The approach described here suffers from the fact that it cannot use all of the features available to the baseline classifier—bigrams and trigrams spanning the first and second words are left out. It also suffers from the fact that the first-word pivot feature set is extremely small—a consequence of the small set of first words that occur at least 10 times in the 29k-utterance corpora. Better performance might be achieved on the non-backchannel classes with the raw features included, as proposed in the original version of SCL. However, the challenge with this is how to set the scaling value appropriately, as discussed above.

### 6.5 Conclusions

We have considered two approaches for domain adaptation for DA tagging, and analyzed their performance for source/target pairs drawn from three different domains. For the English domains, the baseline cross-domain performance was quite good, and both adaptation methods generally led to degradation over the baseline. For the cross-language cases, both methods were effective at improving average per-class recall, and particularly backchannel recall. SCL led to significant accuracy improvement in three cases, while the feature restriction approach did so in two cases. On the other hand, SCL showed poor discrimination between statements and other classes, and did worse on the same-language pair that had little cross-domain degradation. Both methods work by taking advantage of correlations between shared and domain-specific class-discriminative features. Unfortunately in our task, membership in the rare classes is often cued by features that are mutually exclusive, e.g.,
the starting n-gram for questions. Both methods might therefore benefit from additional shared features that are correlated with these n-grams, e.g., sentence-final intonation for questions. Indeed, other work on semi-supervised DA tagging has used a richer feature set: Jeong et al. (2009) included parse, part-of-speech, and speaker sequence information, and Venkataraman et al. (2003) used prosodic information, plus a sequence-modeling framework. (In Chapter 7, we address the task of question detection specifically and apply domain adaptation using prosodic and n-gram features, but we adapt from text and speech, where the prosodic features are only found in one domain.) Finally, from the task perspective, an interesting result of this chapter is that machine translation appears to preserve most of the dialog-act information, in that in-domain performance is similar on original and translated text.
Chapter 7

AUTOMATIC ANNOTATION OF SPOKEN CONVERSATIONS USING TEXTUAL CONVERSATIONS

7.1 Introduction

In this chapter, we investigate two annotation tasks in spoken conversations: automatic question detection and automatic sentence boundary detection. Both prosodic features (extracted from the acoustic signal) and lexical features (extracted from the word sequence) have been shown to be useful for these tasks (Shriberg et al. 1998; Kim and Woodland 2003; Ang et al. 2005; Liu et al. 2006), but access to labeled speech training data is generally required in order to use prosodic features. On the other hand, the Internet contains large quantities of textual data that is already labeled with punctuation, and which can be used to train a system using lexical features. In particular, Internet-based textual conversations offer a possible analogue to spontaneous spoken conversations, and may be useful for building systems to automatically annotate dialog. We consider automatic annotation of spoken conversations in the Meeting Recorder Dialog Act corpus (MRDA) (Shriberg et al. 2004), using textual conversations from Wikipedia “talk” pages. We compare the performance of models trained on the textual domain using lexical features with those trained on MRDA using lexical features and/or prosodic features. Then, we investigate domain adaptation methods that utilize unlabeled MRDA data with lexical and prosodic features. The goal is to use the unlabeled domain-matched data to bridge stylistic differences as well as to incorporate the prosodic features, which are unavailable in the labeled text data.

7.2 Related Work

Our work in this chapter is related to work on automatic dialog act tagging, sentence boundary detection, and punctuation annotation. As discussed in Chapter 6, dialog act tagging...
or speech act tagging aims to classify functions of utterances in conversations, and may include high-level categories (statement, question, backchannel) or more detailed categories (such as question type). Sentence boundary detection, or sentence segmentation, occurs at word boundaries; for conversational speech, the task (also called dialog act segmentation) has been generalized to “sentence units” ([Linguistic Data Consortium 2006]), which need not be syntactically complete (e.g. backchannels and abandoned utterances). Sentence boundary detection is usually considered as a binary task, although performance on different dialog act boundary types is sometimes reported. Punctuation annotation is similar to sentence boundary detection, but involves insertion of different punctuation types (like period, question mark, comma) at word boundaries, so it is sometimes treated as a multiclass tagging problem (with “no boundary” as one of the categories).

In this work, we consider question detection as a binary classification task on pre-segmented utterances, using lexical and prosodic features extracted from the entire utterance. We consider sentence boundary detection as a binary classification task at word boundaries, using lexical features and prosodic features around the boundary. (The prosodic features are a similar set to those in Chapter 5.) The benefit of performing question detection on pre-segmented utterances (rather than at word boundaries, as in punctuation annotation) is that we can include features extracted from the entire utterance. In a fully automatic system, one could apply sentence boundary detection followed by question detection. However, for our work we use the manually annotated sentence unit boundaries, which allows us to analyze the performance on different question types, in isolation from segmentation issues.

Question detection on whole utterances can be considered a subtask of dialog act tagging. Previous work on dialog act tagging or question detection has investigated the utility of various feature types; [Boakye et al. (2009), Shriberg et al. (1998), and Stolcke et al. (2000)] showed specifically that utterance-level prosodic features were useful for discriminating questions from statements in English conversational speech, but (at least in the absence of recognition errors) most of the performance was achieved with words alone. (Related investigations have been performed in other languages, e.g. [Yuan and Jurafsky 2005]; Quang et al. (2007).) Shriberg et al. (1998) also verified that different prosodic cues are useful for
different question types, which had been predicted from previous work. For example, they showed that yes-no and declarative questions benefit from final F0 rise features, while wh questions do not, and they argued that these features are therefore less useful when different question types are grouped together into a single question class.

Cross-corpus training for automatic recognition of dialog acts has been conducted between different spoken domains (Webb and Liu 2008; Rosset et al. 2008). Hu et al. (2009) proposed a new dialog annotation scheme that would apply to both textual and spoken conversations, and used it to annotate telephone conversations and email, comparing relative frequencies of tags across corpora and automatic detection accuracies for in-domain taggers. Moniz et al. (2011) compared question types in different Portuguese corpora, including text and speech. For question detection on speech, they compared performance of a lexical model trained with newspaper text to models trained with speech including acoustic and prosodic features, where the speech-trained model also utilized the text-based model predictions as a feature. They reported that the lexical model mainly identified wh questions, while the speech data helped identify yes-no and tag questions, although results for specific categories were not included.

For automatic punctuation annotation and sentence segmentation of speech, a variety of modeling approaches have been employed. Language models including punctuation or sentence boundary symbols were built from annotated word sequences in many works (Stolcke and Shriberg 1996; Beeferman et al. 1998; Shriberg et al. 2000; Gotoh and Renals 2000; Christensen et al. 2001; Kim and Woodland 2003; Liu et al. 2006; Ang et al. 2005; Mast et al. 1996). As a different kind of lexical model, Shen et al. (2009) combined a cue-word-based classifier with punctuation probabilities from contextual words. Such lexical models can be used to identify punctuation or boundaries based only on a given word sequence (Stolcke and Shriberg 1996; Beeferman et al. 1998), or additionally combined with prosodic models (Shriberg et al. 2000; Ang et al. 2005; Kim and Woodland 2003; Liu et al. 2006; Gotoh and Renals 2000; Christensen et al. 2001; Kolář et al. 2006b; Mast et al. 1996; Shen et al. 2009), such as decision trees or multi-layer perceptrons (MLPs). Segmentation procedures without lexical information have been described as well (Wang and Narayanan 2004; Aylett 2006). Simultaneous speech recognition and punctuation annotation was described
by Kim and Woodland (2003) and Chen (1999), using word/punctuation language models, and assuming certain pronunciations (such as silence) for punctuation. Other approaches to punctuation or sentence boundary detection include discriminative approaches such as boosted decision trees (Cuendet et al. 2007b; Kolár et al. 2006b), maximum entropy models (Huang and Zweig 2002; Liu et al. 2006), conditional random fields (Liu et al. 2006), and MLPs (Christensen et al. 2001). These approaches may incorporate both lexical and prosodic features in the same model. However, one of the benefits of using separate lexical and prosodic models is that they can be trained on separate datasets; the lexical model can be trained on larger amounts of text that may be lacking audio, as is done in many works (Shriberg et al. 2000; Liu et al. 2006; Kim and Woodland 2003; Gotoh and Renals 2000). In those works, the text-only corpora consisted of additional telephone conversation transcripts, broadcast news transcripts, or news scripts, but some works have utilized punctuation in textual web resources. Shen et al. (2009) used an online encyclopedia to train a lexical model for punctuation annotation on news podcasts, and combined it with a pause duration model trained on another speech domain. Chen et al. (2010) used web text with commas in a domain adaptation strategy for prosodic phrase prediction in news text. Gravano et al. (2009) built lexical models from web news articles to annotate punctuation and capitalization in speech; model performance was compared on broadcast news speech and written news. Note that in the above works, performance on the task of automatic question mark insertion was reported only by Christensen et al. (2001), Huang and Zweig (2002), Shen et al. (2009), and Gravano et al. (2009), and was generally much worse than comma and period.

The contribution of prosodic features in addition to lexical features has been explored in several works, e.g., Shriberg et al. (2000); Christensen et al. (2001); Huang and Zweig (2002); Liu et al. (2006); Kim and Woodland (2003); Shen et al. (2009); Ang et al. (2005); Kolár et al. (2006b). These works mostly found that the best results on sentence boundary or punctuation detection were achieved by using both prosodic features (especially pause duration) and lexical features, although the improvement over just using one feature type varied. On the MRDA corpus specifically, Kolár et al. (2006b) found that lexical features alone outperformed prosodic features alone when using reference transcripts; furthermore,
pause duration features were responsible for most (but not all) of the improvement due to the addition of prosody features. Comparisons of features used for sentence segmentation across different speech styles have also been conducted in several works (Shriberg et al. 2000, 2009; Cuendet et al. 2007a,b). These showed that pause duration was very useful in all styles, whereas duration and lexical features were relatively more useful in the meetings and telephone conversations than in broadcast news. The importance of lexical features for meetings may be due to the high proportion of backchannel sentences (Cuendet et al. 2007b).

There has been some investigation of semi-supervised learning and domain adaptation for both dialog act tagging and segmentation. Guz et al. (2010b) performed supervised adaptation for both segmentation and dialog act tagging, between different spoken corpora (Swbd and MRDA). Jeong et al. (2009) performed unsupervised adaptation for dialog act tagging from a speech domain (MRDA/Switchboard) to text domains (emails and forums). Except for pause duration, which was used for segmentation in Guz et al. (2010b), these works did not use prosodic features. However, Venkataramanan et al. (2003) included prosodic features in a semi-supervised (bootstrapping) approach for dialog act labeling within a single spoken domain, and Kolář et al. (2010) described supervised speaker adaptation of language and prosodic models for segmentation within the MRDA corpus. The use of lexical and prosodic/acoustic features as two views in co-training, as we do here, has also been explored previously (Tur 2009; Guz et al. 2010a) for segmentation (and Guz et al. 2010a also explored self-training); and Jeon and Liu (2009b) used lexical/syntactic and acoustic views in co-training for detecting prosodic events, including phrase boundaries. Those works used labeled speech data and incorporated unlabeled speech data in the same domain; in our work, we attempt to adapt from punctuation-tagged web text to unlabeled speech data, incorporating prosodic features which are only available in the speech data.

As mentioned above, Chen et al. (2010) used web text to adapt a system used for prosodic phrase prediction in speech; the web text was used to augment the training data based on a bootstrapping approach, but only textual features were used.

In this work, we focus on spontaneous conversational speech, and utilize a web text source that is somewhat matched in style: both domains consist of goal-directed multi-party
conversations. We compare performance of textual- and speech-trained lexical models, and examine the detection accuracy of different question types and sentence unit boundaries. Finally, we compare domain adaptation approaches to utilize unlabeled speech data: bootstrapping, co-training, and Structural Correspondence Learning (SCL) (Blitzer et al. 2006). Although SCL has been applied to several NLP tasks, it has generally only been applied to lexical features; an exception is Sandu et al. (2010), who experimented with applying it (unsuccessfully) to “conversational features” in emails and spoken meetings, but the conversational features were defined in both domains (including features like sentence length, turn position, or “pause” duration). To our knowledge we are the first to apply it to incorporate prosodic features in one domain, in order to adapt from text to speech.

7.3 Data

As noted in Chapter 6, the MRDA corpus consists of English-language, computer science research meetings, with 3-9 speakers per meeting. We use the close-talking microphone recordings and the manual reference transcriptions for all experiments. The transcriptions include segmentation of the words into utterances (i.e., sentence units), which were used to define the units for question detection and the sentence boundary labels for sentence boundary detection.

The Wiki talk pages consist of threaded posts by different authors about a particular Wikipedia entry. While these lack certain properties of spontaneous speech (such as backchannels, disfluencies, and interruptions), they are more conversational than news articles. Table 7.1 gives an example of some dialog in both domains. The Wiki text contains a larger vocabulary (including misspellings), and on average longer sentences than MRDA. We first cleaned the posts (to remove URLs, images, signatures, Wiki markup, and duplicate posts) and then performed automatic segmentation on lines in each post, as described below.

---

2We use reference transcriptions from SRI that were post-processed for the SRI recognition systems.
Table 7.1: Examples of dialog in MRDA (top) and Wiki (bottom). (Note that the Wiki posts are shown here with their original punctuation and capitalization.)

| person1: | i was wondering          |
| person2: | is that okay             |
| person1: | yeah it’s okay           |
| person3: | sure                    |
| person4: | can you maximize the window |
| person1: | proceed                 |
| person2: | you want me to          |
| person2: | wait what do you want me to do |
| person4: | can you maximize the window so all that stuff on the side isn’t doesn’t appear |
| person3: | no it’s okay             |
| person3: | it’s it’ll work          |
| person2: | well i can do that      |
| person2: | but then i have to end the presentation in the middle so i can go back to open up javabayes |

person1: ... 3rd, i recall that there are first hand accounts by two witnesses that kerry received the wound in a firefight (see the 1st-hand accounts links). i’ve never seen the swiftvets claim that the ph was submitted after a transfer by kerry by his new CO, what’s your source on that? all that said, i definitely agree that kerry’s critics should be accurately represented, though i’m not yet sure that’s not currently the case.

person2: As you said, those are the facts as you believe them. But absent objective proof, much can only be characterized as speculation and should not be included in the article. I agree with Derex that if the SBVT have made claims they could be summarized and addressed - as I believe they are - but to put our own theories in there makes it a POV article.
7.4 Question Detection

7.4.1 Data Labeling and Division

For the Wiki data, we used the statistical sentence segmentation program MXTERMINATOR (Reynar and Ratnaparkhi 1997) to break lines into sentences. We also assume all line breaks and turn endings are sentence boundaries. (Anecdotally, line breaks often occur at paragraph boundaries, but sometimes also occur in the middle of sentences.) We labeled each sentence ending in a question mark (followed optionally by other punctuation) as a question; we also included parentheticals ending in question marks. All other sentences were labeled as non-questions. We then performed some additional text normalization to match the MRDA transcripts, such as verbalization of certain symbols, and of numbers, dates, and some common abbreviations; separation of letters in some common acronyms; and final removed all punctuation, numerals, and capitalization. All sentences lacking alphabetic characters were removed.

The MRDA corpus has been hand-annotated with detailed dialog act tags, using a hierarchical labeling scheme in which each utterance receives one “general” label plus a variable number of “specific” labels (Dhillon et al. 2004). In this work we are only looking at the problem of discriminating questions from non-questions; we consider as questions all complete utterances labeled with one of the general labels wh, yes-no, open-ended, or, or-after-yes-no, or rhetorical question. (To derive the question categories below, we also consider the specific labels tag and declarative, which are appended to one of the general labels.) Utterances labeled with multiple dialog acts were considered questions if any of the dialog act labels was in a question category. All other utterances, including statements, backchannels, floorgrabs/holds, and incompletes (including incomplete questions), are considered non-questions. We removed utterances that are very short (less than 200ms), tagged as non-speech, have no transcribed words, or are missing segmentation times or dialog act label. We performed minor text normalization on the transcriptions, such as mapping all word fragments to a single token, and removing the dot after letters. For prosodic feature extraction, we use waveforms defined by utterance times from a forced alignment provided by SRI (defined by the utterance start time and final punctuation time).
The Wiki training set consists of close to 46k utterances, with 8.0% questions. We derived an MRDA training set of the same size from the training division of the original corpus; it consists of 6.6% questions. For the adaptation experiments, we used the full MRDA training set of 72k utterances as unlabeled adaptation data. We used two meetings (3k utterances) from the original MRDA development set for model selection and parameter tuning. The remaining meetings (in the original development and test divisions; 26k utterances) were used as our test set for the results below.

7.4.2 Features and Classifier

Lexical features consisted of unigrams through trigrams including start- and end-utterance tags, represented as binary features (presence/absence), plus a total-number-of-words feature. All n-gram features were required to occur at least twice in the training set. The MRDA training set contained on the order of 60k n-gram features while the Wiki training set contained over 200k (there were 24k shared). Although some previous work has used part-of-speech or parse features in related tasks, Boakye et al. (2009) showed no clear benefit of these features for question detection on MRDA beyond the n-gram features.

We extracted 16 prosody features from the speech waveforms defined by the given utterance times, using stylized F0 contours computed based on Sönmez et al. (1998) and Lei (2006). The features are designed to be useful for detecting questions and are similar or identical to some of those in Boakye et al. (2009) or Shriberg et al. (1998). They include: F0 statistics (mean, stdev, max, min) computed over the whole utterance and over the last 200ms; slopes computed from a linear regression to the F0 contour (over the whole utterance and last 200ms); initial and final slope values output from the stylizer; initial intercept value from the whole utterance linear regression; ratio of mean F0 in the last 400-200ms to that in the last 200ms; number of voiced frames; and number of words per frame. All 16 features were z-normalized using speaker-level parameters, or gender-level parameters if the speaker had fewer than 10 utterances.

For all experiments we used logistic regression models trained with the LIBLINEAR package (Fan et al. 2008). Prosodic and lexical features were combined by concatenation
into a single feature vector; prosodic features and the number-of-words were z-normalized to place them roughly on the same scale as the binary n-gram features. (We substituted 0 for missing prosody features due to, e.g., no voiced frames detected, segmentation errors, utterance too short.) Our setup is similar to Surendran and Levow (2006), who combined n-gram and prosodic features for dialog act classification using a linear SVM. Since ours is a detection problem, with questions much less frequent than non-questions, we present results in terms of ROC curves, which were computed from the probability scores of the classifier. The cost parameter $C$ was tuned to optimize Area Under the Curve (AUC) on the development set ($C = 0.01$ for prosodic features only and $C = 0.1$ in all other cases). AUC is an attractive performance measure in this case because it does not take into account the relative frequencies of the classes, and so will not be dominated by performance on the non-question class. However, for comparisons with previous work, we also present results in terms of the more standard F measure, for which we tuned both $C$ and the decision threshold on the development set. ($C$ was found to be the same as for AUC, except for training on MRDA with lexical and prosodic features, which used $C = 1$).

7.4.3 Baseline Results

Figure 7.1 shows the ROC curves for the baseline Wiki-trained lexical system and the MRDA-trained systems with different feature sets. The MRDA system with lexical and prosodic features does slightly better than the MRDA system with just lexical features; the Wiki system with lexical features does substantially worse than both systems, but better than the MRDA system with just prosodic features.

Table 7.2 compares performance across different question categories at or near a fixed false positive rate (16.7%). This analysis point is starred in Figure 7.1 and is the equal error rate of the MRDA (lex) case. For analysis purposes we defined the categories in Table 7.2 as follows: tag includes any yes-no question given the additional tag label; declarative includes any question category given the declarative label that is not a tag question; the remaining

---

3 Some false positive rates are not exactly achievable for some systems, due to the fact that the ROC curves are derived from empirical distributions of scores and multiple examples may have the same score. Therefore, when we evaluate systems “near” a target false positive rate, we pick the closest false positive rate less than or equal to the target false positive rate.
Figure 7.1: ROC curves with AUC values for question detection on MRDA; comparison between systems trained on MRDA using lexical and/or prosodic features, and Wiki talk pages using lexical features. Starred (+) points occur near a false positive rate of 16.7%, which is the equal error rate point for the MRDA (lex only) system; they represent the analysis points used for Table 7.2.
Table 7.2: Question detection rates (%) by question type for each system (L=lexical features, P=prosodic features). Detection rates are given near a false positive rate of 16.7% (starred points in Figure 7.1), which is the equal error rate point for the MRDA (L) system. Boldface gives the best result for each type.

<table>
<thead>
<tr>
<th>question type (count)</th>
<th>MRDA (L+P)</th>
<th>MRDA (L)</th>
<th>MRDA (P)</th>
<th>Wiki (L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes-no (526)</td>
<td>89.4</td>
<td>86.1</td>
<td>59.3</td>
<td>77.2</td>
</tr>
<tr>
<td>declarative (417)</td>
<td>69.8</td>
<td>59.2</td>
<td>49.4</td>
<td>25.9</td>
</tr>
<tr>
<td>wh (415)</td>
<td>95.4</td>
<td>93.0</td>
<td>42.2</td>
<td>92.8</td>
</tr>
<tr>
<td>tag (358)</td>
<td>89.7</td>
<td>90.5</td>
<td>26.0</td>
<td>79.1</td>
</tr>
<tr>
<td>rhetorical (75)</td>
<td>88.0</td>
<td>90.7</td>
<td>25.3</td>
<td>93.3</td>
</tr>
<tr>
<td>open-ended (50)</td>
<td>88.0</td>
<td>92.0</td>
<td>16.0</td>
<td>80.0</td>
</tr>
<tr>
<td>or (38)</td>
<td>97.4</td>
<td>100</td>
<td>29.0</td>
<td>89.5</td>
</tr>
<tr>
<td>or-after-YN (32)</td>
<td><strong>96.9</strong></td>
<td><strong>96.9</strong></td>
<td>25.0</td>
<td>90.6</td>
</tr>
</tbody>
</table>

categories (yes-no, or, etc.) include utterances in those categories but not included in declarative or tag. Table 7.3 gives example sentences for each category.

Not surprisingly, the Wiki-trained system does worst on declarative questions, which have the syntactic form of statements. For the MRDA-trained system, prosody alone does best on yes-no and declarative questions. Along with lexical features, prosody is more useful for declarative questions, while it appears to be somewhat redundant with lexical features for yes-no questions. Ideally, such redundancy can be used together with unlabeled spoken utterances to incorporate prosodic features into the Wiki system, which may improve detection of some kinds of questions.

Table 7.4 shows F measure results. Our F measure for in-domain training with prosodic features is close to that reported in Boakye et al. (2009). Our F measures for in-domain training with lexical and with lexical/prosodic features are somewhat worse than the reported values in Boakye et al. (2009) (0.67 and 0.69). Note that their task is not strictly
Table 7.3: Examples for each MRDA question category as defined in this chapter, based on Dhillon et al. (2004).

<table>
<thead>
<tr>
<th>yes-no</th>
<th>did did you do that?</th>
</tr>
</thead>
<tbody>
<tr>
<td>declarative</td>
<td>you’re not going to be around this afternoon?</td>
</tr>
<tr>
<td>wh</td>
<td>what do you mean um reference frames?</td>
</tr>
<tr>
<td>tag</td>
<td>you know?</td>
</tr>
<tr>
<td>rhetorical</td>
<td>why why don’t we do that?</td>
</tr>
<tr>
<td>open-ended</td>
<td>do we have anything else to say about transcription?</td>
</tr>
<tr>
<td>or</td>
<td>and @frag@ did they use sigmoid or a softmax type thing?</td>
</tr>
<tr>
<td>or-after-YN</td>
<td>or should i collect it all?</td>
</tr>
</tbody>
</table>

Table 7.4: F measure for the baseline MRDA and Wiki question detection systems, for which both $C$ and the decision threshold were tuned on the MRDA development set.

<table>
<thead>
<tr>
<th></th>
<th>F measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRDA (L+P)</td>
<td>0.603</td>
</tr>
<tr>
<td>MRDA (L)</td>
<td>0.565</td>
</tr>
<tr>
<td>MRDA (P)</td>
<td>0.256</td>
</tr>
<tr>
<td>Wiki (L)</td>
<td>0.496</td>
</tr>
</tbody>
</table>

Comparable since they attempt to distinguish questions from statements, without including disrupted or backchannel utterances.

7.4.4 Adaptation Results

For bootstrapping, we first train an initial baseline classifier using the Wiki training data, then use it to label MRDA data from the unlabeled adaptation set. We select the $k$ most confident examples that were given the “question” label, and the $k$ most confident examples that were given the “non-question label”, and add them all to the training set using the guessed labels, then retrain the classifier using the new training set. This is repeated for
rounds. In order to use prosodic features, which are available only in the bootstrapped MRDA data, we simply add 16 zeros onto the Wiki examples in place of the missing prosodic features. The values $k = 20$ and $r = 6$ were selected on the dev set.

For SCL, we use as auxiliary tasks all initial words that begin an utterance at least 5 times in each domain’s training set, and predict the presence of each initial word from the other words and prosody features. The idea of using the initial words is that they may be related to the interrogative status of an utterance—utterances starting with “do” or “what” are more often questions, while those starting with “I” are usually not. There were about 250 auxiliary tasks. The prediction features for each auxiliary task include all n-grams occurring at least 5 times in the unlabeled Wiki or MRDA data, except those over the first word, as well as prosody features (which are zero in the Wiki data). Logistic regression classifiers from LIBLINEAR are used to learn the auxiliary tasks; we use the same value $C = 0.1$ as the baseline classifier. We tuned the number of SCL features (100) and their scale factor (1) on the dev set.

Figure 7.2 compares the results using the bootstrapping and SCL approaches, and the baseline unadapted Wiki system. Table 7.5 shows results by question type at the fixed false positive point chosen for analysis. At this point, both adaptation methods improved detection of declarative and yes-no questions, although they decreased detection of several other types.

We experimented with other adaptation approaches on the dev set, which did not give improved results and were not tested on the test set. These methods included: bootstrapping without the prosodic features; co-training approaches using separately prosodic and lexical classifiers, as inspired by Guz et al. (2010a); and using “fake” prosody features to train on the Wiki data, which were predicted from the word features using a linear regression model learned on the MRDA data.

Since we tuned and selected adaptation methods on the MRDA dev set, we compare our adapted systems to a system that used the MRDA dev data in training—we simply added the 3.1k MRDA utterances with prosodic features into the training set, with the prosodic features effectively 0 in the Wiki utterances. This system is represented as “include MRDA dev” in Figure 7.2 and it outperforms both adaptation methods.
Figure 7.2: ROC curves and AUC values for adaptation, baseline Wiki, and Wiki + MRDA dev on the question detection task. Starred (+) points represent the same false positive rate marked in Figure 7.1 and used for analysis in Tables 7.2 and 7.5.
Table 7.5: Detection rates of adapted systems for each question type, near a false positive rate of 16.7% (starred points in Figure 7.2). Boldface indicates adaptation results better than baseline; italics indicate worse than baseline.

<table>
<thead>
<tr>
<th>question type (count)</th>
<th>baseline</th>
<th>bootstrap</th>
<th>SCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes-no (526)</td>
<td>77.2</td>
<td>81.4</td>
<td>83.5</td>
</tr>
<tr>
<td>declarative (417)</td>
<td>25.9</td>
<td>30.5</td>
<td>32.1</td>
</tr>
<tr>
<td>wh (415)</td>
<td>92.8</td>
<td>92.8</td>
<td>93.5</td>
</tr>
<tr>
<td>tag (358)</td>
<td>79.1</td>
<td>79.3</td>
<td>80.7</td>
</tr>
<tr>
<td>rhetorical (75)</td>
<td>93.3</td>
<td>88.0</td>
<td>92.0</td>
</tr>
<tr>
<td>open-ended (50)</td>
<td>80.0</td>
<td>76.0</td>
<td>80.0</td>
</tr>
<tr>
<td>or (38)</td>
<td>89.5</td>
<td>89.5</td>
<td>89.5</td>
</tr>
<tr>
<td>or-after-YN (32)</td>
<td>90.6</td>
<td>90.6</td>
<td>90.6</td>
</tr>
</tbody>
</table>

In addition to AUC, we also consider F measure, shown in Table 7.6. To derive F measure values, we re-tuned the parameters in the adaptation methods using the MRDA dev set ($r = 1$ for bootstrapping and $d = 20$ for SCL), and also tuned the decision thresholds to optimize F measure on the dev set. We include results of the system trained on the labeled MRDA dev and Wiki data together, using the decision threshold found from tuning the Wiki lexical system on the MRDA dev set. Bootstrapping was not successful at improving F measure, and SCL only improved it slightly. Due to the high proportion of non-questions, optimizing F measure generally requires a low false positive rate, but Figure 7.2 indicates that the adaptation approaches are likely not successful at such points on the far left of the ROC curve. However, including the small amount of MRDA labeled data in training leads to the largest improvement, even though this uses the decision threshold of the Wiki-only system.

Our analysis of the adapted systems suggests prosody features are being utilized to

---

4Note that the reported models in Table 7.6 would have different ROC curves, since we tuned the adaptation parameters $r$ and $d$ differently, but we expect that the behavior would be similar.
Table 7.6: F measure for the adapted Wiki systems for question detection; the adaptation parameters and decision threshold were tuned on the MRDA development set.

<table>
<thead>
<tr>
<th>Method</th>
<th>F Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>0.496</td>
</tr>
<tr>
<td>bootstrap</td>
<td>0.493</td>
</tr>
<tr>
<td>SCL</td>
<td>0.506</td>
</tr>
<tr>
<td>MRDAdev + Wiki</td>
<td>0.540</td>
</tr>
</tbody>
</table>

improve performance. In fact, neither method could achieve an improvement on the dev set without the prosody features. However, the adapted systems do not exceed performance of the system trained using the dev data. So while these methods can apparently achieve adaptation from a small amount of labeled speech data and a large amount of unlabeled speech data, the small amount of labeled speech data can be used more effectively (and simply) by adding it to the training set. An open question is whether the adaptation methods might be superior under conditions where the dev set was smaller.

7.5 Sentence Boundary Detection

7.5.1 Data Labeling and Division

Since the focus of this section is sentence segmentation, we did not want to use a statistical model such as MXTERMINATOR, which has been trained on its own corpus (Wall St. Journal). Therefore, in this task we split the Wiki sentences using a non-statistical segmenter from the perl module “Lingua::Sentence,” which uses punctuation/capitalization rules and a list of exceptions. Following segmentation, additional normalization was applied as described in Section 7.4.1. We associated each sentence boundary with the word preceding that boundary, and performed training and detection on the word level.

We classify words in each speaker’s channel, processed as a single long segment. Periods of time in which others are speaking are represented as long pauses. We performed binary boundary/no-boundary classification at each word (referring to the boundary after
the word), using the reference word sequence aligned to the waveforms. Some very short segments such as “s-”, “uh”, “like” were not aligned successfully, and words in those segments were dropped, i.e., not included in the features or in classification; the waveform in those regions is aligned to pauses. Censored segments of the corpus were treated similarly. (Censored segments are segments where words are not transcribed, and where the speech is replaced by a single tone in the waveform.) We performed some minor additional processing of the force-aligned words prior to lexical feature extraction, namely, removal of “.” in spoken letters to match the form of the processed Wiki words.

There are many more sentence boundaries than questions, so rather than the two-meeting development set used for question detection, we use as a development set only the first 1000 words from one speaker in one of the meetings. (We refer to this as the “MRDA dev” set.) Otherwise we use the same training set (of 51 meetings), but we do not subsample the MRDA training set for the in-domain comparison experiments, since there are less MRDA words to begin with: the Wiki training set contains 787k words, while the MRDA training set contains 538k words. Unlike the question detection task, there are very different class proportions for the Wiki and MRDA data, due to the fact that Wiki sentences are longer. About 14% of MRDA words occur before sentence boundaries, vs. 5% of Wiki words. For the test set, we use 10 meetings from the original “dev” division (excluding the one from which our small development set is drawn). The test set has 100k words.

7.5.2 Features and Classifier

As lexical features, we use the current, previous, and next words, and the two bigrams within that sequence (previous/current, current/next). These features are position-dependent, e.g., there is one feature representing the presence of “and” at the current-word position and a different feature representing the presence of “and” at the previous-word position. For the Wiki data, we did not extract features across post boundaries but we included a start/end symbol at these boundaries. For the MRDA data, start/end symbols were used for words at the beginning and end of the speaker’s word sequence. There are about 560k features in

\footnote{We thank Wen Wang at SRI for the forced alignment.}
the Wiki training set and 300k in the MRDA training set (84k shared).

For the MRDA data, we extracted prosody features from each word/word boundary, using word- and phone-level alignment times from the forced alignment. We emphasize that these are word-level features, computed for each word, in contrast with those used above for question detection, which are utterance-level features. In total we used 45 prosody features, representing pitch, energy and duration information (including pause duration); these are similar to the features used in Chapter 5. (In general we selected only one feature for each measurement, preferring the normalized version, so for example, we use a word duration feature normalized by average phone durations, but we do not use raw, un-normalized word duration.) In addition, we use a single binary feature to indicate a turn change; we define a turn change to occur at a given word boundary if both (a) the speaker pauses for at least 0.5 seconds, and (b) another speaker talks before the given speaker’s next word. All prosodic features except for the turn indicator are numeric.

As the classifiers, we build logistic regression models from LIBLINEAR, with prosodic and lexical features concatenated into a single feature vector, as above. All prosodic features are z-normalized using parameters computed over the training set; missing features were set to 0. We tuned the \( C \) value for training LIBLINEAR using the 1000-word development set, and also tuned the threshold for evaluating F measure. (We used \( C = 1 \) for all cases except when training on MRDA with prosodic features only, which used \( C = 100 \) for AUC and \( C = 1 \) for F measure.)

### 7.5.3 Baseline Results

Figure 7.3 shows AUC results for the baseline MRDA and Wiki systems. Table 7.7 shows detection rates across different boundary categories for a fixed false positive rate near 11% (the closest point at or below the equal error rate point of the MRDA lexical-only system). We defined the boundary categories using dialog act tags (specifically, a simplified dialog act tag provided by SRI) so that category of a sentence boundary is defined by the dialog act preceding the boundary. In cases of multiple dialog act tags for different parts of the utterance, we selected the final one. Note that some segments (many at the beginnings and
Table 7.7: Sentence boundary detection rates (%) by boundary type for each system (L=lexical features, P=prosodic features). Detection rates are given at a false positive rate near 11% (starred points in Figure 7.3), which is the equal error rate point for the MRDA (L) system. Boldface gives the best result for each type. Final row represents sentence boundaries whose type is not annotated.

<table>
<thead>
<tr>
<th>boundary type (count)</th>
<th>MRDA (L+P)</th>
<th>MRDA (L)</th>
<th>MRDA (P)</th>
<th>Wiki (L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>statement (7876)</td>
<td>96.4</td>
<td>91.7</td>
<td>72.6</td>
<td>60.6</td>
</tr>
<tr>
<td>incomplete (2080)</td>
<td>88.5</td>
<td>73.9</td>
<td>78.9</td>
<td>40.5</td>
</tr>
<tr>
<td>backchannel (1760)</td>
<td>99.7</td>
<td>98.8</td>
<td>96.2</td>
<td>59.7</td>
</tr>
<tr>
<td>floor grab/hold (957)</td>
<td>93.9</td>
<td>83.0</td>
<td>82.4</td>
<td>39.8</td>
</tr>
<tr>
<td>question (901)</td>
<td>94.1</td>
<td>87.1</td>
<td>73.9</td>
<td>63.8</td>
</tr>
<tr>
<td>boundary, unknown type (201)</td>
<td>97.5</td>
<td>90.5</td>
<td>90.0</td>
<td>65.7</td>
</tr>
</tbody>
</table>

ends of meetings) do not have dialog act tags; these are listed in our table as “boundary, unknown type.”

We note first some differences from the question detection task. Prosodic features are more useful alone for this task, and are useful when added to lexical features for all boundary types in Table 7.7. Based on previous work, such as Shriberg et al. (2000), we assume pause duration is one of the most useful prosodic features, and it could be more useful for this task than the pitch/speaking rate/voicing features are for the question detection task. Comparing the Wiki and MRDA classifiers, all MRDA classifiers outperform the Wiki classifier for all boundary types. Furthermore, the MRDA classifiers with lexical features achieve higher AUC’s than any in-domain classifier in the question detection task, although the Wiki classifier has similar AUC with the Wiki question detector.

Comparing the MRDA- and Wiki-trained classifiers, the Wiki classifier does worse for all boundary types. It does particularly poorly on floors and incompletes, which are obviously missing from the Wiki domain, but interestingly does better on backchannels, which are
Figure 7.3: ROC curves with AUC values for sentence boundary detection on MRDA; comparison between systems trained on MRDA using lexical and/or prosodic features, and Wiki talk pages using lexical features. Starred (+) points occur near a false positive rate of 11%, which is the equal error rate point for the MRDA (lex only) system; they represent the analysis points used for Table 7.7.
also missing. Analysis shows that one common backchannel word, “huh”, is present in the Wiki data, and often precedes a boundary. Other backchannels, such as “uhhuh”, which are entirely missing, are apparently sometimes identified based on the previous/next words. We also note that when training on MRDA, backchannels are more reliably identified based on lexical features than either floors or incompletes.

Table 7.8 reports F measure results for systems where the decision threshold and C value were both tuned to optimize F measure on the development set. In comparison with previous work, Cuendet et al. (2007b); Shriberg et al. (2009) reported F measure results for in-domain training using manual transcripts as: 0.749 for lexical features, 0.710 for prosodic features, and 0.820 for both lexical and prosodic features, using the same set of meetings for training but excluding boundaries with unlabeled type. These results are slightly better than ours, particularly the result with prosodic features alone. The differences may be due to lexical features (they use the trigram consisting of the previous-current-next words and do not use the previous-current bigram), and the fact that they use a larger number of prosodic features. They also use a boosting classifier rather than a linear classifier, which may be superior for combining lexical and prosodic features and for handling missing features. Despite this, our results are close, and our use of the logistic regression classifier is more convenient for adaptation experiments, particularly Structural Correspondence Learning, where we can encourage use of certain features by scaling.

Table 7.8: F measure for the baseline MRDA and Wiki sentence boundary detection systems, for which both C and the decision threshold were tuned on the MRDA development set.

<table>
<thead>
<tr>
<th>System</th>
<th>F Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRDA (L+P)</td>
<td>0.810</td>
</tr>
<tr>
<td>MRDA (L)</td>
<td>0.737</td>
</tr>
<tr>
<td>MRDA (P)</td>
<td>0.696</td>
</tr>
<tr>
<td>Wiki (L)</td>
<td>0.498</td>
</tr>
</tbody>
</table>
7.5.4 Adaptation Results

We experimented with three adaptation methods: bootstrapping, co-training, and SCL, testing these methods first on the MRDA dev set. Bootstrapping was implemented as described above; we tuned \( r = 30 \) iterations and added \( k = 20 \) examples per class on each iteration (\( k \) was not tuned).

Unlike the question detection task, co-training (with lexical and prosodic features as two views) was effective for this task. This difference makes sense given the results in Section 7.5.3 which showed that a classifier based on prosodic features alone is more effective for this task than for question detection. We experimented with different co-training configurations. In one configuration, the lexical classifier is trained first using the Wiki data, and used to label some MRDA data, which is used to train the initial prosodic classifier; subsequently, each classifier adds labeled data to the opposite classifier. Data is drawn from a single unlabeled pool, but separate training sets are maintained for each classifier, with both Wiki and MRDA data used for the lexical classifier. In an alternative configuration, there exists a single labeled training set used for both classifiers (as was done in the original co-training paper (Blum and Mitchell 1998)). Following the “co-adaptation” approach proposed in Christoudias et al. (2006), the training set is initialized from out-of-domain classifiers; we used an out-of-domain lexical classifier trained on the Wiki data. Finally, we considered a version of the latter configuration where the Wiki data was kept in the training set, but only used when training the lexical classifier. None of these configurations gave substantially different results on the dev set, so for our final system we used the first configuration with two training sets. We tuned \( r = 15 \) iterations and \( k = 20 \) examples per class on each iteration (\( k \) was not tuned). In all cases, the final classifier is the lexical classifier after the final iteration.

For SCL, we use two types of auxiliary tasks, corresponding to “current” and “next” word feature predictions. For auxiliary tasks predicting current word features, only the next word is used. For auxiliary tasks predicting next word features, both the current word and prosody features are used (which are 0 in the Wiki data). For example, one auxiliary task belonging to the first type is prediction of whether or not the current word is “the”, given the
following word. One auxiliary task belonging to the second type is prediction of whether or not the next word is “the”, given the current word and prosodic features. Neither the bigram features nor the previous word features are included in SCL. The separation of auxiliary tasks into “views” is similar to the approach in Blitzer et al. (2006) for part-of-speech tagging, which used auxiliary tasks corresponding to current, previous, and next words, with prediction features from the other views. We initially attempted to use the prosodic features in both views, but that appeared to lead to “misalignments.” One possible reason for this is that the prosody features are correlated with the current word as a result of factors other than boundary status. So if the prosody features are used to predict current word features, the collection of learned weight vectors might characterize some aspect of the current word other than boundary status, such as length, part-of-speech, or likelihood of prominence. SCL features are learned separately for each of two views; we then add the SCL features from each view onto the final feature vector. We tuned a single value for the number of features in each view (i.e., we did not tune the number in each view separately).

We experimented with different occurrence thresholds for the pivots and predictor features, based on performance and computational considerations. Ultimately, we required pivot features to occur at least 10 times in each domain, and used the top 400 pivot features for each view by mutual information with the class label in the Wiki data, as described in Blitzer et al. (2007) (“SCL-MI”). We only included predictor words that occur at least 10 times in the unlabeled (MRDA + Wiki) data; this leads to a smaller dimensionality in the auxiliary tasks and SVD, and also possibly serves as regularization. We used logistic regression models to learn the auxiliary tasks, with cost parameter $c = 0.1$, rather than $c = 1$ used in the main task; empirically, we have seen that a smaller $c$ value (increased regularization) may be useful for SCL in some problems. We tuned the number of SCL dimensions to $h = 50$ for each view (100 total), and the scale factor to 1.

Figure 7.4 shows the ROC curves for the adaptation methods on the test set, and Table 7.9 gives the results by boundary type at the false positive point chosen for analysis. All three adaptation methods improved AUC compared to the unadapted Wiki-only baseline, and at the analysis point, they improved performance for all boundary types. In terms of AUC, co-training was superior to bootstrapping; co-training and SCL had similar perfor-
Figure 7.4: ROC curves and AUC values for adaptation, baseline Wiki, and Wiki + MRDA dev on the sentence boundary detection task. Starred (+) points represent the same false positive rate marked in Figure 7.3 and used for analysis in Tables 7.7 and 7.9.

Although the ROC curves for co-training and SCL are similar, Table 7.9 shows some differences by boundary type, with co-training having better detection of backchannels and floor-grab/holds, and SCL having better detection of statements and questions. Although we do not know the cause of these differences, note that co-training differs from the other two approaches in that it involves models (from the prosody view) trained only on (auto-labeled) MRDA data. As Table 7.7 shows, backchannel boundaries have the highest detection rate for the MRDA-trained prosody model, so it is possible that the prosody view co-training models have similar behavior, leading to overall high detection of backchannels.
Table 7.9: Detection rates of adapted systems by sentence boundary type, near a false positive rate of 11% (starred points in Figure 7.4). All adapted results are better than baseline.

<table>
<thead>
<tr>
<th>boundary type (count)</th>
<th>baseline</th>
<th>bootstrap</th>
<th>co-train</th>
<th>SCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>statement (7876)</td>
<td>60.6</td>
<td>63.0</td>
<td>66.4</td>
<td>70.7</td>
</tr>
<tr>
<td>incomplete (2080)</td>
<td>40.5</td>
<td>45.3</td>
<td>46.2</td>
<td>48.6</td>
</tr>
<tr>
<td>backchannel (1760)</td>
<td>59.7</td>
<td>70.9</td>
<td>94.9</td>
<td>77.8</td>
</tr>
<tr>
<td>floor grab/hold (957)</td>
<td>39.8</td>
<td>44.9</td>
<td>54.8</td>
<td>43.6</td>
</tr>
<tr>
<td>question (901)</td>
<td>63.8</td>
<td>66.4</td>
<td>66.6</td>
<td>71.1</td>
</tr>
<tr>
<td>boundary, unknown type (201)</td>
<td>65.7</td>
<td>73.6</td>
<td>74.1</td>
<td>78.1</td>
</tr>
</tbody>
</table>

Table 7.10 gives F measure results, using a decision threshold tuned on the MRDA dev set. All methods outperform the baseline F measure, but co-training performs the best.

We compare the adaptation methods to the approach that uses the 1000-word MRDA dev set in training. This was implemented in the same way as for the question detection task: we simply added the MRDA data, with both lexical and prosodic features, to the Wiki training data, which had 0’s for the prosodic features. To report F measure, we use a decision threshold from the Wiki-trained system tuned on the MRDA dev data. As shown in Figure 7.4 and Table 7.10 this simple approach gives an AUC value and F measure that are substantially better than any of the adaptation approaches, even though the dev data is taken from only one speaker.

This raises a question of whether the adaptation methods could be useful in a real setting. Our results showed that the methods achieved adaptation, but only after being tuned on a labeled set from the target domain, and in our case better performance could be achieved by using the target data for training in a way that did not require any additional tuning. An argument for the adaptation approach might be made for the case when less labeled target data were available—enough to give an estimate of target domain performance, but not
Table 7.10: F measure for the adapted Wiki systems for sentence boundary detection; the adaptation parameters and decision threshold were tuned on the MRDA dev set.

<table>
<thead>
<tr>
<th>Method</th>
<th>F measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>0.498</td>
</tr>
<tr>
<td>bootstrap</td>
<td>0.528</td>
</tr>
<tr>
<td>co-train</td>
<td>0.569</td>
</tr>
<tr>
<td>SCL</td>
<td>0.561</td>
</tr>
<tr>
<td>MRDAdev + Wiki</td>
<td>0.623</td>
</tr>
</tbody>
</table>

enough to train a classifier. Figure 7.5 shows the AUC performance when the Wiki+MRDA classifier is trained with smaller amounts of MRDA data, with and without prosody features. This shows that, with the prosody features, performance is noisy when using less than 100 MRDA words, but with 100 or more words, improvement over the baseline does better than tuned adaptation via bootstrapping, and with 500 words, it does better than co-training and SCL. Note that the prosodic portion of the model, which consists of weights on 46 features, is being learned only on the MRDA examples, which may explain the noisiness when the number of MRDA words added is small. The large improvement with only 100 words may be due to the importance of a few prosody features, e.g., pause duration, which can be learned using a relatively small number of examples.

To analyze whether our adaptation methods would have any advantage over using the dev set in training, we could compare the performance of the adaptation methods using smaller dev sets with the corresponding performances in 7.5. However, it is difficult to “honestly” assess the performance that we would have gotten had we used a smaller development set, because the process involved designing and testing several different approaches. For example, the dev set was used to decide whether to use the prosody features to predict the pivots in both or one view, or whether to use one or two labeled pools in co-training. Furthermore, the process of “tuning” parameters, such as the number of iterations in co-training, was based partly on human judgement—we did not simply select the iteration with the best performance, but also considered performance of nearby iterations and the
Figure 7.5: AUC values on the MRDA test set for the system trained on Wiki + MRDA development data, with and without prosodic features, as a function of the amount of MRDA data added.
overall combination of AUC and F measure. In some sense, the experimental paradigm we followed does not really permit comparison between using a dataset as a development set vs. using it to train the classifier.

Despite this, we did examine the performance of the final tuning step for SCL, which involved selecting the number of dimensions $h$ (out of values 10, 20, 50, 75, and 100) and the scale factor (out of values 0.1, 0.5, and 1), and comparing performance with and without including prosodic features. We observed that, when using only a dev set of size 100, this step yielded the same “optimal” values for these settings as were used above. This suggests that, given some knowledge about a useful adaptation design, final tuning steps might be conducted using only a small amount of labeled target data, which would otherwise be too small to reliably train a model from scratch.

Finally, note that our dev set used data from only one speaker. This seems reasonable when assuming only a small amount of labeled data is available. However, if data with multiple speakers were available, better results might be achieved (either when using the data for tuning or when using it for training).

### 7.6 Conclusions

We investigated the use of online, textual conversations for annotating questions and sentence boundaries in transcribed, spoken conversations, without requiring any manually labeled data for training. We additionally investigated domain adaptation methods which use unlabeled data from the spoken conversations, and which can incorporate prosodic features. On both tasks, we find that, when a small labeled development set from the speech domain is available for selecting methods and tuning parameters, bootstrapping and Structural Correspondence Learning (Blitzer et al. 2006) can both improve Area-Under-the-Curve (AUC) on a held-out test set from the speech domain. Co-training is also effective on the sentence boundary task, but not on the question detection task. Furthermore, we find that simply using the small speech development sets for training gives even better performance than the adaptation methods on both tasks.

One open question is whether very small quantities of labeled speech data would be better employed for tuning adaptation methods vs. used as training data. Another open
question is whether, given the extremely large quantities of textual data available on the Web, we might be able to sample the portion used as the training data in a way to better match the spoken data, for example, by selecting shorter sentences (to better match the utterance length in the spoken data). Finally, perhaps the methods discussed in this chapter might be combined for better performance—for example, perhaps bootstrapping might be applied using SCL features, or the development data might be used in training along with SCL features.
Chapter 8

CONCLUSIONS

In this chapter, we summarize our work in the previous chapters, then discuss our contributions with respect to both spoken language processing and domain adaptation. We then propose some directions for future work.

8.1 Summary

In Chapter 2, we surveyed methods for unsupervised domain adaptation and summarized the previous applications of these methods. We found that some approaches, such as instance weighting for covariate shift, are well-motivated theoretically given certain assumptions about the domains and classifier, but it is unclear whether those assumptions are likely to hold in real domain adaptation scenarios. Other methods, like feature restriction, make intuitive sense but the assumptions supporting them have not been clearly investigated. That motivated our work in Chapter 3, where we analyzed different change-of-feature-representation strategies, and in Chapter 4, where we considered instance weighting.

In Chapter 3, we analyzed the feature restriction approach, starting with the assumption of a linear generating model for the label variable $Y$ given $X$, which is assumed to hold for both domains. We assumed three kinds of features: source-specific $X_s$, target-specific $X_t$, and shared $X_b$; and we considered transferring two different models from the source domain to the target domain: the “full” source domain model over $X_s, X_b$ or the “shared-only” source domain model over $X_b$ only. We showed that “positive correspondence” is one condition under which the shared-only source domain model transfers better than the full source domain model to the target domain, and this condition occurs when, roughly, the source-specific features have the same set of relationships to the shared features and label variable as do the target-specific features. LSA on a merged collection of source and target examples may also be effective in the positive correspondence condition, but LSA features
are likely to describe variation between domains, which can degrade transfer performance. We showed that SCL makes a similar assumption to positive correspondence; in particular, it finds new source-specific and target-specific features, which have the same relationships to a set of shared features, and it assumes they have the same relationship to the label variable. LSA and SCL are more powerful than feature restriction since they learn a “full” model that includes the target-specific features, but we argued that SCL is superior to LSA in that it is less likely to include features that describe variations between domains. We compared all three methods on synthetic and real benchmark domain adaptation problems. This illustrated that SCL is more robust than the other methods to some degradations of the assumptions. However, even SCL was not effective on all benchmark datasets and led to large degradation in some cases. We experimented with methods to automatically remove “bad” LSA or SCL features, an approach motivated by previous theoretical work on transfer generalization error bounds. However, this was generally ineffective.

In Chapter 4, we explored the impact of instance weighting, together with regularization, on expected target domain generalization error, under the covariate shift scenario. In particular, we explored the effect of using a weight adaptive parameter from Shimodaira (2000) together with $L_2$ regularization. We decomposed the excess risk into estimation error, approximation error, and domain mismatch error, which we described qualitatively and then analyzed in a ridge regression example. Our synthetic example illustrated that expected target domain generalization error can be reduced by using both instance weighting and regularization together, compared with using only one approach, under the covariate shift scenario. It also illustrated that the optimal setting for the regularization parameter is not always the same for the source domain risk as for the target domain risk. We then turned to the benchmark document classification tasks, and described two methods for estimating instance weights: one based on language modeling and the other based on the discriminative domain prediction method of Bickel et al. (2007). We compared both methods in a cheating experiment where actual target domain data was mixed into the training set. We found that instance weighting with “oracle” weights did not achieve higher accuracy than the baseline (no weighting), but was beneficial when regularization was large. We also found that the language model length-adapted weight estimation method achieved slightly higher accuracy.
than the other methods. However, on the real tasks, instance weighting had little effect; in fact, the effect of changing the regularization weight was often larger.

Chapter 5 focused on prosody classification, specifically detection of words with pitch accents and detection of prosodic breaks. We explored cross-corpus training using two English corpora, one from the read news domain, and one from the spontaneous conversational domain. We compared cross-corpus training to in-corpus training using similarly-sized training sets with different feature sets (acoustic, textual, and both). For the accent detection task, we found that models built with textual features suffered the largest cross-corpus degradation and that the cross-corpus classifiers with all features did no better than the cross-corpus classifiers with just acoustic features. For the break detection task, in most cases the models built with all features achieved the best cross-corpus performance. We also showed that, for both tasks, BoosTexter and logistic regression models built with both training sets showed little or no degradation compared with models built using only one training set. This suggests that the conditional label probabilities are similar across domains, and that furthermore, sample selection bias is not a major factor in cross-domain degradation. Unsupervised domain adaptation strategies were generally unsuccessful. We found that corpus-level z-normalization of acoustic features, class proportion adjustment, co-training and self-training each improved performance in some task and direction while causing degradation in others; instance weighting generally had no effect.

In Chapter 6, we explored text-based dialog act tagging with high-level categories (statement, question, backchannel, incomplete) across corpora in different languages using machine translation. Degradation of the cross-language, cross-corpus model compared with the same-corpus model was substantial, although its performance was better than chance. We showed that degradation was due primarily to differences between the translated and original-language utterances, not due to loss of information in the translated utterances. We implemented a version of SCL using auxiliary tasks constructed by dividing the utterances into two parts, the first word and the rest of the utterance, and using features from one part to predict features in the other. The top resulting SCL dimension for the first part captured the propensity of some words to occur in short utterances, and of particular interest, captured backchannel words from both domains. This led to improvement in detection
of backchannels across languages, although it did no better in this regard than the feature restriction approach or using only a length feature.

Chapter 7 investigated the use of online textual conversations from Wikipedia talk pages for annotating questions and sentence/dialog act boundaries in meetings. We compared the lexical feature model trained on the Wiki domain with lexical and prosodic feature models trained on the meetings domain. For question detection, the lexical Wiki model did worse than the lexical meetings model but better than the meetings model with prosodic features only. For boundary detection, the lexical Wiki model did worse than the meetings models using any feature set. For question detection, the Wiki model did particularly badly for certain question types, in particular declarative questions, for which prosodic features were shown to be helpful using an in-domain model. This motivated our experiments to incorporate prosodic features using domain adaptation on unlabeled speech data. We experimented with self-training (bootstrapping), SCL, and variations on co-training, using a small set of labeled speech data for model selection and tuning. We found that SCL and self-training led to improved performance on question detection, and all three methods led to improved performance on sentence boundary detection. However, use of the labeled development set in training outperformed all adaptation methods.

8.2 Contributions

Contributions toward understanding of domain adaptation in general:

- We presented models of domain relationships to understand the effect of the feature restriction approach, LSA, and SCL for domain adaptation. Simulations further allowed comparison of the methods under different assumptions about domain relationship. Our models and simulations are useful for understanding results here and for knowing what to expect in other cases.

- We showed with a synthetic data example that for ridge regression under covariate shift, instance weighting can be useful. In particular, we showed that while the adaptive weight parameter $\lambda$ from Shimodaira (2000) and ridge penalty $\gamma$ both trade off
bias and variance, they are not interchangeable—better target domain generalization error can be achieved with both adaptive weighting and regularization.

Contributions to domain adaptation in language processing:

• We argued that for document classification with sparse word features, when feature co-occurrence probabilities are low, feature restriction is unlikely to have a major effect.

• We proposed a method for estimating instance weights for documents based on language modeling, setting the adaptive parameter from Shimodaira (2000) based on document length. This method encourages moderate weight values and worked well in an artificial scenario known to suffer from covariate shift.

• However, our experiments with instance weighting on the Sentiment Classification, 20Newsgroups, and cross-corpus prosody classification tasks suggest that it is not a useful domain adaptation strategy in many real language processing tasks for which covariate shift is not a known source of cross-domain degradation.

• We explored the application of SCL to a classification task involving data units at the utterance level rather than documents or words. The short, but variable, utterance length presented a challenge for designing auxiliary prediction tasks, which require non-overlapping pivot and prediction feature sets co-occurring in the same utterance. The top SCL feature learned in our implementation resulted from predictor features with negative prediction weights across many pivots, identifying words that tend to occur alone.

• We showed that SCL can be used to learn new features combining textual features and continuous features, where the latter are present in one domain only.

• Our work illustrates that cross-domain degradation in high-dimensional NLP and spoken language processing tasks is often the result of unseen target domain features.
Domain adaptation methods that can incorporate those features (such as SCL, self-training, or co-training) must exploit correlations between target-domain-only features and shared features. Using more features of different types (such as prosodic, lexical, syntactic, and conversational features) may help introduce such correlations.

Contributions to spoken language processing:

- We showed that prosody classification (binary detection of pitch accents and prosodic breaks) could be performed on a spontaneous speech corpus using a model trained on read news, and vice versa, with relatively small degradations in accuracy compared with an in-domain model trained on an equivalently-sized training set. In addition, our work suggests that a single model can be used to predict prosody classes in different styles of speech with little or no degradation compared with a style-specific model.

- For pitch accent detection, we showed that acoustic features transferred better than textual features across corpora of different styles.

- We showed that utterance length is an important feature for distinguishing backchannels in both English and Spanish conversations.

- We showed that online textual conversations can be used to detect questions and sentence boundaries in spoken conversations, without requiring hand-labeled training data. We also showed that domain adaptation methods can incorporate prosodic features from unlabeled speech data. In our experiments, the domain adaptation methods improved performance over the baseline text-only model when tuned using a labeled development set, but did worse than using the labeled development set directly in training.

8.3 Future Work

For domain adaptation, our work suggests a couple directions for future work. First, we showed that domain adaptation methods depend on assumptions about the relationship between the source and target distributions. These assumptions might be more formalized
in a probabilistic model describing the joint generation of the source and target domains, which might also suggest other approaches for domain adaptation.

Second, our work illustrates some ongoing questions about evaluating unsupervised domain adaptation methods (and more generally, semi-supervised learning methods). In particular, if a method requires parameter tuning or model selection, and that step is performed using labeled target data, it seems reasonable to also consider other uses of the labeled target data, such as using it for training. One could imagine comparing these approaches theoretically, for example, attempting to analyze the model selection step in the same way as the training step (to get an idea of how many samples are needed to make a good selection). In this way it might be possible to formally compare the use of target data for model selection vs. for training, given a certain number of source and target labeled examples.

Third, several questions remain about using SCL for domain adaptation. For one, our work in Chapter 7 combined textual and continuous acoustic features in SCL features learned on both Wiki and MRDA data. To do this, we normalized the continuous features in the MRDA data to have zero mean and variance one, and we added zeros in place of the continuous features to the Wiki data. However, the textual features occur rarely and typically have much smaller weights in the learned weight vectors than the continuous features. Our work has not fully examined the effect of combining these two types of features in SCL; related questions also remain, such as whether or not normalization of the continuous features is important, and how much the proportion of Wiki to MRDA data matters. Another set of questions about SCL relate to selecting the pivot and predictor features. It is clear that the pivots and predictors should be selected in such a way that the learned weight vectors capture information about the classes of interest, as opposed to other underlying structures. Related to this goal, we would like to avoid “trivial” predictions such as a pivot being predicted from itself. Blitzer (2007) avoided using trivial predictors in individual auxiliary tasks, instead setting their weight to 0; but how to define these trivial predictors, particularly in the case of n-gram features, should be explored in more detail.

For spoken language processing, our work on pitch accent detection showed that the acoustic prosodic features transferred well across our corpora from different styles. Future work might investigate how well they transfer across further styles or different languages.
Another direction for future work is to further explore the use of Web-based textual conversations for annotation of speech. In textual conversations, participants use punctuation, capitalization, emoticons, and fonts to communicate structure, focus, emotion, and other aspects which are conveyed through prosody in speech. Therefore, intelligently using textual conversations as training data might reduce the need for hand-annotated speech data. Chapter 7 considered two tasks, question detection and sentence boundary detection. Future work might consider: (i) detecting prominent/emphasized words, based on bold or italicized words in text; (ii) detecting exclamations, based on exclamation marks in text; (iii) classifying emotion, based on emoticons. In addition, it is possible that a method like SCL could be used to discover correspondences between speech features and text features. This would likely require as input a vector of “possibly-relevant” speech features (computed from the F0 contour, for example) on one or more candidate windows associated with the unit level (like word or sentence), since it is unclear how such a method could discover anything useful from a variable-length sequence of sample values. Another challenge is designing appropriate auxiliary tasks such that the learned correspondences would be meaningful.
BIBLIOGRAPHY


J. Gao, W. Fan, J. Jiang, and J. Han. Knowledge transfer via multiple model local structure mapping. In *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2008.


J. He, Y. Liu, and R. Lawrence. Graph-based transfer learning. In *Proc. ACM Conf. on Information and Knowledge Management (CIKM)*, 2009.


D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proc. 13th Int. Conf. on Very Large Data Bases (VLDB)*, 2004.


VITA

Anna Margolis grew up in Pennsylvania, U.S.A., and attended college at University of Chicago, earning a B.A. in Mathematics in 2003. She began studies at University of Washington in 2004, initially focusing on stochastic modeling of computer networks. She earned an M.S. in Electrical Engineering in 2006. She then began studying language processing and machine learning. She completed an internship at MIT Lincoln Laboratory in 2007, and was an intern and visiting student at Toyota Technological Institute at Chicago from 2009-2011. In addition to her dissertation research, she has worked on projects involving supervised and unsupervised detection of prosodic events, topic classification of telephone conversations, and classification of participant roles in broadcast and textual conversations. She earned a Ph.D. in Electrical Engineering from University of Washington in 2011.