

Practical Natural Language Processing for Low-Resource Languages

by

Benjamin Philip King

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2015

Doctoral Committee:

Associate Professor Steven P. Abney, Co-Chair
Professor Dragomir Radkov Radev, Co-Chair
Associate Professor Steven Bird, University of Melbourne
Assistant Professor Michael John Cafarella
Assistant Professor Ezra Russell Keshet
Associate Professor Rada Mihalcea

© Benjamin Philip King 2015

ACKNOWLEDGMENT

I would like to thank my advisors Dragomir Radev and Steven Abney, who pushed me to do the best work that I could and to consider new ideas and to see them through to completion. They helped me to develop as a researcher, a student, and a person. In addition to countless hours, they provided inspiration, encouragement, and guidance. I am also extremely grateful to the members of my thesis committee, Ezra Keshet, Rada Mihalcea, Michael Cafarella, and Steven Bird. They provided criticism, feedback, and encouraged me to consider other perspectives.

I am also most fortunate to have been able to collaborate with many wonderful researchers and academics. First I've been able to work closely with many talented and hard-working students at Michigan: Vahed Qazvinian, Terry Szymanski, Amjad Abu-Jbara, Rahul Jha, Wanchen Lu, Reed Coke, and Cathy Finegan-Dollak. Also through the SCIL and FUSE projects, I've been able to collaborate with leading researchers from other institutions, including Kathy McKeown, Hal Daumé III, Owen Rambow, and Fei Xia, who helped me to expand my understanding of NLP.

Finally, I would like to thank my fiancé Steve Watson and my parents, who have supported and encouraged me through the last five years. Without their help, none of this work would have been possible.

TABLE OF CONTENTS

ACKNOWLEDGMENT	ii
LIST OF FIGURES	viii
LIST OF TABLES	x
ABSTRACT	xiii
Chapter I. Introduction	1
1.1 Low-Resource Languages	1
1.2 Previous approaches to low-resource languages	3
1.3 Our approach to low-resource languages	4
1.4 Outline of this thesis	6
Chapter II. Background and related work	7
2.1 Low-Resource Languages	7
2.2 Language Identification and Multilingual Text	9
2.3 Conditional Random Fields	12
2.3.1 Linear Chain CRFs	13
2.3.2 Tree-Structured CRFs	14
2.4 Weakly-Supervised Learning	15
2.4.1 Generalized Expectation Criteria	16
2.4.2 Posterior Regularization	17

2.4.3	Direct Transfer Learning	18
2.5	Low-Resource POS Tagging	19
2.6	Low-Resource Dependency Parsing	22
2.7	Useful Resources	24
Chapter III. A system for collecting and processing low-resource language text from the Web		28
3.1	Introduction	28
3.2	Design Decisions	30
3.3	Features	30
Chapter IV. Labeling the Languages of Words in Mixed-Language Documents using Weakly Supervised Methods		36
4.1	Introduction	36
4.2	Task Definition	38
4.2.1	Evaluation Data	39
4.2.2	Annotation	40
4.2.3	Human Agreement	42
4.2.4	Training Data	42
4.3	Word-level Language Classification	43
4.3.1	Features	44
4.3.2	Classifiers	44
4.4	Methods	45
4.4.1	Conditional Random Field Model trained with Generalized Expectation	46
4.4.2	Hidden Markov Model trained with Expectation Maximization	48
4.4.3	Logistic Regression trained with Generalized Expectation	49
4.4.4	Word-level Classification	49
4.5	Evaluation and Results	50
4.6	Error Analysis	51
4.7	Conclusion	53

Chapter V. Bayesian Methods for Word-Level Language Identification in Mixed-Language Documents	55
5.1 Introduction	55
5.2 Background	56
5.3 Data	57
5.4 Methods	58
5.4.1 Basic Mixture Model	61
5.4.2 Hierarchical Model	63
5.4.3 HMM-based Model	65
5.5 Evaluation and Results	68
5.5.1 Initialization and Granularity	68
5.5.2 Baselines and Comparisons	69
5.5.3 Measures	69
5.6 Discussion	70
5.7 Conclusion	72
Chapter VI. Sharing Grammars Between Treebanks for Type-Supervised Part-of-Speech Tagging	74
6.1 Introduction	74
6.2 A Motivating Example	79
6.3 Data	80
6.3.1 Treebanks	80
6.3.2 Transferred Grammars	81
6.4 Methods	85
6.4.1 HMM	86
6.4.2 Tag Dictionary Expansion	86
6.4.3 Model Minimization	88
6.4.4 Guided Model Minimization	89
6.4.5 EM Training	89
6.4.6 Seeded EM Training	90

6.5	Results	90
6.6	Discussion and Future Work	92
6.7	Conclusion	93
Chapter VII. Multi-Source, Multi-Target Cross-lingual Part-of-Speech Tag Projection		95
7.1	Introduction	95
7.2	Methods	97
7.2.1	Overview	98
7.2.2	Learning Model	99
7.2.3	Baseline: Pruning the Lattice	101
7.2.4	Features	102
7.3	Data	102
7.3.1	Parallel Corpus	102
7.3.2	Tagging Dictionaries	103
7.3.3	Training and Testing Data	103
7.4	Experiments and Results	104
7.4.1	Low-Resource Experiments	105
7.5	Discussion	107
7.6	Conclusion	110
Chapter VIII. Multi-Source Cross-lingual Dependency Parser Projection		111
8.1	Introduction	111
8.2	Methods	112
8.2.1	Single-source projected training	112
8.2.2	Multi-source projected training	114
8.3	Experiments	115
8.3.1	Data	115
8.3.2	Features	116
8.4	Results and Discussion	116

Chapter IX. Conclusions and Future Directions	119
9.1 Summary of Contributions	119
9.2 Future Directions	121
Bibliography	123

LIST OF FIGURES

3.1.1 Example of a document profile page on the Minority Language Server.	29
4.3.1 Learning curves for logistic regression, naïve Bayes, decision tree, and Winnow2 on the independent word classification problem as the number of sampled words in each training example changes from 10 to 1000.	45
4.5.1 Learning curves for naïve Bayes, logistic regression trained with GE, HMM trained with EM, and CRF trained with GE as the number of sampled words in each training example changes from 10 to 1000.	50
5.3.1 The frequency of documents with different numbers of languages in our corpus.	59
5.4.1 Plate diagrams for the Bayesian language inference models: (5.4.1a) the basic mixture model, (5.4.1b) the hierarchical model, and (5.4.1c) the HMM-based model.	60
6.5.1 Unlabeled attachment scores (UAS) in percentage for a multi-source delexicalized dependency parser trained on the type-supervised predicted tags and the fully-supervised predicted tags for each language.	91

7.1.1	Examples of POS projection from three different source languages (English, Romanian, and Polish, respectively) onto the same Spanish text. Errors in tagging and word alignment, along with syntactic differences between languages mean that no single target language is able to correctly tag the text. Shown below them is the lattice allowed by the Spanish tagging dictionary with the correct path highlighted. By combining information from all these sources, the Spanish text can be correctly tagged.	96
7.2.1	A diagram of information flow in this chapter's system. This diagram shows three source languages and two target languages, but the method is applicable to any number of source and target languages.	97

LIST OF TABLES

4.1	Languages present in the corpus and their number of words before separating out English text.	40
4.2	An example of text from an annotated English-Sotho web page.	41
4.3	Number of total words of training data for each language.	43
4.4	Logistic regression accuracy when trained using varying features.	44
4.5	Types of errors and their proportions among the different methods. NE stands for Named Entity, SW stands for Shared Word, and Other covers all other types of errors.	52
5.6	Examples of mixed language usage in web pages in our corpus.	58
5.7	Parameters common to the models.	59
5.8	Results for the inference methods with various granularity and initialization parameters, along with baselines and comparisons. <i>A</i> stands for accuracy, <i>P</i> stands for minority class precision, <i>R</i> stands for minority class recall, and <i>F</i> stands for minority class F1-score.	67
6.9	The top five most frequent tag bigrams in the Italian, Portuguese, and Bulgarian treebanks after mapping to the universal POS tags.	76

6.10	An example of type-supervised tagging in German. Below each word is listed the set of possible tags. The words “endet” and “Familienfeier” do not appear in the tagging dictionary and therefore can take any possible tag. The tags selected by this chapter’s tagging method are displayed in bold.	76
6.11	POS tagging accuracies for second-order HMM taggers with emission tables learned from the target language and transition tables learned from a different language. Bolded numbers represent the highest accuracy using any source grammar. The underlined number in each column represents the highest accuracy using a grammar from a source language other than the target language.	77
6.12	WALS word order rules for creating WALS language-similarity vectors. . . .	83
6.13	Average tagging accuracies when using different methods for creating shared grammars.	85
6.14	Tagging accuracies of different POS tagging pipelines. HMM is the hidden Markov model from Section 6.4.1; EM is expectation maximization from Section 6.4.5; SEM is seeded EM from Section 6.4.6; TDE is tag dictionary expansion from Section 6.4.2; MM is model minimization from Section 6.4.3; and GMM is guided model minimization from Section 6.4.4. The most accurate model for each column is listed in boldface. A single underline indicates that a value is statistically significantly greater than the corresponding value in the row directly above at $p < 0.05$. A double underline indicates statistical significance at $p < 0.01$	87

7.15	Accuracies of this chapter’s methods on each of the target languages. Bolded items represent the highest achieved accuracy for each language. A * indicates that an entry is statistically significantly better than the single-source single-target entry with $p < 0.01$	104
7.16	Accuracies of this chapter’s methods on the various languages when applied to the low-resource setting using the Bible for parallel text. Bolded items represent the highest achieved accuracy for each language. A * indicates that an entry is statistically significantly better than the single-source single-target entry with $p < 0.01$	106
7.17	Translations of the Bible used for each language. Entries marked with † contain only a new testament translation.	106
7.18	The prevalence of four types of errors in the Europarl experiments. Rates of each type of error are quite similar in the Bible experiments.	109
8.19	UAS for single-source and multi-source projection for dependency parsing on various languages.	117

ABSTRACT

As the Internet and World Wide Web have continued to gain widespread adoption, the linguistic diversity represented has also been growing. While the Web began as an overwhelmingly English phenomenon, it now contains extant text in thousands of languages. Simultaneously the field of Linguistics is facing a crisis of the opposite sort. Languages are becoming extinct faster than ever before and linguists now estimate that the world could lose more than half of its linguistic diversity by the year 2100. This is a special time for Computational Linguistics; this field has unprecedented access to a great number of low-resource languages, readily available to be studied, but needs to act quickly before political, social, and economic pressures cause these languages to disappear from the Web.

Most work in Computational Linguistics and Natural Language Processing (NLP) focuses on English or other languages that have text corpora of hundreds of millions of words. In fact, most NLP tools are trained using machine learning techniques over large annotated corpora, which are not available for most of the world's languages. In this work, we present methods for automatically building NLP tools for low-resource languages with minimal need for human annotation in these languages. We start first with language identification, the problem of recognizing a text's language in the absence of an explicit label. We specifically focus on word-level language identification, an understudied variant that is necessary for processing

Web text and develop highly accurate machine learning methods for this problem. From there we move onto the problems of part-of-speech (POS) tagging and dependency parsing. With both of these problems we extend the current state of the art in projected learning to make use of multiple high-resource source languages instead of just a single language. In both tasks, we are able to improve on the best current methods. All of these tools are practically realized in the “Minority Language Server,” an online tool that brings these techniques together with low-resource language text on the Web. The Minority Language Server, starting with only a few words in a language can automatically find webpages that contain text in that language. The system is then able to run this text automatically through its language identifier and part-of-speech tagger. We hope that this system is able to provide a convincing proof of concept for the automatic collection and processing of low-resource language text from the Web, and one that can hopefully be realized before it is too late.

CHAPTER I

Introduction

1.1 Low-Resource Languages

It is estimated that there are presently between six and seven thousand languages spoken in the world [1–3], but research in Natural Language Processing (NLP) focuses on only a small number of those languages, English, Chinese, French, German, etc. As of February 2013, there are only about 30 languages that have a published dependency treebank [4], and Google Translate, a popular machine translation tool, supports 80 languages. The majority of the world’s languages have no NLP technology at all.

Languages that have received relatively less attention from NLP usually are less popular due to their lack of available resources and are often called *low-resource languages*. Languages that have an abundance of NLP resources and tools usually do so because of social, political, and financial reasons. In fact, of the 50 most spoken languages in the world [5], 36 are supported by Google Translate. Of the remaining languages, eleven are minority languages within the country in which they are primarily spoken, and five are primarily spoken in a country with poor economics and education, whose Human Development Index falls in the lowest quartile [6]. Though focusing money and effort toward the most widely-

spoken languages makes sense from an economic standpoint, it is difficult for researchers to produce significant resources for current low-resource languages without funding. Research into language-independent NLP methods that are appropriate in low-resource settings is desperately needed as such techniques can be applied to many low-resource languages at once.

Adding to the urgency needed for NLP in low-resource languages is the disappearance of endangered languages. The linguist Michael Krauss famously estimated that up to 90% of the world's languages could disappear by the year 2100 [7]. A major factor that Krauss did not include in his analysis is the emergence of the World Wide Web. While it began as an overwhelmingly English phenomenon, as it has matured, it has begun to diversify, now containing text in thousands of languages, though political, social, and economic pressures have even caused some languages to disappear from the Web [8]. The Web could either serve to accelerate the process of language disappearance if speakers of minority and endangered languages find that they have to use a different language to communicate on the Web, or it could help to preserve languages if speakers feel that they are able to communicate on the Web in those languages. The availability of NLP tools, such as machine translation (MT), on the Web could help speakers of low-resource languages to continue to use that language rather than abandon it in favor of a majority language.

An obvious question raised is why the usual techniques used in NLP cannot simply be applied to these low-resource languages? The reason is that NLP, since the 1990s has grown increasingly to depend on statistical methods and machine learning, to the point now where it's rare to find a paper published using any rule-based or other non-statistical method. The power of these statistical methods comes from having large amounts of data from which to learn parameters for machine learning models. In fact in fields such as MT, research

that focuses on finding more training data is regularly presented alongside research on novel methods [9–11]. When these same techniques are applied to low-resource languages, the performance is often poor. The concentration of NLP on majority languages has also led to a phenomenon whereby even the techniques developed are fit too strongly to these major languages to be applicable to languages with different features and norms [12]. For example, one of the most popular methods for syntactically-informed MT [13] relies on constituency parsing, which is difficult to apply to languages with freer word orders, as this results in discontinuous constituents [14].

1.2 Previous approaches to low-resource languages

Previous work on NLP for low-resource languages can put into two major categories: (1) approaches that focus on a small set of languages, and (2) approaches that are applied to a very large set of languages simultaneously.

The first approach generally focuses on a single language or a small set of related languages, starting with a data collection phase to compile text or speech in the language(s) of interest, and usually also producing an NLP tool [15–20]. These approaches do yield useful results, but typically require aid from an expert and are not immediately applicable to other languages.

One of the most notable examples of an approach that works simultaneously on a very large set of languages is Kevin Scannell’s Crúbadán project [21]. By carefully crafting Web search queries designed to return Web-pages in specific low-resource languages, they are able to build corpora for 1872¹ different languages. These corpora have enabled the development of a number of different tools and resources for low-resource languages, including thesauri [22],

¹Retrieved from <http://bore1.slu.edu/crubadan/stadas.html> on February 9, 2014

diacritic restoration [23], and MT [24]. A drawback of Scannell’s approach is its reliance on the manual effort of expert volunteers, without which none of these resources and tools can be created. The data collected in this project also cannot be distributed due to copyright concerns.

Other examples of the many language approach include the proposed Human Language Project [25], which describes a common format for annotated text corpora and issues the challenge of creating a universal corpus containing all the world’s languages, and the Leipzig Corpora Collection [26], which has built corpora for 124 distinct languages and (though the texts again can’t be distributed due to copyright) offers statistics about each of these languages, such as word frequencies and contexts as well as dictionaries [27].

1.3 Our approach to low-resource languages

We also take the approach of processing many languages at once, working on developing techniques that are as language-independent as possible, since language independent techniques can be applied to many languages at once.

We believe that progress in useful low-resource NLP depends on having access to real, representative text in those languages. While resources such as word lists, translation dictionaries, and morphological descriptions are useful resources and can help improve NLP, these is no substitute for a corpus of representative language usage, which demonstrates usages of words in context, allows for building language models, generating fluent text, and can often have a broader coverage even than resources like dictionaries. We use the approach of the Crúbadán project as a starting point and collect text from the Web for a large number of low-resource languages. (See Chapter III for more details) Our system for collecting web text continuously

adds new data by searching the Web.

One benefit of constantly adding new low-resource language text to the system is that the models, tools, and resources for these languages can be continually improved. For example, if we start out only knowing 100 words in Swahili, these words can be used to build a language identifier for Swahili, to search for more documents in Swahili, to identify Swahili words in these documents, and to add these new words to the list of known Swahili words. This process is described in more detail in Chapter III.

Yarowsky et al. [28] introduced a method for transferring annotations from one language to another by projecting them across bilingual word alignments. This is generally done by projecting annotations from an amply-resourced language like English that has many treebanks, billion-word corpora, etc. onto a language that has very few resources in order to learn a tool like a part-of-speech-tagger (POS tagger) or named-entity recognizer (NER). We extend this idea by projecting simultaneously from annotations in many languages. For example, the CoNLL X and CoNLL 2007 shared tasks collected 13 different dependency treebanks [29, 30], each of which can be used to project annotations onto a target language. (See Chapter VII for more detail) We find that errors in the different source-target pairings tend to be mostly independent and that projecting from multiple sources drives the accuracy higher than is possible using any single language. Though most parallel texts are not highly parallel (parallel across many languages), Bible text is especially useful for this technique, as there are translations of the same text in thousands of languages (see Section 2.7).

1.4 Outline of this thesis

Chapter II describes related work and background material that is helpful for understanding this thesis’s contributions. In Chapter III, we describe the Minority Language Server, a system that demonstrates the techniques described in this thesis. Building on previous work on corpus-building, we demonstrate a method that can automatically build a corpus of text for a low-resource language starting with only a handful of words in that language. This method uses a Web search engine to find pages containing some of the few known words. We extend previous work by including a step that identifies the languages of individual words.

Most of the remaining chapters are devoted to describing techniques for low-resource NLP. In Chapters IV and V, we develop the first automatic language identification techniques that are applicable at the level of words. Our methods only require a very small amount of sample text in each language and are effective even when there are many possible languages. Chapter VI transitions to learning of NLP tools via cross-lingual techniques and describes a method for direct transfer of type-supervised POS-taggers with little data that improves tagging performance for low-resource languages. Chapter VII also looks at cross-lingual POS tagging, but makes use of parallel text and improves on the previous state of the art in this task by projecting a tagger onto a target language simultaneously from multiple source languages. Chapter VIII takes a similar approach to the problem of cross-lingual dependency projection, also improving on the previous state of the art with multi-source projection. Finally in Chapter IX, we summarize the contributions of this thesis and lay out avenues for future research.

CHAPTER II

Background and Related Work

This chapter presents background that is necessary to understand the contributions of the thesis as a whole, also describing related work in order to make the relationships with such work clear. We begin with a brief description of issues that often accompany low-resource languages in Section 2.1. In Section 2.2 we describe the task of word-level language identification and survey related work. From there we transition to two sections that provide a technical background: Section 2.3 describing conditional random fields and Section 2.4 describing weakly-supervised learning. In Section 2.5 we survey previous methods for low-resource part-of-speech tagging and do the same for low-resource dependency parsing in Section 2.6. Finally, we describe several useful data resources in Section 2.7.

2.1 Low-Resource Languages

Language documentation and description are the tasks of collecting samples of a language and analyzing the samples in order to describe properties of the language. These two tasks are in the midst of a slow transition from older physical storage methods to electronic digital storage [31]. While this continues to be the domain of field linguists, NLP and Computational

Linguistics are in a great position for potential collaboration on these tasks, though little research into the necessary methods has been conducted so far [25].

NLP methods designed for low-resource languages are likely to encounter many of the same issues experienced by documentary and descriptive linguists working with minority languages. So it is instructive to lay out these issues in order for NLP researchers to learn what to expect when dealing with these types of languages. First one of the biggest problems with low-resource languages is that resources are difficult to obtain. Much of the language description that exists is either unpublished or exists only in paper format. What does exist in electronic format can often be in unusual or other unusable formats. As a consequence, even raw text in an under-resourced language can be difficult to obtain and use [31]. (This work takes advantage of text in low-resource languages that is already available in a usable electronic format on the Web.)

Second, orthographies for low-resource languages may not be standardized. Word boundaries may not be standardized, spellings can vary, and even the usage of the language itself may not be consistent from speaker to speaker [32]. While many languages do have some sort of dictionary or word list, many of these resources were created by foreign linguists and don't necessarily represent a standardized spelling.

A third problem is that while dialects and their relationships to one another are well-understood for the world's most prominent languages, they are often quite unclear when dealing with less studied languages. While mutual intelligibility is often used as a measuring stick for determining the relationship of two languages (whether separate languages or merely separate dialects), even this measure can be somewhat subjective [33]. As a result, it is not uncommon to find that two different sources of text presented as belonging to the same language can exhibit substantial differences, leading to questions for NLP researchers

of whether their tools are even applicable to the presented texts.

While text-based methods certainly have merit given the amount of written text available even in low-resource languages, it is still worth considering that the majority of the world’s languages are unwritten. Many languages that do have some sort of writing system may primarily see linguists making use of that system, while the literacy rates of native speakers remains low [32]. Thus any text-based approach will be fundamentally limited in its scope.

2.2 Language Identification and Multilingual Text

Language identification is one of the older NLP problems [34], especially in regards to spoken language [35], and has received a fair share of attention through the years [36]. In its standard formulation, language identification assumes monolingual documents and attempts to classify each document according to its language from some closed set of known languages.

Many approaches have been proposed, such as Markov models [37], Monte Carlo methods [38], and more recently support vector machines with string kernels, but nearly all approaches use the n -gram features first suggested by [39]. Performance of language identification is generally very high with large documents, usually in excess of 99% accuracy, but Xia et al. [40] mention that current methods still can perform quite poorly when the class of potential languages is very large or the texts to be classified are very short. This is also confirmed by Lui and Baldwin, who find that the best accuracy currently achievable on a corpus of microblog data from Twitter is less than 87% [41].

In this thesis, we explore methods for performing language identification on documents that contain more than one language. Specifically, we consider the problem of word-level language

identification, where a language label must be assigned to each word in the text. In addition to word-level language identification, there are other useful language identification tasks to consider regarding multilingual documents, including detecting a set of languages present in a document [42], detecting the proportions of languages [43, 44], and segmentation by language [45]. However, since all three of these representations can be easily derived from labeled words (though perhaps with slightly lower accuracy than a method that directly produces the other representation), we choose to focus on language identification at the word level.

Perhaps the earliest approach to multilingual language identification in text was Prager’s Linguini system [43]. It not only could detect the language of monolingual documents, but could also detect when a document was bilingual and could predict the proportions of the languages present. Linguini worked by constructing feature vectors for documents and comparing them against feature vectors learned for each language. For bilingual documents, it could compare the document vector against linear combinations of two language profile vectors. Linguini was also one of the first language identification systems to be tested on Web text.

A more recent and comprehensive approach at detecting language proportions was undertaken by Lui and Baldwin [44]. Representing documents as mixtures of n-grams, they applied a mixture model based on Latent Dirichet Allocation (LDA) to infer the proportions of different languages in a document. Their method is quite similar to a method we describe in Chapter V, except that we instead represent documents as mixtures of words and are able to use the mixture model to infer the languages of the words directly. In addition to inferring language proportions, Lui and Baldwin also present a thresholding method that can be used to produce a set of languages present in the document.

Yamaguchi and Tanaka-Ishii presented a method for dividing a multilingual document into contiguous monolingual segments [45]. They describe a dynamic program that attempts to find a segmentation of the text with a minimum description length. The description length of the text is based on a balance between having many segments and having segments that are considered improbable by a language model, both of which increase the description length of the segment.

Following the publication of the work in Chapter IV, there have been a few other papers that look at written language identification both at the word level and in multilingual documents. Nguyen and Dogruöz [46] experiment with various methods for word-level language identification in webpages containing Turkish and Dutch. There has also been a number of papers considering this problem in domains with very short texts, such as multilingual queries in the information retrieval community [47] and code-switched posts in microblog texts [48, 49].

Another related problem that has received attention in NLP is the study of *code-switching* within NLP literature. Most of the work done has been on automatically identifying code-switch points [50, 51]. Initially, the problem of identifying language in the presence of code-switching saw the most attention in the realm of speech processing [52, 53], but recently it has garnered enough attention to merit a workshop devoted specifically to code-switching in text [49].

Though code-switching has been well-studied linguistically, it is only one possible reason to explain why a document contains multiple languages, and we find it to not even be the most common reason for multiple languages to be present in the same document. This leads us to consider the problem more generally, not assuming any specific generating process for documents.

2.3 Conditional Random Fields

Many tasks in NLP involve performing some type of *structured prediction*, meaning that the output of the machine learning algorithm is a structured object, such as a sequence or a tree. A popular method for performing certain types of structured predictions is *conditional random fields* (CRFs). CRFs are discriminative models, which means that the probability $p(y|x)$ of the output y given the input x is modeled directly instead of modeling the joint probability $p(x, y)$, as in a generative model [54]. Discriminative models can typically achieve asymptotically higher levels of accuracy than generative models [55].

CRFs are in the family of log-linear models, meaning that the logarithm of the model's probability is a linear combination of weighted features:

$$p(\mathbf{y}|\mathbf{x}) \propto \exp(\theta \cdot f(\mathbf{x}, \mathbf{y})), \quad (2.3.1)$$

where θ is a parameter vector containing weights corresponding to each of the features and $f(\mathbf{x}, \mathbf{y})$ is a feature function that extracts features from a structured input \mathbf{x} and structured output \mathbf{y} . CRFs are globally normalized, which means that the exponential term in equation 2.3.1 is normalized by the sum of weights for all possible structured outputs $\mathbf{y}' \in \mathcal{Y}$:

$$p(\mathbf{y}|\mathbf{x}) = \frac{\exp(\theta \cdot f(\mathbf{x}, \mathbf{y}))}{\sum_{\mathbf{y}' \in \mathcal{Y}} \exp(\theta \cdot f(\mathbf{x}, \mathbf{y}'))}. \quad (2.3.2)$$

Training and Inference

CRFs are typically trained using a corpus of labeled sequences (X, \hat{Y}) in a supervised fashion by seeking to maximize the log likelihood of the corpus given the gold labeling, leading to

the following optimization problem:

$$\arg \max_{\theta} \sum_{(\mathbf{x}, \hat{\mathbf{y}}) \in (\mathbf{X}, \hat{\mathbf{Y}})} \log p_{\theta}(\hat{\mathbf{y}}|\mathbf{x}). \quad (2.3.3)$$

After a training phase, the parameter vector θ is typically clamped and used to find the best labeling for unlabeled input sentence:

$$\arg \max_{\mathbf{y}} p_{\theta}(\mathbf{y}|\mathbf{x}) \quad (2.3.4)$$

As explained in the following sections, under most configurations, both training and inference are intractable. The two following sections describe two common configurations for CRFs and how they address the intractability problem: linear chain CRFs for sequence labeling and tree-structured CRFs for dependency parsing.

2.3.1 Linear Chain CRFs

The most basic use of CRFs for structured learning is in sequence labeling. In the sequence labeling problem, the input is a sequence \mathbf{x} and the output is a sequence \mathbf{y} of labels where each label y_i corresponds to the input x_i and is drawn from a fixed set of labels \mathcal{L} .

Computing the normalization term (known as the partition function) in Equation 2.3.2 tends to be the most computationally burdensome part of both training and inference. In fact, as the equation is written, it is nearly impossible to do for all but the shortest sequences, since the number of possible outputs grows exponentially with the length of the sequence. In order to make this calculation more tractable, we require that \mathbf{f} factorizes over items in the sequence as follows:

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{|\mathbf{x}|} \mathbf{f}(\mathbf{x}, y_i, y_{i-1}). \quad (2.3.5)$$

In conjunction with the forward-backward and Viterbi algorithms, this allows both training and inference to be performed tractably (both space and time complexity grow quadratically with the length of the sequence).

2.3.2 Tree-Structured CRFs

In addition to predicting a label for each item in a sequence, CRFs can also be used to predict a tree structure for a sequence. These tree structures commonly take one of two different forms: (1) a constituent tree or (2) a dependency tree. We will focus here on dependency trees, which have the following structure. Each item x_i in the sequence \mathbf{x} must be directionally linked to a *governor*, which is either another item in the sequence or `ROOT`, a special symbol that acts as a governor for the topmost item in the tree. A dependency tree must be connected, must have exactly one item governed by `ROOT`, and must not have any simple undirected cycles. The set \mathcal{Y} of possible outputs here is the set of all trees that satisfy these rules.

As with sequence labeling, there are exponentially many possible parse trees, making both inference and training intractable under this basic formulation. To address this, we assume that the total tree probability factors into more manageable chunks, in the case of this thesis, edges.² Practically, this means that the tree’s feature vector can be computed by summing the feature vectors for each of the edges e :

²though multiple other factorizations have been explored in order to build higher-order parsers [56–58]

$$f(\mathbf{y}, \mathbf{x}) = \sum_{e \in \mathbf{y}} f(e, \mathbf{y}, \mathbf{x}). \quad (2.3.6)$$

For tree-structured CRFs, there is an analog of the forward-backward algorithm, called the inside-outside algorithm, and an analog of the Viterbi algorithm, called Eisner’s algorithm [59]. Both of these algorithms have a cubic space and time complexity when the probability factorizes over edges.

2.4 Weakly-Supervised Learning

Machine Learning has historically considered three major learning paradigms: supervised learning, semi-supervised learning, and unsupervised learning. In recent years, a fourth paradigm has begun to receive significant attention. Weakly-supervised learning is a setting in which some training data (usually not very much) is available, but the training data is of a different type than the test data. For example, in the task of POS tagging, the training data may contain tagged word types, while the test data contains sequences to label. This should be considered to be distinct from domain adaptation, where the training and test data are of the same type, but drawn from different populations. Because low-resource languages lack annotated resources appropriate for directly training NLP, many of the tasks we consider in this thesis can be considered weakly-supervised, making use of other types of data.

A number of methods have been proposed in recent years to apply to the problem of weakly-supervised learning. Excluding self- and co-training methods, these methods can be categorized into two broad classes: those which bootstrap from a small number of tokens (sometimes called prototypes) [60, 61], and those which impose constraints on the under-

lying unsupervised learning problem [62–65]. Constraint-based weakly supervised learning has been applied to some sequence labeling problems, through such methods as contrastive estimation [66], generalized expectation criteria [67], alternating projections [68], and posterior regularization [65]. Below we describe generalized expectation criteria and posterior regularization, two frameworks for performing weakly supervised learning.

2.4.1 Generalized Expectation Criteria

The generalized expectation (GE) criteria framework [64] can be used to incorporate additional preferences about the learned model into the optimization problem. GE criteria are terms added to the objective function which specify the expected behavior of the model for certain input features. When the model is a linear chain CRF, we can straightforwardly express these criteria in the objective function with a KL-divergence term between the expected values of the current model \tilde{p} and the preferred model \hat{p} [67].

$$\mathcal{O}(\theta) = \sum_d \log p_\theta(y^{(d)}|x^{(d)}) - \frac{\sum_k \theta_k}{2\sigma^2} - \lambda \cdot KL(\hat{p}||\tilde{p}_\theta)$$

Sometimes the l_2 norm is used as an alternative loss function for \hat{p} and \tilde{p} . In either case, it is straightforward to apply standard gradient methods for optimization by taking the derivative. GE criteria have an advantage over some other methods of weakly-supervised learning in that they are easily interpretable, with a user able to specify these preferences in terms of input features and output labels, rather than having to alter model parameters directly [69].

2.4.2 Posterior Regularization

Posterior Regularization (PR) is another weakly supervised learning framework that is similar in some ways to GE. Whereas GE simply attempts to minimize the distance between the model’s posterior distribution and the preferred distribution, PR actually constrains the space of possible posterior distributions. These constraints take the form of expectations over feature functions $\phi(\mathbf{X}, \mathbf{Y})$ that are directly computable from the posterior labeling. Because the constraints are in expectation only, PR can be more flexible than using hard constraints, which can also be easily cause optimization to become intractable.

Importantly these feature functions need not have any relation to the model features. They can represent desired behavior that would be totally intractable to include in the model directly (even global properties such as, in POS tagging, the proportion of open- and closed-class words). Feature functions, however, in structured learning must factorize exactly as the model features do. We use \mathcal{Q} to represent the set of labellings $q(\mathbf{Y})$ that are compatible with the constraints:

$$\mathcal{Q} = \{q(\mathbf{Y}) : \mathbf{E}_q[\phi(\mathbf{X}, \mathbf{Y})] < \mathbf{b}\}. \quad (2.4.1)$$

The objective function for a CRF under PR is

$$\min_{\theta} \sum_d \log p_{\theta}(y^{(d)}|x^{(d)}) + \frac{\sum_k \theta_k}{2\sigma^2} + \mathbf{KL}(\mathcal{Q}||p_{\theta}(\mathbf{Y}|\mathbf{X})), \quad (2.4.2)$$

where $\mathbf{KL}(\mathcal{Q}||p) = \min_{q \in \mathcal{Q}} \mathbf{KL}(q||p)$. By introducing Lagrange multipliers, the objective function can be expressed in a form that is amenable to standard gradient methods.

In general, GE and PR are quite similar methods that are both appropriate in many of the

same situations. In fact, it has been shown that the PR objective function can be derived from simply taking a variational approximation of the GE objective function. The choice between GE and PR is often simply a pragmatic one, depending on the problem setup and definition.

2.4.3 Direct Transfer Learning

A final type of learning that could be categorized as weakly-supervised is *direct transfer learning*. Previous studies of multilingual learning for NLP have found that when annotations in different languages can be mapped to a common representation, resource-rich languages can be used to build effective NLP tools for low-resource languages. Direct transfer learning works by training a model in a high-resource source language (whose resources have been converted to this common representation) and applying the model directly to text in the target language that has also been converted to use the common representation. This technique has been demonstrated for tasks such as dependency parsing [70] and named entity recognition [71].

This idea has also seen some application to POS tagging. Feldman, Hana, and Brew [72, 73] described a method for creating taggers in a resource-poor language by combining a POS tagger and morphological analyzer in a closely-related source language to produce a POS tagger and morphological analyzer in the low-resource target language. Their method makes use of morphological commonalities between the languages and is unfortunately only applicable for closely related languages. This idea has also been applied to other pairs of closely related languages [74].

2.5 Low-Resource POS Tagging

The task of POS tagging has been considered in many different formulations, including many interesting formulations that are applicable to low-resource settings. The most basic formulation, fully supervised POS tagging has often acted as a benchmark for sequence labeling methods [75, 76]. Little work, however, has been done recently on fully supervised POS tagging, with English accuracies above 97% [77]. Other formulations assume access to less information than the fully supervised case. Broadly these can be categorized by the resources that they make use of: tagging dictionaries, parallel text, or limited annotations.

Tagging Dictionaries POS tagging with access only to a tagging dictionary and raw text (often called type-supervised POS tagging) was originally introduced in [78]. The method described, which continues to be one of the most popular methods for this task is to learn a hidden Markov models (HMM) tagger using expectation maximization (EM) [78]. In this scenario, the initial HMM has an emission table produced from a tag dictionary, with each word having a uniform distribution over each possible tag listed for that word. The HMM’s initial transition table is generally completely uniform over all possible bigrams. EM training then proceeds until convergence. This method is well-known to achieve poor results as compared to fully-supervised HMMs due to EM’s behavior in non-convex search spaces.

Further work on type-supervised POS tagging has largely focused on techniques to handle noisy or incomplete tag dictionaries [79–81], or addressed the weaknesses inherent in EM training of HMMs [82, 83]. One notable weakness is the tendency of EM+HMM to distribute probability mass uniformly among the various entries in the transition and emission tables. Ravi and Knight address this by focusing on the size of the model’s description, finding

the smallest model that is able to explain the treebank given the tag dictionary [84, 85]. Empirically this method produces taggers that are more accurate than those produced by ordinary EM training and that better align with human intuition. Other recent directions have focused on good initialization points for unsupervised learning [86] and converting crowd-sourced dictionaries to use a compatible tag set [87].

Parallel Text Another resource that can be useful for POS tagging in low-resource languages is parallel text. The general technique of combining an NLP tool in a high-resource language with parallel text to produce a tool in a low-resource language is called projection [28]. While early projection attempts were able to produce taggers with fairly high accuracy, the projection required some heuristics and didn't generalize well when tested on other languages [88].

With the development of the Universal POS Tagset [89], it became possible to develop POS tag projection systems that could be directly applied to many languages. Das and Petrov [88] established a new state of the art in cross lingual tagging when they presented their model using projected tags with a graph-based label propagation step to regularize the projected labels. The first model to make use of both projected labels and a tagging dictionary was the partially observed CRF model of Täckstöm et al. [90], which used both sources of information to prune the CRF lattice before training (see Section 7.2.3 for more details). This work was later reformulated within the Posterior Regularization framework [91], which allowed the hard projection constraints to be treated as soft constraints and to be ignored when advantageous by the model, leading to improved performance.

A drawback of most projection systems described is that they project from only a single language. In practice, this can lead to a number of different issues. First, typological

differences between the source language and the target language, such as a concept being expressed verbally in one language and nominally in another, can lead to annotations that are correct in the source language being incorrectly applied to the target language. Imperfect correspondence of words in the source language to words in the target language can also lead to incorrect annotations being projected, if for example, a source language word aligns to two target language words that should receive different tags. Finally, errors in alignment or source-language tagging also often result in incorrect taggings.

Using more than one source language could help with all of these problems, representing a greater variety of linguistic phenomena and increasing the chance that at least one source language mirrors the behavior of the target language. More source languages also increases redundancy, which helps to eliminate mistakes caused by errors in source language tagging or alignment. While this thesis is not the first work to suggest projection from multiple source languages, previous approaches used a simple majority vote, leading to results that are no longer competitive with the state of the art in single language projection [92]. How multiple sources can be integrated into modern projection frameworks is not as clear and is the focus of Chapter VII of this thesis.

Limited Annotations Garrette et al. [93,94] explored how the effort of human annotators should best be spent, whether on creating tag dictionaries or annotating text, and how much time should be spent on each. They also develop methods for building automatic POS taggers from tag dictionaries created in this way. Further methods in this vein were developed by Duong et al. [95] to make use of 1000 annotated tokens in the low-resource language.

Finally at the extreme of low-resource approaches is unsupervised POS tagging, also known as POS induction, which does not allow access to any type or token annotations. Among

unsupervised NLP tasks, unsupervised POS tagging has received perhaps the most attention [82, 96, 97]. The main difficulty with this task has been evaluation as it is difficult to meaningfully map between induced word clusters and true POS tag classes [98].

2.6 Low-Resource Dependency Parsing

As with POS tagging, the most studied formulation of dependency parsing is fully supervised dependency parsing. Though recent years have seen dependency treebanks published for a number of languages, the creation of a dependency treebank remains a major undertaking, even using partially automated techniques [99]. Though supervised dependency parsing remains far from solved, the problem of building a dependency parser without a large treebank is a problem that has been considered by a number of NLP researchers.

Unsupervised Parsing The most general approaches use fully unsupervised parsing, in which generative models learn grammar from plain, unannotated text (though in practice, most systems induce grammars from part-of-speech (POS) annotations rather than plain text). Unsupervised parsing has not yet achieved accuracy comparable to supervised systems, keeping such parsers from finding extensive use in downstream applications [70].

Delexicalized Transfer Parsing A second major approach to parsing in a language with no treebank is known as *delexicalized transfer parsing*, a type of direct transfer learning. In delexicalized transfer parsing, we assume access to a source language that both has a treebank and uses the same POS tagset as the target language. On the source side, a dependency parsing model is trained on only the POS tags and the actual words are ignored. The target

language text is POS tagged and the trained delexicalized model is run directly on these tags to parse the target language text. Delexicalized models have accuracies much closer to supervised accuracy than unsupervised methods have.

Delexicalized transfer of parsers has been demonstrated on many different languages [100–102], but it has been most effective when multiple source languages are used [70,103]. McDonald et al. [70] showed that simply concatenating delexicalized treebanks from many languages, it was possible to get higher target language accuracy than with a single source language. Since delexicalized transfer parsing has been so successful, many different approaches have been proposed for a two stage process that starts with delexicalized transfer parsing and then relexicalizes the target language parser for even higher accuracy [70,103,104].

Projected Parsing A third approach is called cross lingual projected learning and makes use of parallel text in which the source language has a treebank. This method relies on the direct correspondence assumption, which states that after word-aligning the text, words that are aligned to each other are each governed by a word that aligns to the other. That is to say, that syntactic relations are preserved through projection across word alignments. In projected parsing, the source side of the parallel text is parsed and the parse trees are projected across alignments. The parsed sentences on the target side are used to train a dependency parser. The best projected parsers perform even better than the best delexicalized parsers and continue to close the gap with supervised learning.

Projected parsing was first proposed by Hwa et al. [105], who found that much language-specific tuning was necessary to coax reasonable performance from the projected parsers. Since that time, research on projected parsing has followed two different tracks, one that used projections as constraints for unsupervised grammar induction methods [106–109], and

another that trained target language parsers using semi- and weakly-supervised learning techniques [110–112]. While no clear winner has emerged from these techniques, our own work uses the model of Ma and Xia [112], which demonstrated superior accuracy to many of the alternative models, as a starting point.

One other line of work that does not fit neatly into either of the previously mentioned tracks is work on parser projection using interlinear gloss text (IGT) [113–115]. IGT is a resource often created by linguists to enable easy understanding of morphemes, cases, and word order. IGT takes the form of a foreign language sentence and an English sentence³ with manually created word alignments between them. It also adds a gloss line that usually explains which morphemes in the foreign language correspond to which English words and often also gives notes about cases, numbers, genders, etc. Work in this area is notable for several reasons. First, it achieves higher levels of accuracy than with the word alignments alone. Second, it uses methods that are much more linguistically attuned than the methods of the previous paragraph. Third, these methods are examples of the type mentioned earlier that would be needed to help bridge the gaps with traditional documentary and descriptive linguists. Such linguists are well-versed in creating IGT and there exists an abundance of IGT for many languages due to their efforts. A large collection of IGT can be found in the Online Database of INterlinear text (ODIN) [116].

2.7 Useful Resources

Throughout this thesis, we make use of data resources that have been published by other groups and researchers before us. Below are descriptions of five data sets that find use in

³This is most often English, but can be any language in which the linguist is fluent.

multiple chapters of this thesis.

Universal Declaration of Human Rights The Universal Declaration of Human Rights (UDHR) is a document created by the United Nations in 1948 in response to actions taken the second World War [117]. Since its initial publication, it has been translated into many languages. As of May 2015 there were 444 versions available from the UN’s website⁴. The text of the UDHR is useful for training language classifiers. Because its translation is carefully controlled, it is known to be representative text in its respective language and free of borrowings, errors, and other problems that frequently plague other sources of language samples.

CoNLL (and Other) Treebanks The Conference on Natural Language Learning (CoNLL) sponsors a shared task. In 2006 and 2007, this shared task focused on dependency parsing in many languages. The organizers collected 13 dependency treebanks in different languages [29, 30], converting them to a common format and making them easy to obtain for participants. These treebanks can be used to train both supervised POS taggers and supervised dependency parsers.

While these treebanks were appropriate for supervised tasks in a single language, a problem common to cross-lingual transfer of syntax is that different treebanks tend to use different conventions. For example, the Prague Dependency Treebank, by convention, labels the coordinating conjunction as the head of a coordinated noun phrase, while in the Penn Treebank, the head of the first noun phrase is the head of the entire phrase. To help address these issues, a group headed by Google researchers has produced a conversions of these treebanks

⁴<http://www.ohchr.org/EN/UDHR/Pages/SearchByLang.aspx>

using a universal syntactic scheme. These are published with an unrestrictive license as the Google Universal Treebanks v2.0 collection [118]. This effort has since been subsumed by the Universal Dependencies project⁵, which has opened this project up to a larger group.

Europarl Europarl is a corpus containing the proceedings of the European Parliament translated into 21 European languages [119]. The proceedings are generally conducted in English and translated by professional translators into the other languages. The proceedings stretch from 1996 until 2011 and comprise over 55 million words of text. Because of its size and quality, Europarl is a popular corpus for training statistical machine translation systems. While not all language pairs are published with sentence alignments already computed, a tool is included for aligning sentences and standard unsupervised word alignment systems can be used to align texts at the word level. We use this corpus as a parallel corpus for projecting NLP tools.

Bible Translations A useful source of text in many languages is the text of the Bible. Though some researchers have made use of this text for NLP [17, 28, 120–122], it has largely been ignored as a resource for doing NLP across a large set of low-resource languages. Not only does a complete Bible represent a large text collection for a low-resource language (generally about 30,000 sentences and 600,000 words), the text is often easy to obtain. Bible translations can be easily harvested from the Web for over 800 languages [123].

Multiple translations in a language are sometime available (most often for widely spoken languages). Translations can differ in how literal they are. In the Bible translation community, translations are considered in a spectrum spanning formal equivalence, often called

⁵<http://universaldependencies.github.io/docs/>

word-for-word translation, producing very literal translations, and dynamic equivalence, often called sense-for-sense translation, producing translations that try to communicate the meaning of the original at the expense of literality [124]. After selecting two translations in different languages, the text can be aligned across languages to create pairs of parallel sentences, which can be used for many purposes, such as MT and projecting annotations to a new language. Since dynamic translations can vary greatly in their sentence structure, when multiple translations are available, we try to use formally equivalent translations, which tend to result in better alignments.

Panlex Panlex is a project that compiles information from many different dictionaries and lexicons, including for many low-resource languages, into a single resource [125]. Because it combines information from many different sources, Panlex has a complex schema, relating word types, definitions, parts-of-speech and other data in multiple ways. Using Panlex, it's possible to build translation dictionaries and tagging dictionaries for over 500 languages.

CHAPTER III

A system for collecting and processing low-resource language text from the Web

3.1 Introduction

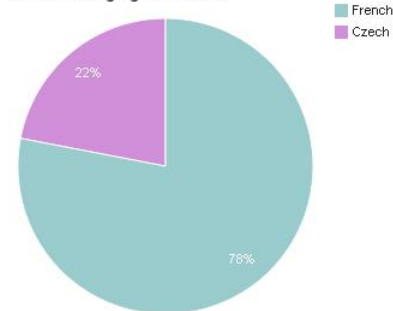
The Minority Language Server is a Web-based platform that implements features of this planned thesis, demonstrates its techniques on real-world data from the Web, and provides resources for low-resource languages. It collects text by performing targeted crawls of the Web, focusing on pages that contain a specific low-resource language.

Each language in the system has a profile page in the system's web interface that lists high-confidence word types in the language and links to documents that contain that language. Each document and each word type also has its own page. The type's page lists its language, the other languages that contain that type, links to documents containing that type, and examples of the type's usage. The document's page displays the web page's text and allows users to view annotations of the text. Types of annotations include language identification and part-of-speech.

Document Statistics

Language	Proportion
French	78.02%
Czech	21.98%

Document language distribution



Document Text

La lecture en tchèque Si vous essayez de pratiquer votre lecture en tchèque, alors cette page devrait vous aider. Vous pourrez pratiquer votre compréhension avec le texte ci-dessous, qui fait partie de l'article 26 et 27 de la déclaration universelle des droits de l'homme. La première section est en tchèque et la deuxième partie est en français. Après avoir terminé avec cette page, veuillez consulter notre page principale Apprendre le tchèque pour plus de leçons. La lecture en tchèque Apprendre la lecture en tchèque est très important parce que sa structure est utilisée dans toutes les conversations quotidiennes. La seule solution est de maîtriser la grammaire et le vocabulaire pour être en mesure de parler la langue couramment. Mais nous devons d'abord savoir le rôle que la lecture en tchèque jouent dans la langue. La lecture en tchèque Článek 26 Každý má právo na vzdělání. Vzdělání nechtí je bezplatné, alespoň v počátečních a základních stupních. Základní vzdělání je povinné. Technické a odborné vzdělání budíž všeobecně přístupné a rovněž vyšší vzdělání má být stejně přístupné všem podle schopností. Vzdělání má směřovat k plnému rozvoji lidské osobnosti a k posílení úcty k lidským právům a základním svobodám. Má napomáhat k vzájemnému porozumění, snášenlivosti a přátelství mezi všemi národy a všemi skupinami rasovými i náboženskými, jakož i k rozvoji činnosti Spojených národů pro zachování míru. Rodiče mají přednostní právo volit druh vzdělání pro své děti. Článek 27 Každý má právo svobodně se účastnit kulturního života společnosti, užívat plodů umění a podílet se na vědeckém pokroku a jeho výtěžcích. Každý má právo na ochranu morálních a materiálních zájmů, které vyplývají z jeho vědecké, literární nebo umělecké tvorby. Après avoir examiné le tableau ci-dessus sur la lecture en tchèque, essayez de créer d'autres exemples utilisant les mêmes mots que vous avez appris de cette liste. Il est toujours utile de savoir comment jouer avec ces mots dans une phrase. La lecture en français Voici la traduction du texte ci-dessus, vérifiez que vous avez compris sans l'aide du dictionnaire. Après avoir lu la traduction française. Essayez de revenir au texte ci-dessus et vérifiez si vous pouvez comprendre autres mots de plus. La lecture en français Article 26 1) Toute personne a droit à l'éducation. L'éducation doit être gratuite, au moins en ce qui concerne l'enseignement élémentaire et fondamental. L'enseignement élémentaire est obligatoire. L'enseignement technique et professionnel doit être généralisé; l'accès aux études supérieures doit être ouvert en pleine égalité à tous en fonction de leur mérite. 2) L'éducation doit viser au plein épanouissement de la personnalité humaine et au renforcement du respect des droits de l'homme et des libertés fondamentales. Elle doit favoriser la compréhension, la tolérance et l'amitié entre toutes les nations et tous les groupes raciaux ou religieux, ainsi que le développement des activités des Nations Unies pour le maintien de la paix. 3) Les parents ont, par priorité, le droit de choisir le genre d'éducation à donner à leurs enfants. Article 27 1) Toute personne a le droit de prendre part librement à la vie culturelle de la communauté, de jouir des arts et de participer au progrès scientifique et aux bienfaits qui en résultent 2) Chacun a droit à la protection des intérêts moraux et matériels découlant de toute production scientifique, littéraire ou artistique dont il est l'auteur. Nous espérons que vous avez profité de cette leçon sur la lecture en tchèque y compris la déclaration universelle des droits de l'homme. Après avoir terminé de cette page, veuillez consulter notre page principale Apprendre le tchèque pour plus de la grammaire et le vocabulaire. N'oubliez pas d'ajouter cette page aux

Figure 3.1.1: Example of a document profile page on the Minority Language Server.

3.2 Design Decisions

The current version of the Minority Language Server is designed to be useful for linguists and natural language processing researchers who are interested in low-resource languages. As new features are added, we would like to also make the site useful for speakers of low-resource languages as well, with tools, resources, and links for them. This section describes design decisions that were made.

Always collecting data With thousands of languages and billions of pages on the Web, there should never be a shortage of material for the Minority Language Server to explore. The system has the ability to continuously collect new data both by querying a search engine and by crawling domains. This requires, among other things, having a schedule to determine how much and how often different languages should be explored.

Capacity for human intervention Of course, continuously running a system like this means that mistakes will be made, and may even begin to propagate. For this reason, the system logs every action that it takes and any of these actions can be rolled back and undone.

3.3 Features

Bootstrap Web Crawling One of the major challenges in processing low-resource languages is collecting text to build a corpus in these languages. The World Wide Web has been used extensively in the past for corpus building.

The Minority Language Server’s bootstrap Web crawling method is based on the method used in Kevin Scannell’s Crúbadán Web crawler [21].

Algorithm 1 Algorithm used for bootstrap Web crawling.

```
Sample three high-confidence tokens from  $L$ 
Submit tokens to search engine
for each page returned do
    Find the document's main text
    Label the language of each word in the main text
    Update the confidences for each word-language pair
end for
Return to the start
```

The entire process is iterated. Starting from a small set of high confidence words in a language L , this algorithm attempts to find pages containing additional text in L . As a new word continues to be labeled as belonging to L , the confidence of that word belonging to L increases and this new word eventually becomes part of the set of high-confidence words in L . In addition to retrieving URLs from a search engine, the crawler has the ability to retrieve all the pages from a single domain, since finding a language on one page in a site indicates that there may be more of that language elsewhere on the site.

Because of the nature of Web pages and the need to navigate from one page to another, pages often contain “boilerplate” text, such as menus, navigational links, and copyright information that does not contribute to the page’s message. To remove this extraneous text, we use the heuristic method found in the BootCaT toolkit [126], which attempts to find a stretch of text that maximizes the ratio between the number of text words and the number of HTML tags. This method, despite its simplicity is often quite successful, though it does occasionally produce false negatives (e.g. removing the title or byline) and false positives (e.g. failing to remove boilerplate text that occurs between posts in a discussion forum).

Whereas Scannell used rule-based filtering and manual intervention to isolate text in the target language, we instead incorporate word-level language identification. Our experience

with low-resource language text on the Web has shown that, more often than not, Web pages containing text in a low-resource language also contain some text in another language. This allows for our method to find not only text in the target language and improve its models and resources for the target language, but also text in the other languages and improve performance in those languages at the same time.

Word-Level Language Identification For each document that is returned by the search engine, we pass it through a word-level language identifier to label the words according to the language to which they belong. We use the Bayesian language identification model described in Section 5.4.3, a model based on a hidden Markov model that is both fast to run and highly accurate. We use a sentence-level granularity and document-level initialization, which results in the highest accuracy among the methods tested, at approximately 93% accuracy. Though the method does make mistakes, these mistakes are kept from propagating by the confidence estimation techniques discussed below.

Continuous Learning with High-Confidence Knowledge Self-training is a method for learning from unlabeled data using automatically produced labels in order to improve the accuracy of a statistical classifier. Typically a classifier is run over an unlabeled corpus to produce automatic labels. Then all or some of these labels are used to retrain the classifier, which hopefully shows improvement after retraining. In many cases, self-training has shown little benefit [127, 128]. It has however shown promise in two areas of application: domain adaptation [129], and in cases where data is continuously added to the system [130]. Fortunately, these are both aspects that apply to the problem of the Minority Language Server.

Many iterative relation-learning systems have performed self-training by assigning confidence scores to relation candidates and promoting high-confidence relations to be used as training data for future iterations [130–132]. Throughout this section, we will use language identification as a running example, though the technique applies equally well to word-POS-tag confidences, translation pair confidences, etc. In this case, rather than learning arbitrary typed relations from text, we are learning “word-*belongs-to-language*” relations from the output of word-level language identification.

Our paradigm for updating global confidences is to process a single document at a time and then update confidences for all the words that appear in the document. Each document is given an equal weight when computing updates. Our scheme for updating confidences is Bayesian, using the previous confidence as a prior. For each word type w and possible language label ℓ in a newly-labeled document,

$$P(w \in \ell | evidence) = \frac{P_{prior}(w \in \ell)P(evidence|w \in \ell)}{P_{prior}(w \in \ell)P(evidence|w \in \ell) - (1 - P_{prior}(w \in \ell))P(evidence|w \notin \ell)}. \quad (3.3.1)$$

Confidence here is synonymous with $P(w \in \ell)$. We estimate $P(evidence|w \in \ell)$ with a document-specific confidence P_{doc} and $P(evidence|w \notin \ell)$ with $1 - P_{doc}$. This allows for different methods of calculating confidences to be used depending on the type of document. Note that $P(w \in \ell)$ could either increase or decrease because of the evidence in the document.

When the document is known to be completely monolingual (an example of this would be the Universal Declaration of Human Rights) in ℓ , it is safe to set the confidence of every word in that document to be 100% for ℓ . But when the document has the languages of its words automatically identified, we can adopt a more nuanced scheme. In this case, for all word

types w and all language labels ℓ that are actually assigned to any word in the document, we initialize $P_{doc}(w \in \ell)$ to 0.5. We then update these confidences as we iterate over each observed language and each token t with its type $w(t)$ and language $l(t)$:

Algorithm 2 Updating word-language confidences for automatically labeled documents.

```
for each labeled token  $t = (w, \ell)$  do
  for each language appearing in the document  $l$  do
    Let  $P_{prior}(w \in l) = P_{doc}(w \in l)$ 
    if  $l == \ell$  then
      Let  $P(evidence|w \in l) = \eta$ 
    else
      Let  $P(evidence|w \in l) = (1 - \eta)$ 
    end if
    Estimate  $P(evidence|w \notin l)$  with  $1 - P(evidence|w \in l)$ 
    Update  $P_{doc}(w \in l)$  according to equation 3.3.1
  end for
end for
```

where η is a constant that represents the probability of automatic language identification being correct (about 0.93 for the current method). Every instance of w labeled as ℓ acts as positive evidence for $P(w \in \ell)$ and every instance not labeled as ℓ acts as negative evidence.

POS-tags Using the methods of Chapter VII, we automatically create POS taggers for low-resource languages. These taggers are then run on every sentence in the appropriate language. As with language labels, we assign a confidence score to each POS tag. If, for example, a specific word consistently appears in a context that suggests that it is a verb and the tagger continues to label it as a verb in many different documents, the system's confidence in VERB as an appropriate tag for the word continues to grow.

Automatically-built dictionaries Just as the bootstrap Web-crawling process is able to iteratively create lists of high-confidence words, dictionaries can be similarly created.

Language identification can create language-specific word lists, POS-tagging can create tag dictionaries, bitext and alignment can create translation dictionaries and match foreign words with English definitions. Language identification word lists are already available and as the necessary features are added, the Minority Language Server will also make available for download the other types of dictionaries.

CHAPTER IV

Labeling the Languages of Words in Mixed-Language Documents using Weakly Supervised Methods

4.1 Introduction

In this chapter, we explore techniques for performing language identification at the word level in mixed language documents. Our results show that one can do better than independent word language classification, as there are clues in a word’s context: words of one language are frequently surrounded by words in the same language, and many documents have patterns that may be marked by the presence of certain words or punctuation. The methods in this chapter also outperform sentence-level language identification, which is too coarse to capture most of the shifts between language.

Our motivation for studying this problem stems from issues encountered while attempting to build language resources for low-resource languages. In trying to extend parts of Kevin Scannell’s Crúbadán project [21], which automatically builds low-resource language corpora from the Web, we found that the majority of webpages that contain text in a low-resource

language also contain text in other languages. Since Scannell’s method builds these corpora by bootstrapping from the pages that were retrieved, the corpus-building process can go disastrously wrong without accounting for this problem. And any resources, such as a lexicon created from the corpus, will also be incorrect.

To evaluate our methods, we collected and manually annotated a corpus of over 250,000 words of bilingual (though mostly non-parallel) text from the web. After running several different weakly-supervised learning methods, we found that a conditional random field model trained with generalized expectation criteria is the most accurate and performs quite consistently as the amount of training data is varied.

This chapter attempts to address three of the ongoing issues specifically mentioned by Hughes et al. [36] in their survey of textual language identification: supporting low-resource languages, sparse or impoverished training data, and multilingual documents.

In section 4.2, we define the task and describe the data and its annotation. Because the task of language identification for individual words had not been explicitly studied in the literature at the time of this work, and because of its importance to the overall task, we examine the features and methods that work best for independent word language identification in section 4.3. We begin to examine the larger problem of labeling the language of words in context in section 4.4 by describing our methods. In section 4.5, we describe the evaluation and present the results. We present our error analysis in section 4.6 and conclude in section 4.7.

4.2 Task Definition

The task we describe in this chapter is a sequence labeling problem, labeling a word in running text according to the language to which it belongs. In the interest of being able to produce reliable human annotations, we limit ourselves to texts with exactly two languages represented, though the techniques developed in this chapter would certainly be applicable to documents with more than two languages. The two languages represented in the document are known *a priori* by the labeler and the only training data available to the labeler is a small amount of sample text in each of the two languages represented.

In most NLP sequence labeling problems, the researchers can safely assume that each sequence (but not each item in the sequence) is independent and identically distributed (*iid*) according to some underlying distribution common to all the documents. For example, it is safe to assume that a sentence drawn from WSJ section 23 can be labeled by a model trained on the other sections. With the task of this chapter we cannot assume that sequences from different documents are *iid*, (*e.g.* One document may have 90% of its words in Basque, while another only has 20%), but we do make the simplifying assumption that sequences within the same document are *iid*.

Because of this difference, the labeler is presented each document separately and must label its words independently of any other document. And the training data for this task is not in the form of labeled sequences. Rather, the models in this task are given two monolingual example texts which are used only to learn a model for individual instances. Any sequential dependencies between words must be bootstrapped from the document. It is this aspect of the problem that makes it well-suited for weakly-supervised learning.

It is worth considering whether this problem is best approached at the word level, or if

perhaps sentence- or paragraph-level language identification would suffice for this task. In those cases, we could easily segment the text at the sentence or paragraph level and feed those segments to an existing language identifier. To answer this question we segmented our corpus into sentences by splitting at every period, exclamation point, or question mark (an overly aggressive approximation of sentence segmentation). Even if every sentence was given the correct majority label under this sentence segmentation, the maximum possible word-level accuracy that a sentence-level classifier could achieve is 85.8%, and even though this number reflects quite optimistic conditions, it is still much lower than the methods of this chapter are able to achieve.

4.2.1 Evaluation Data

To build a corpus of mixed language documents, we used the BootCat tool [126] seeded with words from a low-resource language. BootCat is designed to automatically collect webpages on a specific topic by repeatedly searching for keywords from a topic-specific set of seed words. We found that this method works equally well for languages as for topics, when seeded with words from a specific language. Once BootCat returned a collection of documents, we manually identified documents from the set that contained text in both the target language and in English, but did not contain text in any other languages. Since the problem becomes trivial when the languages do not share a character set, we limited ourselves to languages with a Latin orthography.

We found that there was an important balance to be struck concerning the popularity of a language. If a language is not spoken widely enough, then there is little chance of finding any text in that language on the Web. Conversely if a language is too widely spoken, then

Language	# words	Language	# words	Language	# words
Azerbaijani	4114	Hausa	2899	Oromo	28636
Banjar	10485	Hungarian	9598	Pular	3648
Basque	5488	Igbo	11828	Serbian	2457
Cebuano	17994	Kiribati	2187	Slovak	8403
Chippewa	15721	Kurdish	531	Somali	11613
Cornish	2284	Lingala	1359	Sotho	8198
Croatian	17318	Lombard	18512	Tswana	879
Czech	886	Malagasy	6779	Uzbek	43
Faroese	8307	Nahuatl	1133	Yoruba	4845
Fulfulde	458	Ojibwa	24974	Zulu	20783

Table 4.1: Languages present in the corpus and their number of words before separating out English text.

it is difficult to find mixed-language pages for it. The list of languages present in the corpus and the number of words in each language reflects this balance as seen in Table 4.1.

For researchers who wish to make use this data, the set of annotations used in this chapter is available from the author’s website⁶.

4.2.2 Annotation

Before the human annotators were presented with the mixed-language documents fetched by BootCat, the documents were first stripped of all HTML markup, converted to Unicode, and had HTML escape sequences replaced with the proper Unicode characters. Documents that had any encoding errors (*e.g.* original page used a mixture of encodings) were excluded from the corpus.

⁶<http://www-personal.umich.edu/~benking/resources/mixed-language-annotations-release-v1.0.tgz>

ENG:	if changing government for the better means losing the next election, so be it
SOT:	ntate Thabane, hao che sephali Wa Dihlaneng Ke DUMELA HORE KA MOTSOTSO ONA RE ...
ENG:	I so wish NUL will also be cartered for,
SOT:	hee re kena sekolo ha bohloko
ENG:	because of LUTARU.Thank you ntate T.T! Sevice delivery for 3% people Mosisilli, he said that ...
SOT:	Retselisitsoemonethi ekare jwale hotla sebetswa
ENG:	Lesotho is heading 4 development big-ups Mr Tom Thabane Mohanuo a please people do not ...
SOT:	Mathabo Letsie http://www.facebook.com/taole.stawie Taole Stawie Mokobori
ENG:	As Zuma did he should introduce a way of we can report corruption straight to his office

Table 4.2: An example of text from an annotated English-Sotho web page.

Since there are many different reasons that the language in a document may change (*e.g.* code-switching, change of authors, borrowing) and many variations thereof, we attempted to create a broad set of annotation rules that would cover many cases, rather than writing a large number of very specific rules. In cases when the language use was ambiguous, the annotators were instructed simply to make their best guess. Table 4.2 shows an example of an annotated document.

Generally, only well-digested English loanwords and borrowings were to be marked as belonging to the foreign language. If a word appeared in the context of both languages, it was permissible for that word to receive different labels at different times, depending on its context.

Ordinary proper names (like "John Williams" or "Chicago") were to be marked as belonging to the language of the context in which they appear. This rule also applied to abbreviations (like "FIFA" or "BBC"). The exception to this rule was proper names composed of common nouns (like "Stairway to Heaven" or "American Red Cross") and to abbreviations that spelled out English words, which were to be marked as belonging to the language of the words they were composed of.

The annotators were instructed not to assign labels to numbers or punctuation, but they were allowed to use numbers as punctuation as clues for assigning other labels.

4.2.3 Human Agreement

To verify that the annotation rules were reasonable and led to a problem that could potentially be solved by a computer, we had each of the annotators mark up a small shared set of a few hundred words from each of eight documents, in order to measure the inter-annotator agreement.

The average actual agreement was 0.988, with 0.5 agreement expected by chance for a kappa of 0.975.

4.2.4 Training Data

Following Scannell [21], we collected small monolingual samples of 643 languages from four sources: the Universal Declaration of Human Rights, non-English Wikipedias⁷, the Jehovah's Witnesses website⁸, and the Rosetta project [133]. The samples are about 2000 words long on average.

Only 30 of these languages ended up being used in experiments. Table 4.3 shows the sizes of the monolingual samples of the languages used in this chapter. They range from 92 for Chippewa to 16469 for English. Most of the languages have between 1300 and 1600 words in their example text. To attempt to mitigate variation caused by the sizes of these language

⁷As of February 2011, there were 113 Wikipedias in different languages. Current versions of Wikipedia can be accessed from http://meta.wikimedia.org/wiki/List_of_Wikipedias

⁸As of February 2011, there were 310 versions of the site available at <http://www.watchtower.org>

Language	# words	Language	# words	Language	# words
Azerbaijani	211	Hausa	2677	Pular	1285
Banjar	450	Hungarian	1541	Serbian	1515
Basque	1378	Igbo	2079	Slovak	1504
Cebuano	1898	Kiribati	1891	Somali	1871
Chippewa	92	Kurdish	1674	Sotho	2154
Cornish	2096	Lingala	1816	Tswana	2191
Croatian	1505	Lombard	2955	Uzbek	1533
Czech	1503	Malagasy	4038	Yoruba	2454
English	16469	Nahuatl	3544	Zulu	1075
Faroese	1585	Ojibwa	167		
Fulfulde	1097	Oromo	1443		

Table 4.3: Number of total words of training data for each language.

samples, we sample an equal number of words with replacement from each of English and a second language to create the training data.

4.3 Word-level Language Classification

We shift our attention momentarily to a subproblem of the overall task: independent word-level language classification. While the task of language identification has been studied extensively at the document, sentence, and query level, little or no work has been done at the level of an individual word. For this reason, we feel it is prudent to formally evaluate the features and classifiers which perform most effectively at the task of word language classification (ignoring any sequential dependencies at this point).

Features	Accuracy
Unigrams	0.8056
Bigrams	0.8783
Trigrams	0.8491
4-grams	0.7846
5-grams	0.6977
{1,2,3,4,5}-grams	0.8817
{1,2,3,4,5}-grams, word	0.8819

Table 4.4: Logistic regression accuracy when trained using varying features.

4.3.1 Features

We used a logistic regression classifier to experiment with combinations of the following features: character unigrams, bigrams, trigrams, 4-grams, 5-grams, and the full word. For these experiments, the training data consisted of 1000 words sampled uniformly with replacement from the sample text in the appropriate languages. Table 4.4 shows the accuracies that the classifier achieved when using different sets of features averaged over 10 independent runs.

The use of all available features seems to be the best option, and we use the full set of features in all proceeding experiments. This result also concurs with the findings of [39], who found 1-5-grams to be most effective for document language classification.

4.3.2 Classifiers

Using all available features, we compare four MALLET [134] classifiers: logistic regression, naïve Bayes, decision tree, and Winnow2. Figure 4.3.1 shows the learning curves for each classifier as the number of sampled words comprising each training example is varied from

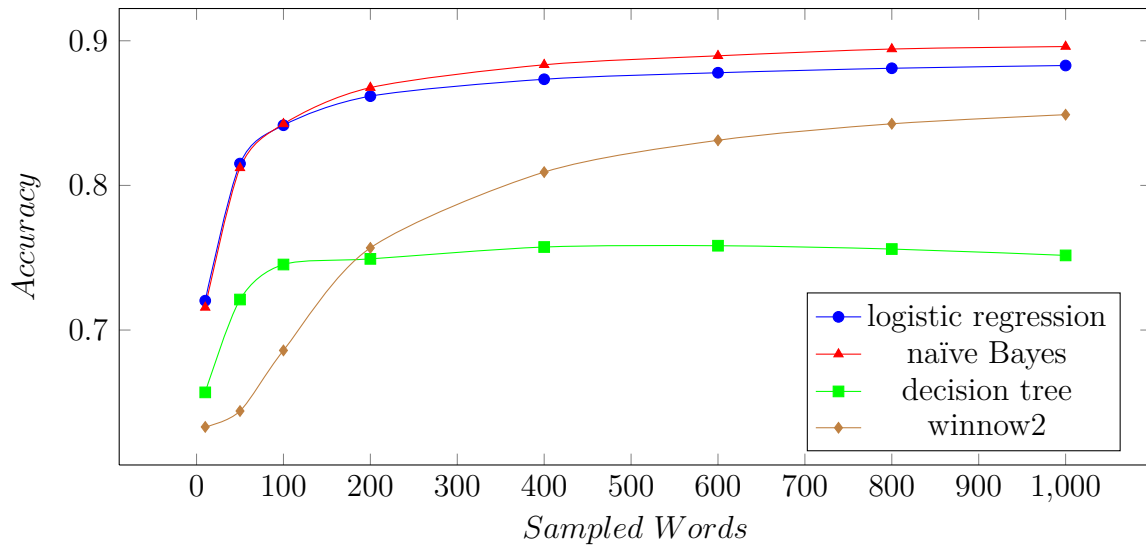


Figure 4.3.1: Learning curves for logistic regression, naïve Bayes, decision tree, and Winnow2 on the independent word classification problem as the number of sampled words in each training example changes from 10 to 1000.

10 to 1000.

Since a naïve Bayes classifier gave the best performance in most experiments, we use naïve Bayes as a representative word classifier for the rest of the chapter.

4.4 Methods

Moving onto the main task of this chapter, labeling *sequences* of words in documents according to their languages, we use this section to describe our methods.

Since training data for this task is limited and is of a different type than the evaluation data (labeled instances from monolingual example texts *vs.* labeled sequences from the multilingual document), we approach the problem with weakly- and semi-supervised methods.

The sequence labeling methods are presented with a few new sequence-relevant features,

which are not applicable to independent word classification (since these features do not appear in the training data):

- a feature for the presence of each possible non-word character (punctuation or digit) between the previous and the current words
- a feature for the presence of each possible non-word character between the current and next words

In addition to independent word classification, which was covered in section 4.3, we also implement a conditional random field model trained with generalized expectation criteria, a hidden Markov model (HMM) trained with expectation maximization (EM), and a logistic regression model trained with generalized expectation criteria.

We had also considered that a semi-Markov CRF [135] could be useful if we could model segment lengths (a non-Markovian feature), but we found that gold-standard segment lengths did not seem to be distributed according to any canonical distribution, and we did not have a reliable way to estimate these segment lengths.

4.4.1 Conditional Random Field Model trained with Generalized Expectation

Recall that generalized expectation (GE) criteria [64] are terms added to the objective function of a learning algorithm which specify preferences for the learned model. When using a linear chain CRF as we are here, the objective function is as follows:

$$\mathcal{O}(\theta) = \sum_d \log p_\theta(y^{(d)}|x^{(d)}) - \frac{\sum_k \theta_k}{2\sigma^2} - \lambda KL(\hat{p}||\tilde{p}_\theta), \quad (4.4.1)$$

where \hat{p} is the preferred distribution and \tilde{p} is the distribution under the current model. Practically, to compute these expectations, we produce the smoothed MLE on the output label distribution for every feature observed in the training data. For example, the trigram “**ter**” may occur 27 times in the English sample text and 34 times in the other sample text, leading to an MLE of $\hat{p}(\mathbf{eng}|\mathbf{ter}) \approx 0.44$.

Because we do not expect the true marginal label distribution to be uniform (*i.e.* the document may not have equal numbers of words in each language), we first estimate the expected marginal label distribution by classifying each word in the document independently using naïve Bayes and taking the resulting counts of labels produced by the classifier as an MLE estimate for it: $\hat{p}(\mathbf{eng})$ and $\hat{p}(\mathbf{non})$.

We use these terms to bias the expected label distributions over each feature. Let \mathcal{F}_{eng} and \mathcal{F}_{non} respectively be the collections of all training data features with the two labels. For every label $l \in \mathcal{L} = \{\mathbf{eng}, \mathbf{non}\}$ and every feature $f \in \mathcal{F}_{eng} \cup \mathcal{F}_{non}$, we calculate

$$p(l|f) = \frac{\text{count}(f, \mathcal{F}_l) + \delta}{\text{count}(f, \cup_i \mathcal{F}_i) + \delta|\mathcal{L}|} \times \frac{\hat{p}(l)}{p_{uniform}(l)},$$

the biased maximum likelihood expected output label distribution. To avoid having $p(l|f) = 0$, which can cause the KL-divergence to be undefined, we perform additive smoothing with $\delta = 0.5$ on the counts before multiplying with the biasing term.

We use the implementation of CRF with GE criteria from MALLET [134], which uses a gradient descent algorithm to optimize the objective function [67, 136].

4.4.2 Hidden Markov Model trained with Expectation Maximization

A second method we used was a hidden Markov model (HMM) trained iteratively using the Expectation Maximization algorithm [137]. Here an HMM is preferable to a CRF because it is a generative model and therefore uses parameters with simple interpretations. In the case of an HMM, it is easy to estimate emission and transition probabilities using an external method and then set these directly.

To initialize the HMM, we use a uniform distribution for transition probabilities, and produce the emission probabilities by using a naïve Bayes classifier trained over the two small language samples.

In the expectation step, we simply pass the document through the HMM and record the labels it produces for each word in the document.

In the maximization step, we produce maximum-likelihood estimates for transition probabilities from the transitions between the labels produced. To estimate emission probabilities, we retrain a naïve Bayes classifier on the small language samples along the set of words from the document that were labeled as being in the respective language. We iterated this process until convergence, which usually took fewer than 10 iterations.

We additionally experimented with a naïve Bayes classifier trained by EM in the same fashion, except that it had no transition probabilities to update. This classifier’s performance was almost identical to that of the GE-trained MaxEnt method mentioned in the following section, so we omit it from the results and analysis for that reason.

4.4.3 Logistic Regression trained with Generalized Expectation

GE criteria can also be straightforwardly applied to the weakly supervised training of logistic regression models. The special case where the constraints specified are over marginal label distributions, is called *label regularization*.

As with the CRF constraint creation, here we first use an ordinary supervised naïve Bayes classifier in order to estimate the marginal label distributions for the document, which can be used to create more accurate output label expectations that are biased to the marginal label distributions over all words in the document.

We use the MALLET implementation of a GE-trained logistic regression classifier, which optimizes the objective function using a gradient descent algorithm.

4.4.4 Word-level Classification

Our fourth method served as a baseline and did not involve any sequence labeling, only independent classification of words. Since naïve Bayes was the best performer among word classification methods, we use that the representative of independent word classification methods. The implementation of the naïve Bayes classifier is from MALLET.

We also implemented a self-trained CRF, initially trained on the output of this naïve Bayes classifier, and trained on its own output in subsequent iterations. This method was not able to consistently outperform the naïve Bayes classifier after any number of iterations.

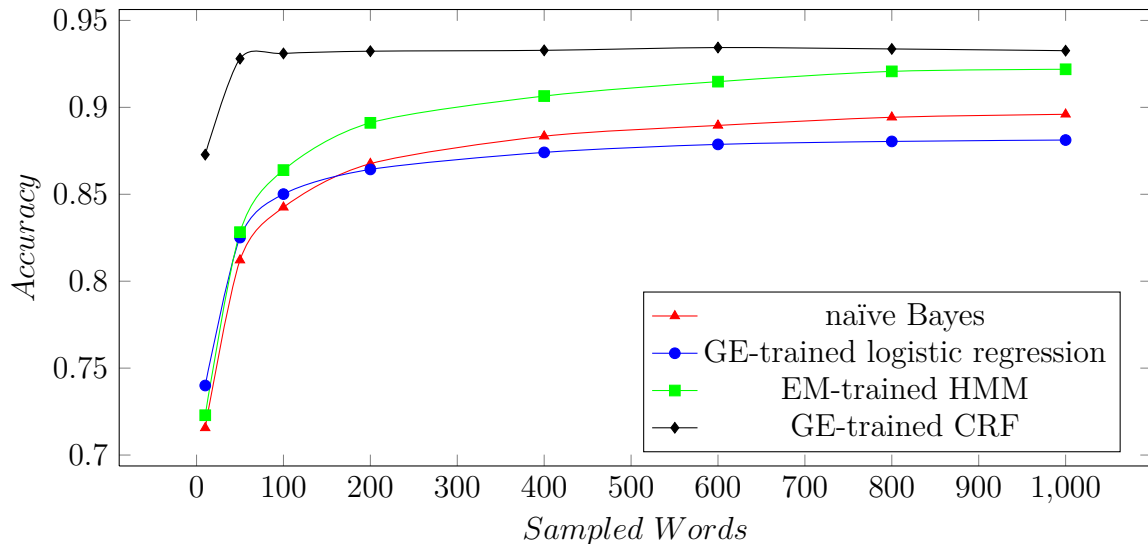


Figure 4.5.1: Learning curves for naïve Bayes, logistic regression trained with GE, HMM trained with EM, and CRF trained with GE as the number of sampled words in each training example changes from 10 to 1000.

4.5 Evaluation and Results

We evaluated each method using simple token-level accuracy, *i.e.* whether the correct label was assigned to a word in the document. Word boundaries were defined by punctuation or whitespace, and no tokens containing a digit were included. Figure 4.5.1 displays the accuracy for each method as the number of sampled words from each language example is varied from 10 to 1000.

In all the cases we tested, CRF trained with GE is clearly the most accurate option among the methods examined, though the EM-trained HMM seemed to be approaching a similar accuracy with large amounts of training data. With a slight edge in efficiency also in its favor, we think the GE+CRF approach, rather than EM+HMM, is the best approach for this problem because of its consistent performance across a wide range of training data sizes.

In its favor, the EM+HMM approach has a slightly lower variance in its performance across different files, though not at a statistically significant level.

Contrary to most of the results in [138], a logistic regression classifier trained with GE did not outperform a standard supervised naïve Bayes classifier. We suspect that this is due to the different nature of this problem as compared to most other sequence labeling problems, with the classifier bootstrapping over a single document only. In the problems studied by Mann and McCallum, the GE-trained classifier was able to train over the entire training set, which was on average about 50,000 instances, far more than the number of words in the average document in this set (2,500).

4.6 Error Analysis

In order to analyze the types of mistakes that the models made we performed an error analysis on ten randomly selected files, looking at each mislabeled word and classifying the error according to its type. The results of this analysis are in Table 4.5. The three classes of errors are (1) named entity errors, when a named entity is given a label that does not match the label it was given in the original annotation, (2) shared word errors, when a word that could belong to either language is classified incorrectly, and (3) other, a case that covers all other types of errors.

Our annotation rules for named entities specified that named entities should be given a label matching their context, but this was rather arbitrary, and not explicitly followed by any of the methods, which treat a named entity as if it was any other token. This was the one of most frequent types of error made by each of the methods and in our conclusion in section 4.7, we discuss ways to improve it.

Method	NE	SW	Other
GE+CRF	41%	10%	49%
EM+HMM	50%	14%	35%
GE+MaxEnt	37%	12%	51%
Naïve Bayes	42%	17%	40%

Table 4.5: Types of errors and their proportions among the different methods. NE stands for Named Entity, SW stands for Shared Word, and Other covers all other types of errors.

In a regression analysis to determine which factors had the greatest correlations with the GE-trained CRF performance, the estimated proportion of named entities in the document had by far the greatest correlation with CRF accuracy of anything we measured. Following that in decreasing order of correlation strength were the cosine similarity between English and the document’s second language, the number of words in the monolingual example text (even though we sampled from it), and the average length of gold-standard monolingual sequences in the document.

The learning curve for GE-trained CRF in Figure 4.5.1 is somewhat atypical as far as most machine learning methods are concerned: performance is typically non-decreasing as more training data is made available.

We believe that the model is becoming over-constrained as more words are used to create the constraints. The GE method does not have a way to specify that some of the soft constraints (for the labels observed most frequently in the sample text) should be more important than other constraints (those observed less frequently). When we measure the KL-divergence between the label distributions predicted by the constraints and the true label distribution, we find that this divergence seems to reach its minimum value between 600 and 800 words, which is where the GE+CRF also seems to reach its maximum performance.

The step with a naïve Bayes classifier estimating the marginal label distribution ended up being quite important overall. Without it, the accuracy dropped by more than a full percentage point absolute. But the problem of inaccurate constraint estimation is one that needs further consideration. Some possible ways to address it may be to prune the constraints according to their frequency or perhaps according to a metric like entropy, or to vary the GE-criteria coefficient in the objective function in order to penalize the model less for varying from the expected model.

4.7 Conclusion

This chapter addresses three of the ongoing issues specifically mentioned by Hughes et al. [36] in their survey of textual language identification. Our approach is able to *support low-resource languages*; in fact, almost all of the languages we tested on would be considered low-resource languages. We also address the issue of *sparse or impoverished training data*. Because we use weakly-supervised methods, we are able to successfully learn to recognize a language with as few as 10 words of training data⁹. The last and most obvious point we address is that of *multilingual documents*, which is the focus of the chapter.

We present a weakly-supervised system for identifying the languages of individual words in mixed-language documents. We found that across a broad range of training data sizes, a CRF model trained with GE criteria is an accurate sequence classifier and is preferable to other methods for several reasons.

One major issue to be improved upon in future work is how named entities are handled. A straightforward way to approach this may be to create another label for named entities,

⁹With only 10 words of each language as training data, the CRF approach correctly labels 88% of words

which (for the purposes of evaluation) would be considered not to belong to any of the languages in the document. We could simply choose not to evaluate a system on the named entity tokens in a document. Alternatively, the problem of language-independent named entity recognition has received some attention in the past [139], and it may be beneficial to incorporate such a system in a robust word-level language identification system.

The methods in this chapter make two key assumptions about the input documents that might limit their applicability. First they assume that (and have only been tested in the case where) documents have exactly two languages mixed together. Second they assume knowledge of which two languages are present in the document. While these methods do bring us closer to being able to build an automated corpus building system for low-resource languages, methods that are applicable in a broader range of situations will ultimately be necessary. The next chapter describes methods for word-level language identification that make neither assumptions about the number of language, nor assumptions about the identities of those languages.

CHAPTER V

Bayesian Methods for Word-Level Language Identification in Mixed-Language Documents

5.1 Introduction

The problem we consider in this chapter is a more general version of the previous chapter's problem of identifying the languages of words in documents that have one or more languages mixed together in the same document. For this chapter, the set of possible languages is large and the number of languages present in the document is not known a priori. To our knowledge, while research has gone into more restricted versions of this problem, no previous research has taken on this more general task.

Unfortunately this broader formulation means that there are no previous methods that are directly comparable. For the sake of comparison, we implement several new methods in this chapter and compare against several baselines. The new methods we describe in this chapter are Bayesian generative models: (§ 5.4.1) a basic multinomial mixture model, (§ 5.4.2) a hierarchical mixture model, and (§ 5.4.3) an HMM-based model.

This chapter’s contribution is the creation of generative models and inference methods to the problem of inferring the languages of words in a document. We apply these methods to a corpus of mixed-language web documents with a large set of possible languages and are able to correctly label the languages of words with greater than 96% accuracy.

5.2 Background

Many of the models we use in this chapter are based on unsupervised topic models. The key assumption behind topic models is that text in a collection of documents can be grouped into topics, groups of semantically related words that ideally line up with the human understanding of topics. For our adaptations of these models, we call the groups of related words “languages” instead of “topics” and don’t learn languages in an unsupervised way (as topic models learn topics), but learn which words belong to which languages using monolingual training data.

The most popular early topic model was latent semantic indexing (LSI), which was based on matrix decomposition [140]. This model was later given greater expressiveness and a stronger statistical footing with probabilistic latent semantic indexing (PLSI) [141]. PLSI was able to represent both word-topic distributions and document-topic distributions, but not a prior distribution over either of these. Latent Dirichlet Allocation [142] was the first fully Bayesian topic model in this family of topic models. It introduced Dirichlet priors, which simultaneously reduced the over-fitting issues that were common with PLSI and allowed for classification of documents outside the training set. Since that time, many generative Bayesian topic models have been proposed with different features [143–146] to be appropriate in a number of different settings [147–149].

5.3 Data

As in Chapter IV, we make use of small monolingual language samples, using the same set of 643 documents. The corpus of the previous chapter, however, was not ideal in its current form for use in these experiments, since every document contained exactly two languages. To build an appropriate corpus for this chapter, we expanded the corpus by collecting and annotating documents, but not restricting them to have exactly two languages mixed together. As before, we found that agreement between annotators on our corpus was greater than 99%. We used the following procedure to collect and annotate documents:

Algorithm 3 Corpus building method.

```
for  $q$  = query language do
  Sample three tokens from text in  $q$ 
  Submit tokens to Google search engine
  for  $p$  = page in result set do
    Manually label each word in  $p$  according to its language
    Store  $q$  as  $p$ 's query language
  end for
end for
```

We issued queries for 44 low-resource languages, leading to a corpus with 487 documents, 1.2 million words and 51 different languages. Whenever possible, language variants were mapped to ISO 639-3 macro-languages. Each word in each document was manually labeled according to its language. We did not find any instances of a page containing text in a language for which we did not also have a corresponding monolingual language sample. Table 5.6 below shows examples of mixed language text in our corpus.¹⁰ Not counting the query language, English was the most common language to find in a web document, followed by Spanish, French, Portuguese, and Czech.

¹⁰We also plan to release this dataset publicly, but because the paper submission corresponding to this chapter has not yet been accepted, it has not yet been made available

Ojibwa, English	...Dibishkoo asiniig gii- izhinaagoziwag. Like stones was their look. Aaniin ekidoyan? What did you say? ...
Nahuatl, English	... ocatcah tēl īpan Teignmouth Secondary School achto xiuhpan 1990s...
Basque, Dutch, French	...burutzeko eskubidea ematen zaionik. Met dank aan Django Encore merci! Milesker! 08:35 Gepost door ...
Cornish, English	...Standard rag Screfa an Tavas Kernowek Improving the Standard Written Form Amendya an Form Screfys Standard ...
Kurdish, English	... ve tēn rêvebirin. Li gorî rêbaza “States Reorganisation Act” a 1956’an her rêveberiya cihan ...
Croatian, Spanish	...postao je profesorom na UNAM (Universidad Nacional Autónoma de Mexico) i osnovalo psihoanalitički odjel u medicinskoj školi...

Table 5.6: Examples of mixed language usage in web pages in our corpus.

Figure 5.3.1 illustrates the frequencies at which documents in the corpus contain different numbers of languages. Monolingual documents are the most common, and additional languages in the same document are increasingly uncommon. Because monolingual web pages are so commonly returned by the search engine, pages with text in only a single language were also included to ensure that the methods developed would also be appropriate for real-world applications.

5.4 Methods

We start out first by describing the variables that all the models have in common. Table 5.7 shows descriptions of these variables.

The matrix β represents a collection of multinomial distributions over the vocabulary, one distribution for each language. Each entry $\beta_{(w,\ell)}$ contains $P(w|\ell)$. By applying Bayes’ rule, $P(w|\ell) \propto P(\ell|w)\dot{P}(w)$, where $P(\ell)$ is a normalizing constant that can be dropped. The term $P(w)$ also ends up cancelling out when performing inference, since the variable or block

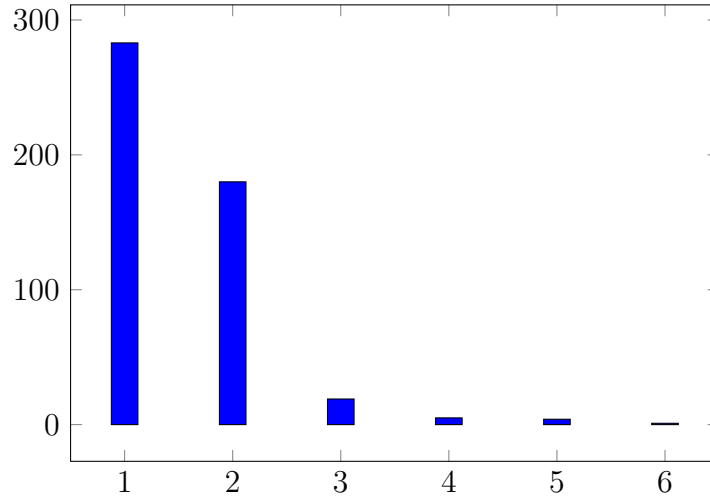


Figure 5.3.1: The frequency of documents with different numbers of languages in our corpus.

Symbol	Description
D	a document.
\mathcal{L}	a set of possible languages.
\mathcal{V}	a vocabulary containing all words observed in any document.
\mathbf{w}	the words of D .
\mathbf{z}	a hidden variable, replicated once per word per document, representing a word's language.
β	a $ \mathcal{V} \times \mathcal{L} $ matrix with each entry (w, ℓ) representing the $P(w \ell)$.
θ	the multinomial language distribution of each document.
α	A Dirichlet prior distribution over the document-language distribution.

Table 5.7: Parameters common to the models.

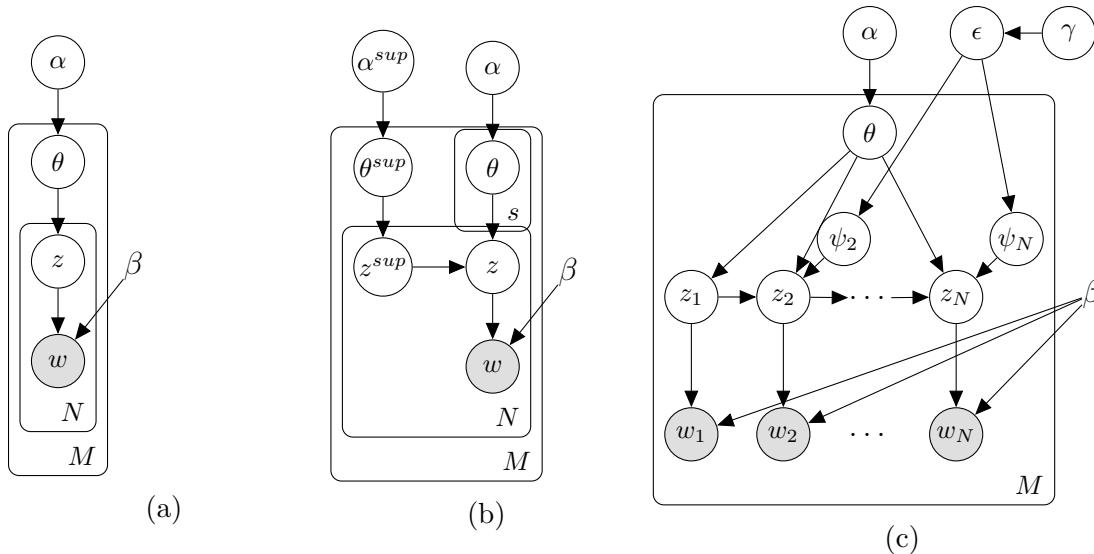


Figure 5.4.1: Plate diagrams for the Bayesian language inference models: (5.4.1a) the basic mixture model, (5.4.1b) the hierarchical model, and (5.4.1c) the HMM-based model.

being sampled at any time is fixed.

We populate β using a logistic regression classifier that assigns a probability distribution over the set of languages to each word observed in the corpus. To train this classifier, we use 1000 sampled words for each language drawn from its respective monolingual language sample. Each word is represented as a bag of features, with the full word and all 2-, 3-, 4-, or 5-grams comprising the features. We had also considered using a naïve Bayes classifier to estimate $P(w|\ell)$ directly, but we found that this severely degraded the performance of the models.

The experimental setup we use is leave-one-query-language-out cross validation. This means that we divide the set of annotated documents according to the language of the query by which they were retrieved, using documents from 43 of the query languages for a labeled “training” set, and the documents from the remaining query language as a test set. The methods are allowed to look at the labels in the training set, and these labels are only used

for estimating priors.

This simulates a corpus-building scenario, where researchers want to crawl the Web looking for text in a new language and use the methods of this chapter to label the languages. Having a set of labeled documents is also not unrealistic. In a real-world scenario, the corpus of labeled documents used here, which we intend to make available to other researchers, could act as the labeled corpus.

5.4.1 Basic Mixture Model

The first model is a mixture of language multinomials, similar to Latent Dirichlet Allocation (LDA) [142]. It models each document as having a hidden multinomial distribution θ over the set of possible languages, representing the proportions of different languages present in that document. Each word in the document has both a word identity w and a latent language z . The matrix β models the distribution of word types over languages.

More succinctly, this model has the following generative story for a document D :

1. Draw $\theta \sim \text{Dirichlet}(\alpha)$
2. For i in $(1, \dots, |D|)$:
 - (a) Choose a language $z_i \sim \text{Multinomial}(\theta)$
 - (b) Choose a word $w_i \sim \text{Multinomial}(\beta_{z_i})$

Figure 5.4.1a also shows a plate diagram for the generative process. We use an inference method similar to the collapsed Gibbs sampling-based inference for LDA described by Griffiths and Steyvers [150]. In place of LDA topics, we have languages. The set of languages is

fixed beforehand and does not need to be learned from the corpus. In each iteration of Gibbs sampling, we sample the latent language label for each according to the following update rule:

$$P(z_i = \ell | \mathbf{z}_{-i}, \mathbf{w}) \propto \frac{n_{-i,\ell} + \alpha_\ell}{n_{-i,\cdot} + \sum_{\ell'} \alpha_{\ell'}} \beta(w_i, \ell)$$

where \mathbf{z}_{-i} is the current set of language labels excluding the i th word and $n_{-i,\ell}$ is the number of words currently labeled with language ℓ not including the i th word.

Because β is a constant that is independent of the document (recall that it was trained on sample monolingual data), the Gibbs sampling method is appropriate for performing inference on each document separately. Therefore the labeled data is used only to infer appropriate values for α , the Dirichlet prior over document-language distributions. Practically this prior influences two things: (1) the number of different languages in a document and (2) the relative proportions of those languages.

We use an asymmetric Dirichlet prior, which allows our system to learn, for example, that certain colonial languages, like English or French, tend to appear much more frequently than other languages. In the leave-only-query-language-out cross validation setting, we use the labeled data to estimate α_ℓ for each language ℓ that appears in the training set. This, of course, leads to a problem when the language q in which we issued the queries does not appear in the labeled data. To avoid this problem, we estimate α_q for the query language by dividing the training set up by query language and averaging the estimated α value for each training set query language over its documents. This same method is also used for the hierarchical and HMM-based models.

5.4.2 Hierarchical Model

Languages organize naturally into hierarchies, most often into phylogenetic trees, though in this model we group languages together when they commonly occur in the same document.. Organizing languages into a hierarchy could help prevent a common error made by the basic mixture model, in which the model assigns probability mass over a set of similar languages rather than settling on just one language. The documents in our corpus tend to have very few languages, and therefore it is much more likely, for example, that words a and b in a document are actually both Czech words or both Slovak words than it is that a is Czech and b is Slovak, or vice-versa.

In this model, we organize languages into a two-level hierarchy, with languages grouped into “super-languages.” This is similar to Pachinko Allocation [151], an extension to LDA that puts topics into a hierarchy. This model has the following generative story, first selecting a super-language for each word and then using the super-language to select an actual language label (see Figure 5.4.1b):

1. Draw $\theta^{sup} \sim \text{Dirichlet}(\alpha^{sup})$
2. For j in $(1, \dots, |\mathcal{L}|)$:
 - (a) Draw $\theta_j \sim \text{Dirichlet}(\alpha)$
3. For i in $(1, \dots, |D|)$:
 - (a) Choose a super-language $z_i^{sup} \sim \text{Multinomial}(\theta^{sup})$
 - (b) Choose a language $z_i \sim \text{Multinomial}(\theta_{z_i^{sup}})$
 - (c) Choose a word $w_i \sim \text{Multinomial}(\beta_{z_i})$

The number of super-languages is a parameter that must be fixed before iteration. Because mixed-language documents tend to have very few languages in a single document, the number of super-languages does not need to be high. We use five super-languages in our inference.

There is also the issue of reasonably grouping languages into super-languages. The labeled data has every word labeled according to its language, and from the co-occurrences of languages in documents, we need learn to associate languages with super-languages. This problem turns out to be another instance of the topic modeling problem, except that here the language labels are the “words” and the super-languages are the “topics.” We run 100 iterations of Gibbs sampling LDA on the labeled data in order to associate languages with super-languages. The language Dirichlet prior α is estimated from the labeled data as usual, but the super-language Dirichlet prior α^{sup} is estimated from the output of this LDA process.

After organizing the languages into super-languages, we again perform inference by collapsed Gibbs sampling. The algorithm we used is based on the Gibbs sampling algorithm in [151] for Pachinko allocation. The super-languages for each word are initialized to the single most likely super-language given the word’s initial language. (See section 5.5.1 for more details on how the languages for each word are initialized.) We use the following update rule:

$$P(z_i = \ell, z_i^{sup} = k | \mathbf{z}_{-i}, \mathbf{z}_{-i}^{sup}, \alpha, \alpha^{sup}) \propto \frac{n_{-i,k}^{sup} + \alpha_k^{sup}}{n_{-i,\cdot}^{sup} + \sum_{k'} \alpha_{k'}^{sup}} \frac{n_{-i,\ell,k} + \alpha_\ell}{n_{-i,\cdot,k} + \sum_{\ell'} \alpha_{\ell'}} \beta(w_i, \ell) \quad (5.4.1)$$

Because the sampler must update every combination of topic and super-topic on each iteration in addition to learning language-to-super-language associations, the hierarchical model is much slower than the basic mixture model even though their inference methods are quite similar.

5.4.3 HMM-based Model

From observing real-world mixed-language documents, it is quite clear that the *bag-of-words* assumption made by the previous models is inappropriate. These documents typically have long stretches of words in the same language, so it is reasonable to consider models that do not treat each word independently, but assign it a language that depends not only on its own identity, but also on the identities and/or labels of nearby words.

This third model is based on a hidden Markov model (HMM) and chooses a word's language based on the observed word and the label of the previous word. It also makes one crucial simplifying assumption over ordinary HMMs: the transition matrix does not model every possible language bigram. In an ordinary HMM, the transition matrix has $O(|\mathcal{L}|^2)$ parameters that must be learned. Since the transition matrix is different for every document, this is simply too many parameters to learn with the amount of data available.

Instead of the normal transition matrix, all $z_i \rightarrow z_i$ transitions along the diagonal are constrained to take the same value and every other $z_i \rightarrow z_j$ transition depends only on z_j , as in [152]. Stated another way, at every position a choice (ψ) is made to either use the same label as the previous word ($\psi = 0$) or to sample a new word from θ , the document-language multinomial distribution ($\psi = 1$). In most real documents, ψ will have a value of zero much more often than a value of 1, meaning that there are long stretches of text in the same language. The model uses a binomial prior with parameter ϵ to steer ψ toward being zero most of the time.

This model has the following generative story for each document (Fig. 5.4.1c):

1. Draw $\theta \sim \text{Dirichlet}(\alpha)$

2. Draw $\epsilon \sim \text{Beta}(\gamma_1, \gamma_2)$
3. Set $\psi_1 = 1$
4. For i in $(2, \dots, |D|)$:
 - (a) Draw $\psi_i \sim \text{Binomial}(\epsilon)$
5. For i in $(1, \dots, |D|)$:
 - (a) If $\psi_i = 0$, set $z_i = z_{i-1}$
 - (b) Else draw $z_i \sim \text{Multinomial}(\theta)$
 - (c) Choose a word $w_i \sim \text{Multinomial}(\beta_{z_i})$

The beta prior on ϵ is defined by its two parameters γ_1 and γ_2 . We set γ_1 to be equal to the number of consecutive tokens in the labeled data that share the same label and γ_2 to be equal to the number with different labels. Without such a prior on ϵ , its probability of selecting $\psi = 1$ was systematically overestimated.

For token- and sentence-based inference, we use an EM algorithm, which is based on the algorithm presented in [153]. In the E-step, we use the forward-backward algorithm to estimate each z_i and ψ_i . In the M-step, we re-estimate θ and ϵ from these values. We use 30 iterations of EM for each document, which seems to be sufficient for convergence. The EM inference method is the fastest of the models we compare in this chapter. For type-based inference, we use a version of the algorithm in [154] that is appropriate for the simplified structure of the HMM's transition matrix.

Gran.	Init.	Mixture				Hierarchical				HMM-based			
		A	P	R	F	A	P	R	F	A	P	R	F
token	rand.	0.868	0.741	0.356	0.481	0.886	0.729	0.387	0.505	0.894	0.828	0.434	0.570
token	word	0.868	0.742	0.357	0.482	0.880	0.877	0.372	0.489	0.894	0.827	0.435	0.570
token	sent.	0.873	0.747	0.368	0.493	0.877	0.693	0.358	0.472	0.893	0.827	0.433	0.568
token	doc.	0.896	0.758	0.418	0.539	0.877	0.722	0.367	0.487	0.896	0.821	0.439	0.572
sent.	rand.	0.937	0.672	0.514	0.582	0.932	0.735	0.516	0.607	0.937	0.872	0.576	0.693
sent.	word	0.962	0.810	0.655	0.737	0.940	0.806	0.569	0.667	0.937	0.872	0.576	0.693
sent.	sent.	0.962	0.726	0.655	0.688	0.939	0.745	0.548	0.632	0.937	0.872	0.576	0.693
sent.	doc.	0.893	0.096	0.081	0.088	0.935	0.657	0.499	0.567	0.937	0.872	0.576	0.693
type	rand.	0.882	0.755	0.388	0.513	0.817	0.660	0.263	0.376	0.889	0.791	0.413	0.543
type	word	0.881	0.757	0.386	0.511	0.892	0.719	0.396	0.511	0.818	0.726	0.282	0.406
type	sent.	0.883	0.755	0.389	0.513	0.887	0.672	0.370	0.477	0.822	0.726	0.287	0.411
type	doc.	0.908	0.766	0.452	0.568	0.892	0.710	0.394	0.507	0.882	0.761	0.390	0.515

Baselines and Comparisons	A	P	R	F
random	0.017	0.017	0.002	0.003
mixture of words	0.491	0.466	0.083	0.141
mixture of sentences	0.893	0.741	0.407	0.526
document classification	0.893	0.096	0.081	0.088
CRF+GE	0.630	0.619	0.142	0.231
HMM	0.370	0.308	0.046	0.080

Table 5.8: Results for the inference methods with various granularity and initialization parameters, along with baselines and comparisons. A stands for accuracy, P stands for minority class precision, R stands for minority class recall, and F stands for minority class F1-score.

5.5 Evaluation and Results

We evaluate the methods described along with a few baseline and comparison methods. All the Gibbs sampling methods were run for 500 iterations with a burn-in period of 100 iterations and thinning of 10 (sampling every 10th iteration). Table 5.8 shows the results of evaluation on the output of the different methods under various conditions of initialization and granularity.

5.5.1 Initialization and Granularity

We experimented with several different initialization methods along with three different types of granularity.

We compared token-level inference, where the label of each word was inferred separately, with sentence-level inference, where the labels of all the words in a sentence were inferred jointly, and type-based inference [154], where the labels of every instance of a word type were inferred jointly.

We used four types of initialization:

- Random – the initial label was selected at random among the set of all possible languages
- Best word – each word was initialized to its most likely language
- Best sentence – the words in each sentence were initialized to the most likely language for its sentence (product over the words in the sentence)

- Best document – each word was initialized to the most likely language for the whole document (product over all the words in the document)

5.5.2 Baselines and Comparisons

We compare against four baselines: *random*, where each word is given a random label from among the set of all possible languages, *mixture of words*, where each word is independently given its most likely label, *mixture of sentences*, where each word in a sentence is given the label that maximizes the joint likelihood of all the words in the sentence, and *document classification*, where each word in the document is given the label that maximizes the joint likelihood of all the words in the document.

We also compare against models from Chapter IV, which are appropriate when the document contains a small number of known languages. Since the number of languages and the identity of languages is unknown in our scenario, we first run *DetectLang* from [44], which automatically produces a set of languages believed to be contained in the document. This set of languages is used as input for both the CRF+GE and the HMM models.

5.5.3 Measures

The most natural way to evaluate each of these label-producing systems is with label-level **accuracy**. Accuracy, however, is an imperfect measure in this task, since it is possible to achieve high accuracy (89%) by simply applying the most likely label for the whole document to each word. On average, each document’s *majority class* actually covers more than 90% of the tokens. To measure how well minority classes are handled, we compute **minority class precision**, **minority class recall**, and **minority class F1-score** as follows.

Let m be the set of true minority class instances, and TP , FP , FN be the set of true positive, false positive, and false negative instances, respectively.

$$Precision = \frac{|TP \cap m|}{|(TP \cap m) \cup (FP \cap m)|} \quad (5.5.1)$$

$$Recall = \frac{|TP \cap m|}{|(TP \cap m) \cup FN|} \quad (5.5.2)$$

Here we still consider the distinction between minority classes, but we simply don't include performance on the majority class. Minority class F1-score is defined in terms of minority class precision and minority class recall in the normal way.

5.6 Discussion

It is clear that accuracy and precision/recall on the minority class are measuring different things, as the two baseline methods, *mixture of sentences* and *document classification* achieve the same accuracy, but have very different minority class F1-scores. The most accurate model was the basic mixture model with sentence-level granularity, which achieved 96.2% accuracy. This model also achieved the highest minority class F1-score, though the HMM-based model with sentence-level granularity was not far behind. Due to its speed (less than 10 seconds per document on average), we would recommend the use of the HMM-based model to any future researchers working on this same task.

As expected, the context of a word is an important clue in determining its correct language label. Sentence-based inference, in which the label of a word is constrained to match the

label of the other words in its sentence context, achieves both the best individual and best average values of the three types of inference compared for all four metrics. Additionally, the HMM-based model, in which a word’s label depends on the previous word in context, also achieves the highest average scores for minority class precision, recall, and F1-score of all the models compared. Surprisingly, we observe quite poor performance for the CRF+GE and HMM models when paired with *DetectLang* [44]. This seems to be due not to a poor selection of languages in the first stage, but due to the models ending up in bad local maxima when there are three or more languages in the set of selected labels.

By far, the most common type of error made by any of the methods explored in this chapter was to confuse two similar languages. For example, words in Czech were also given high probability of belonging to Slovak (a closely-related language) and often mislabeled as such. A sparse Dirichlet prior over θ was used by each of this chapter’s models to help prevent this type of error from occurring by preventing too many languages from co-occurring in the same document. In practice, though, the Gibbs sampling inference method does not allow a sparse prior to exert much influence in steering the samples toward a sparse θ . For example, with the basic mixture model, we see that as the Dirichlet concentration approaches zero,

$$\lim_{\alpha \rightarrow 0} P(z_i = \ell | \mathbf{z}_{-i}, \mathbf{w}) \propto \frac{n_{-i,\ell}}{n_{-i}} \beta(w_i, \ell).$$

In the extreme case where every $\alpha_\ell = 0$ (under this prior, every θ should assign all its probability mass to only one language, leading to only monolingual documents), sampling occurs with respect to the document’s current language distribution, and the Dirichlet prior does not even appear in the sampling formula. Because the priors seemed to be so ineffectual, we experimented with several types of initialization in the hopes of starting the sampler in a

point closer to the global maximum, and indeed we found that in many cases, initialization makes a much bigger difference in the final performance than do the priors.

In addition to the methods described in this chapter, we also experimented with other methods which either gave no benefit or led to poorer performance. The naïve Bayes classification method is very popular for language identification of documents and sentences, but when we applied it to word-level language identification, it led to very poor results across all models. We also experimented with using Lui et al.'s *DetectLang* method to narrow down the set of possible languages, but found that it did not improve performance, and actually hurt it slightly. Finally, we also tried another generative model that used a logistic-normal prior over the document-language distribution θ . In theory this should have allowed the model to learn which pairs of languages are (un-)likely to occur together. In practice, the logistic-normal prior required $O(|\mathcal{L}|^2)$ parameters, which was too many to learn reliably.

5.7 Conclusion

In this chapter we present several generative models for inferring the languages of words in mixed-language documents from a large set of possible languages. A basic multinomial mixture model using sentence-level inference, which makes use of a word's context to assign a language label, is the best-performing model among those compared at 96% accuracy. This model also achieves the best minority-class F1-score.

We believe that there is ample room for future work in this line of research. The models explored here, while encouraging in their accuracy, are rather naïve with respect to their rather unrealistic generative stories. The good performance of sentence-level inference and of the HMM-based model makes it clear that an accurate model must incorporate contextual

information. Future models may directly model long spans of single language text and other properties of real-world mixed-language documents, such as parallel sentences or code-switching.

Accurately identified language tags for words enables higher order NLP tasks by allowing appropriate models to be chosen, e.g. using an English POS tagger for English text, etc. The remaining chapters in this thesis consider these higher order tasks as part of a pipeline that has word-level language identification as its first stage. We consider the tasks of POS tagging and dependency parsing in a way that is appropriate for low-resource languages. While there are certainly many more tasks that we could consider, these are two foundational tasks that help to enable many others.

CHAPTER VI

Sharing Grammars Between Treebanks for Type-Supervised Part-of-Speech Tagging

6.1 Introduction

Recall that type-supervised part-of-speech (POS) tagging is the task of learning a POS tagger from a type dictionary (a list of words in the language and the possible tags they can take) and raw text [155]. This scenario is especially appropriate for the study of building POS taggers for low-resource languages, which may have a small dictionary, some raw text, and no other electronic resources. While token-supervision quite often allows for higher accuracy taggers than type supervision [95], it is often much easier to obtain some sort tagging dictionary for a low-resource language (possibly from a resource like Panlex) than it is to find even a small amount of annotated text. For their broad applicability, type-supervised tagging is worth considering in addition to other POS tagging paradigms.

In this chapter, we apply ideas from direct transfer learning to the task of POS tagging in a way that is general, i.e. does not require that the two languages be closely related as

previous work has [72–74]. By mapping tagsets from different treebanks to the universal part-of-speech tagset [89], we show that tagging grammars from high-resource languages can be used to build effective type-supervised POS taggers that beat the current state of the art in most cases.

We use the term “grammar” in a very broad sense, not necessarily referring to any type of constituent or dependency grammar, but simply meaning rules that determine the syntax of a language. This chapter focuses on bigram HMM taggers, and so when we use the term “tagging grammars,” we are referring to the tagger’s transition matrix, which controls what tags can transition to what other tags and what the probabilities of these different transitions are.

An inspection of various treebanks after mapping their parts-of-speech to the universal tagset shows that despite the differences in annotation schemes and the inherent differences between the languages, the underlying grammars are still surprisingly similar. Table 6.9 shows the most frequent bigrams in each of the Italian, Portuguese, and Bulgarian treebanks. It may not be surprising that Italian and Portuguese have similar lists, but Bulgarian, which is rather distantly related still shares three of the top five most frequent bigrams. The similarity of these bigram grammars suggests that some pairs of these grammars could be reasonably substituted for one another. At the very least, it seems that another language’s grammar could be a better starting point for optimization than a uniform distribution over all possible bigrams.

As a basis for our experiments in this chapter, we use the type-supervised POS tagging system described in [93], which is specifically designed for the low-resource type-supervised setting. The system includes three major stages: (1) tag dictionary expansion, (2) model minimization, and (3) EM tagger training. The system also included a fourth stage that

	Italian	Portuguese	Bulgarian
1.	(DET, NOUN)	(DET, NOUN)	(NOUN, .)
2.	(ADP, DET)	(ADP, DET)	(ADP, NOUN)
3.	(NOUN, ADP)	(NOUN, .)	(NOUN, ADP)
4.	(NOUN, .)	(NOUN, ADP)	(ADJ, NOUN)
5.	(ADP, NOUN)	(ADP, NOUN)	(PRON, VERB)

Table 6.9: The top five most frequent tag bigrams in the Italian, Portuguese, and Bulgarian treebanks after mapping to the universal POS tags.

Und	der	Protest	endet	wie	eine	Familienfeier	.
CONJ	PRON	NOUN	NOUN	CONJ	PRON	NOUN	.
	DET		VERB	PRON	DET	VERB	
			ADJ			ADJ	
			⋮			⋮	
			PRT			PRT	

Table 6.10: An example of type-supervised tagging in German. Below each word is listed the set of possible tags. The words “endet” and “Familienfeier” do not appear in the tagging dictionary and therefore can take any possible tag. The tags selected by this chapter’s tagging method are displayed in bold.

		Emission and evaluation treebank								
		bl	da	de	en	es	fi	it	pt	sv
Transition treebank	bl	0.935	0.890	0.766	0.819	0.810	0.800	0.752	0.788	0.876
	da	0.912	0.927	0.774	0.806	0.807	0.837	0.752	0.785	0.875
	de	0.915	0.898	0.903	<u>0.914</u>	0.889	0.827	0.896	0.916	<u>0.908</u>
	en	0.857	<u>0.908</u>	<u>0.900</u>	0.947	0.897	<u>0.844</u>	0.900	0.919	0.903
	es	0.909	0.873	0.855	0.878	0.912	0.784	<u>0.910</u>	<u>0.928</u>	0.890
	fi	0.906	0.900	0.787	0.816	0.801	0.854	0.697	0.778	0.862
	it	0.910	0.880	0.862	0.881	<u>0.909</u>	0.812	0.911	0.926	0.894
	pt	0.907	0.884	0.867	0.881	0.903	0.791	0.904	0.929	0.898
	sv	<u>0.926</u>	0.906	0.777	0.832	0.814	0.819	0.755	0.792	0.914
%age of supervised score possible with shared grammar		99.0	97.9	99.7	96.5	99.7	98.8	99.9	99.9	99.3

Table 6.11: POS tagging accuracies for second-order HMM taggers with emission tables learned from the target language and transition tables learned from a different language. Bolded numbers represent the highest accuracy using any source grammar. The underlined number in each column represents the highest accuracy using a grammar from a source language other than the target language.

trained a maximum entropy Markov model (MEMM) from the HMM, but we omitted this step as we found that it consistently yielded negative results in our scenario. Table 6.10 shows an example of type-supervised POS tagging and the output of this chapter’s system.

In this chapter, we show that type-supervised POS taggers can be learned for low resource languages by using grammars shared from other treebanks to guide the EM-based learning of an HMM POS tagger. The method can very easily be applied to a new language, needing only a dictionary of types with possible POS tags and access to tagged sentences in resource-rich languages. The contribution of our work beyond previous work is threefold. First while previous work has experimented with borrowing the transition matrix from the HMM tagger of a closely related language [72–74], we show how this can be applied even to languages that are not closely related. We show how grammars from multiple source languages can be combined to create a grammar that is even better for the target language than a single closely-related languages. Finally, we don’t simply insert the shared grammar into the target language HMM tagger, but use EM to refine it and tune it for the target language. While we make use of the system of Garrette et al. [93] to demonstrate these techniques, we do not consider this an extension of their work, but rather a technique that can easily be applied to any type-supervised POS tagger to improve the tagging accuracy by several percentage points on average.

The structure of the chapter is as follows. We start with an example to motivate this line of research in Section 6.2, describe the data used for experiments in Section 6.3, explain the various methods used in our pipeline in Section 6.4, show the experimental results in Section 6.5, discuss the results and future work in Section 6.6, and conclude in Section 6.7.

6.2 A Motivating Example

We begin with an example to motivate this line of research, showing that POS-tagging grammars can be easily shared between treebanks and can be used to build a tagger with an accuracy that is surprisingly close to the accuracy of a fully supervised tagger. For each of nine treebanks, we convert the labeled parts of speech to the universal part-of-speech tagset [89] and train a supervised second-order HMM on each. This model is compared against similar second-order HMM models where the emission table is retained, but the transition table learned from a different treebank is substituted.

Table 6.11 shows the results of this experiment. The supervised results appear in boldface along the diagonal of the table, while the other models appear off the diagonal. The column represents the target language treebank that is used for testing and for learning the emission table, and the row represents the treebank that is being used to learn the transition table.

Several trends are apparent from this table. First, every treebank seems to have at least one partner within the set of nine treebanks whose grammar can be substituted with very little additional error incurred. In most of the cases, the treebank can be parsed using a different grammar and still have $> 99\%$ of the accuracy of the fully supervised case.

Second, treebanks whose languages are closely related linguistically seem to have more compatible grammars. For example, the most compatible grammars for Italian, Portuguese, and Spanish (three Romance languages) are Spanish, Spanish, and Italian, respectively. The extent to which this holds true is limited however. Swedish and Danish are both North Germanic languages, but the most compatible grammars for each are English and German, respectively, both West Germanic languages.

Third, larger treebanks seem to have grammars that are compatible for more different lan-

guages. The German and English treebanks are the two largest among the nine treebanks and German or English are the most compatible grammars for five of the nine languages.

Though the experiment described here is not strictly type-supervised POS tagging (since the HMM has emission probabilities learned from target language sentences instead of from a tag dictionary), its conclusions are applicable to the type-supervised setting. Indeed, if we instead initialize the HMM’s emission table using a tag dictionary, we again find that in all cases, by using a different language’s grammar to learn the HMM’s transition table, we can achieve $> 95\%$ the accuracy that is possible using the target language’s grammar.

6.3 Data

In this section, we describe the data that will be used for the experiments in the remainder of the chapter.

6.3.1 Treebanks

We use nine treebanks for our experiments: the same that were used in Section 6.2: Bulgarian [156], Danish [157], English [158], Finnish [159], German [160], Italian [161], Portuguese [162], Spanish [163], and Swedish [164]. These languages were chosen to be a reasonably sized set of languages, over which results could easily be reported in a standard length research paper, to represent a variety of different types of languages, and to provide examples of compatible shared grammars where possible.

To simulate a low-resource POS tagging environment, we split each treebank in half, sampling 1000 sentences from the first half for tag dictionary creation and sampling 250 sentences from

the other half to serve as evaluation data. (For reasons that we discuss in Section 6.6, it is important to also downsample the testing data to match the low-resource scenario.) The tag dictionary is created by pairing each word that appeared in the 1000 sentences with the set of different tags that word received. On average, each such tag dictionary contained 5700 words, with an average of 1.01 different tags per type and 1.29 different tags per token.

6.3.2 Transferred Grammars

Section 6.2 described an experiment to determine which grammars from other treebanks are most compatible for POS-tagging a given treebank. But this is not the only way to select a grammar to transfer between treebanks. In this section, we will describe and evaluate a number of different methods for building grammars to transfer between treebanks. Although we downsample from the treebanks in the previous section in order to simulate a low-resource scenario for the language being tagged, when a grammar is shared from another language, we would like for that language to be a high-resource language. So we use complete treebanks to construct the shared grammars.

Ideally we would simply select the treebank from Section 6.2 that gives the highest accuracy and use its grammar. Unfortunately, this experiment used hundreds of sentences to determine the best treebank, which is an unrealistic amount¹¹.

Ethnologue If possible, we would like to find some external way to determine which pairs of languages are more likely to have compatible grammars. The **Ethnologue** [3] is an encyclopedia of language, one which includes the classifications of languages into nested

¹¹With access to hundreds of tagged sentences, fully supervised tagging (rather than type-supervised tagging) becomes the dominant paradigm

language families. For example, English is classified as “Indo-European > Germanic > West.” To find language most similar to a given language, we find the least common subsumer of the target language and another language in the classification tree. The resulting grammar is built from the concatenation of all other languages that share that same least common subsumer.

WALS The World Atlas of Language Structures (**WALS**) [165] is a resource that contains descriptions of many of the world’s languages. For each language, it enumerates a number of properties related to Morphology, Phonology, Lexicon, etc. Features based on word order, such as “order of adjective and noun,” and “order of subject, object, and verb,” are relevant to part of speech tagging. For each of the nine languages, we create a vector based on the possible values of each of the word order rules (rules 81–97 in Table 6.12) and compute the similarity between the vectors of the different languages. Since all the vectors are of the same length, similarity is proportional to the number of identical feature-value pairs between the two languages. Using this similarity function, we use the grammar of the most similar language, concatenating multiple languages in the case of a tie.

Pooled Based on the observation that grammars from larger treebanks seem to transfer better, we can take this idea one step further. We create as large a grammar as possible by concatenating together all eight of the other treebanks and using the grammar from the pooled treebank. This is similar to the technique that was used successfully in [70] to build delexicalized dependency parsers from other treebanks.

Linear Combination An alternative to simply concatenating together all the grammars is to create a weighted linear combination of the grammars to more closely match the grammar

Rule	Description	Rule	Description
81	Order of Subject, Object and Verb	89	Order of Numeral and Noun
82	Order of Subject and Verb	91	Order of Degree Word and Adjective
83	Order of Object and Verb	92	Position of Polar Question Particles
84	Order of Object, Oblique, and Verb	93	Position of Interrogative Phrases in Content Questions
85	Order of Adposition and Noun Phrase	94	Order of Adverbial Subordinator and Clause
87	Order of Adjective and Noun	95	Relationship between the Order of Object and Verb and the Order of Adposition and Noun Phrase
88	Order of Demonstrative and Noun	96	Relationship between the Order of Object and Verb and the Order of Relative Clause and Noun
87	Order of Numeral and Noun	97	Relationship between the Order of Object and Verb and the Order of Adjective and Noun
88	Order of Demonstrative and Noun		

Table 6.12: WALs word order rules for creating WALs language-similarity vectors.

of the target language. For this method, we allow ourselves access to five randomly selected tagged sentences in the target language (which would be quite trivial for a native speaker or linguist to produce). We then run a hill-climbing algorithm to tune the coefficients in a weighted linear combination of the other grammars to minimize the perplexity between the linear combination grammar and the grammar observed in the tagged sentences. If $p(x)$ is the probability of a bigram in the sample and $q(x)$ is the probability assigned to that bigram by the linear combination grammar, the perplexity is measured using the following equation:

$$2^{-\sum_x p(x)\log_2(q(x))}$$

Because the five tagged sentences are so sparse, representing only a small portion of the possible phenomena that occur, we expect that the linear combination of other treebanks will outperform the grammar learned from these five sentences alone (**5-sent supervised**).

Lowest Perplexity With access to five tagged sentences, we can also search for the single treebank that best fits the grammar observed in those sentences. This approach compares each language’s treebank’s grammar with the grammar from the five tagged sentences and selects the one with the lowest perplexity.

In Table 6.13, we evaluate each of these different methods for grammar selection. To do this, we train an HMM POS tagger using EM in the ordinary fashion, except that the initial transition probabilities for the HMM are learned from the external grammar. (See section 6.4.6 for more details)

Of the methods evaluated, the best performer was “linear combination”, the method of building a weighted linear combination of the other treebank grammars, though “lowest

Method	Average accuracy
Ethnologue	0.857
WALS	0.851
Pooled	0.867
Linear Combination	0.876
5-sent Supervised	0.863
Lowest Perplexity	0.872

Table 6.13: Average tagging accuracies when using different methods for creating shared grammars.

perplexity” was not far behind and would yield a decrease in accuracy of less than 1% based on this evaluation. If it is the case that it is infeasible to produce five tagged sentences, the method of pooling the other treebanks would also work quite well, with a decrease in accuracy of only slightly over 1%. The relatively poor performance of the WALS and Ethnologue methods may indicate that factors like similarity in annotation style or domain are more important factors than linguistic similarity in determining the transferability of the grammar. Throughout the rest of the chapter, we will use “linear combination” as the method for building shared grammars.

6.4 Methods

In this section, we describe the different components of a type-supervised POS tagging pipeline that we experiment with in this chapter. We must necessarily leave out some implementation details for methods that have already been thoroughly described in previous literature. More details on EM training, tag dictionary expansion, and model minimization can be found in [93].

6.4.1 HMM

As a baseline, we use an ordinary HMM with its transition probabilities learned from the external grammar. The emission table is built from the tag dictionary with the emission probability initialization process from [155], which assigns greater probability to tags that are known to be more open¹².

6.4.2 Tag Dictionary Expansion

Tag dictionary expansion addresses the issue of unknown words, which can take any of the possible tags and can pose a major issue in the low resource POS tagging setting. The implementation of tag dictionary expansion in this chapter is exactly as in [93], who create a graph with nodes for word types, word tokens, context tokens, and suffix features and use label propagation to learn distributions over possible tags for the various unknown tokens. Known word tokens that have only a single label in the tag dictionary are used to seed the graph. Using modified adsorption [166], a label propagation algorithm, distributions over the various tags are propagated to the unknown word tokens.

This process ends up giving every unknown word a distribution over all possible tags. To trim the set of possible tags for an unknown word, any tag probabilities below 0.1 are dropped and the remaining distribution is renormalized. If any unknown words did not have their tagset reduced by this process, they use the emission probability initialization scheme in [155].

The output of label propagation is an expanded tag dictionary with a reduced set of entries for each unknown word. For more details on tag dictionary expansion, please consult [93].

¹²The open-ness of a tag is related conceptually to open- and closed-class words. A tag that is more open is more likely to be associated many different word types.

Pipeline	bl	da	de	en	es	fi	it	pt	sv	avg
1. HMM	0.810	0.836	0.794	0.848	0.859	0.762	0.835	0.834	0.752	0.814
2. EM	<u>0.831</u>	0.816	0.712	0.855	0.857	0.680	0.644	0.722	0.632	0.750
3. SEM	0.814	0.824	<u>0.782</u>	0.845	0.853	0.667	<u>0.855</u>	<u>0.828</u>	<u>0.777</u>	<u>0.805</u>
4. EM + TDE	<u>0.888</u>	<u>0.860</u>	<u>0.826</u>	<u>0.921</u>	<u>0.899</u>	<u>0.756</u>	0.825	0.751	<u>0.814</u>	<u>0.828</u>
5. SEM + TDE	0.878	<u>0.867</u>	<u>0.859</u>	0.890	0.896	0.758	<u>0.895</u>	<u>0.886</u>	<u>0.831</u>	<u>0.863</u>
6. EM + TDE + MM	<u>0.888</u>	0.856	0.840	<u>0.922</u>	0.898	0.747	0.831	0.761	0.791	0.837
7. SEM + TDE + MM	0.879	<u>0.867</u>	<u>0.859</u>	0.902	0.897	0.755	<u>0.901</u>	<u>0.886</u>	<u>0.828</u>	<u>0.864</u>
8. SEM + TDE + GMM	0.879	0.867	0.860	0.901	0.897	0.756	0.901	0.883	0.830	0.864

Table 6.14: Tagging accuracies of different POS tagging pipelines. HMM is the hidden Markov model from Section 6.4.1; EM is expectation maximization from Section 6.4.5; SEM is seeded EM from Section 6.4.6; TDE is tag dictionary expansion from Section 6.4.2; MM is model minimization from Section 6.4.3; and GMM is guided model minimization from Section 6.4.4. The most accurate model for each column is listed in boldface. A single underline indicates that a value is statistically significantly greater than the corresponding value in the row directly above at $p < 0.05$. A double underline indicates statistical significance at $p < 0.01$.

6.4.3 Model Minimization

We use the model minimization implementation of [93], which is based off the greedy algorithm described in [85]. This algorithm has two stages, which both crucially rely on the concept of tag paths through a sentence. When a word has many possible tags, then there are many possible tag paths that pass through the word. For example if we have a short sentence “the can,” with “the” only allowed to take the tag DET and “can” allowed to take the tags “NOUN” or “VERB,” there are two paths through the sentence: DET \rightarrow NOUN or DET \rightarrow VERB. If the current model includes the bigram (DET, NOUN), but does not contain (DET, VERB), then the only valid path through the sentence is DET \rightarrow NOUN.

The first stage greedily selects tag bigrams for use in the minimized model by iteratively selecting the unused tag bigram that is able to cover the most new word tokens. A word is considered to be covered if there is some bigram that connects it either to the word before it or the word after it. After all word tokens have been covered by at least one bigram, there may still not be a valid path through a sentence, as a word needs only to connect to the word before *or* the word after it. When two consecutive words in a sentence are not connected by any bigram in the grammar, this is referred to as a *hole*.

The second stage greedily select bigrams to fill these holes. It iteratively selects the unused bigram that is able to fill the most holes. This second step is complete when there is a valid tag path through every sentence.

These two stages are modified in [93] by using the weights from tag dictionary expansion to pick not the bigrams that cover the most tokens or fill the most holes, but bigrams that cover/fill the greatest aggregate node weight. The output of model minimization is a noisily labeled corpus, the result of tagging the raw text using the minimized model.

6.4.4 Guided Model Minimization

A shared bigram grammar from another treebank is able to give us an idea of which bigrams might be most prevalent in this language’s grammar. We can use this information from the shared grammar to help guide the process of model minimization. In guided model minimization, we perform the same two steps from ordinary model minimization, but at each iteration, we limit the selection of tag bigrams only to the top- n most frequent unused bigrams from the shared external grammar. Experiments with development treebanks (not included in the nine treebanks of this chapter) suggest that 10 is a reasonable for value for n .

6.4.5 EM Training

We use EM to iteratively train an HMM starting from an initial HMM. When EM training is the first stage of the pipeline, this is similar to the EM training of [78], with the initial HMM as described earlier. The major differences from Merialdo’s work are the emission table initialization scheme using one-count smoothing for unseen words [167].

If EM is not the first stage of the pipeline, the previous stages are used to produce a noisy labeling of the raw text. This noisy labeling along with the tag dictionary (which may have also been modified in previous stages) is then used to initialize the HMM, which is refined using EM.

6.4.6 Seeded EM Training

Seeded EM training uses an external shared grammar to initialize the transition probabilities of the HMM. If this is not the first stage of the pipeline, the grammar from the noisily labeled raw text is concatenated with the external shared grammar to initialize the transition probabilities. EM training or seeded EM training is always the final stage of the pipeline.

6.5 Results

The average accuracies of various POS tagging pipelines are reported in Table 6.14. Averaged over all the treebanks, every pipeline that included seeded EM outperformed its counterpart using ordinary EM with statistical significance at $p < 0.01$. Though some other methods worked best for individual treebanks, the best pipeline over all the treebanks was seeded EM + tag dictionary expansion + guided model minimization.

Since POS tags are used as input for many different types of applications, it is important to test the usefulness of automatically generated POS tags in a downstream application. Though accuracy is an important measure, it may not correlate perfectly with performance in another application, as certain types of errors may be more costly than others in certain applications [168].

We use multi-source delexicalized dependency parsing as a downstream application [70]. This fits nicely in the low resource setting, as automatic POS tags are necessary in order to perform any real-world parsing in this framework. Figure 6.5.1 compares the unlabeled attachment scores (UAS) of a delexicalized dependency parser trained on a 5000 sentence treebank sampled from the concatenation of the eight other treebanks. The type-supervised tags are

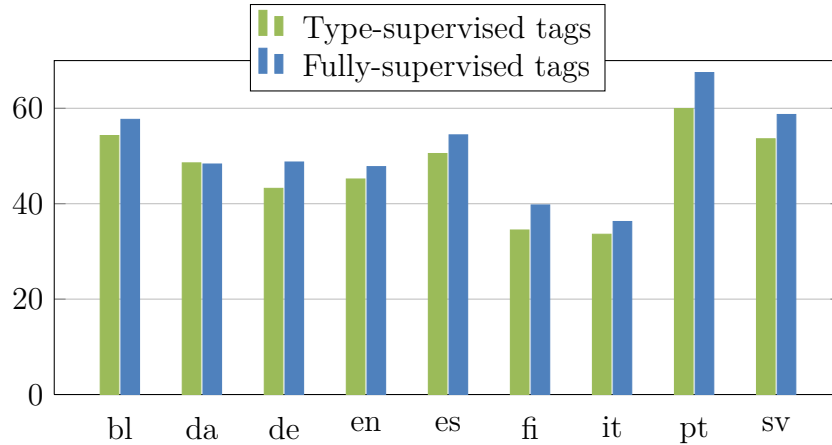


Figure 6.5.1: Unlabeled attachment scores (UAS) in percentage for a multi-source delexicalized dependency parser trained on the type-supervised predicted tags and the fully-supervised predicted tags for each language.

generated with the pipeline that includes seeded EM training, tag dictionary expansion, and guided model minimization.

The type supervised tags are compared against tags produced by a fully-supervised tagger. To build this tagger, we take the 1000 sentences used to create the type dictionary and instead use them to train a supervised POS tagger [169]. Recall that the sets of 250 gold-tagged sentences used earlier as evaluation came from dependency treebanks. We substitute the type-supervised and fully-supervised tags for the gold tags of these sentences and use these as input for a delexicalized MSTParser [170], and evaluate the predicted arcs against the CoNLL treebank annotations. Dependency parsers generally see an increase in error of between 10% and 15% when using predicted tags instead of gold tags.

The UAS for the parser run with type-supervised tags is approximately 92% of the score for fully-supervised tags on average. These results would suggest that POS tags generated using shared grammars should be appropriate for use in a real-world low-resource parsing setting, as a parser using tags produced by this method works nearly as well as a parser using tags

produced by a fully-supervised tagger.

6.6 Discussion and Future Work

We believe there is further work to be done in building shared tag bigram grammars from multiple sources. The method of pooling all eight other treebanks to create a grammar worked surprisingly well, suggesting that there is some sort of shared universal structure inherent in all the languages we tested here. While this could be due to the fact that most of the languages are Indo-European, the pooling method actually worked quite well for Finnish (the only Uralic language in the set). We would suggest that the universal functions of tags naturally make some tag bigrams more or less likely in almost any language. For example, (ADP, NOUN) is a common bigram in both prepositional and postpositional languages, whereas (ADP, X) is uncommon in any of the nine treebanks.

Guided model minimization gave a slight improvement over ordinary model minimization. The greedy minimization algorithm already does a pretty good job of picking reasonable tag bigrams. In most cases, the bigram selected from the set of 10 tags provided by the guided algorithm matched the bigram selected from the full set of available bigrams.

We observed a general trend that model minimization tends to work best when given a large corpus of unlabeled text, as the distributions in a large corpus are usually not as noisy as those observed in a smaller corpus. Indeed when run on a large corpus like the full Penn Treebank, model minimization can provide a slightly better initialization for EM than a shared grammar can, leading to a more accurate tagger. Because of the semi-supervised nature of this problem, the amount of unlabeled data makes a difference in the sorts of methods that are appropriate. We would suggest that initializing EM using a shared grammar is most ben-

eficial when working with an unlabeled corpora of no more than a few thousand sentences. As the unlabeled corpus grows beyond that size, the performance of model minimization improves and the relative benefit of EM with a shared grammar narrows.

Previous work has noted that when EM is used on a second-order HMM for POS tagging, it seems to handle its additional degrees of freedom poorly, finding even worse solutions than in the first-order setting [84]. By seeding a second-order model with a trigram grammar shared from a different treebank, we may be able help guide the EM process for second-order models to build type-supervised POS taggers that are able to outperform first-order models.

6.7 Conclusion

We presented a simple approach for sharing grammars between treebanks in order to learn more accurate type-supervised POS taggers for low resource languages. After mapping the treebank-specific tagsets to the universal POS tagset, many of the languages had quite similar grammars, similar enough in fact, that in a supervised setting they could often be substituted for one another with almost no loss in performance. We used bigram grammars shared from different treebanks to create a seed HMM that serves as a better starting point for EM-based learning of a POS tagger. Experiments were performed using a state-of-the-art type-supervised POS tagging system. This method of seeding the HMM was able to improve tagging performance in all cases, and is a simple method that should be easily usable by other researchers building type-supervised POS taggers.

While the methods described here are broadly applicable, requiring only a tagging dictionary and raw text, the next chapter describes a method that assumes slightly more in the way of available resources (also requiring bitext), but gives a much greater gain over the methods

discussed here. As we see in Section 6.5, using automatically predicted tags in a pipeline setting will generally increase the error in the next stage by 10-15%. In Chapter VIII, when we consider the problem of low-resource dependency parsing, we use gold POS tags to be consistent with previous work, but we expect that using automatically predicted tags would incur a penalty similar to what is seen in this chapter.

CHAPTER VII

Multi-Source, Multi-Target Cross-lingual Part-of-Speech Tag Projection

7.1 Introduction

Recent work in cross-lingual part-of-speech (POS) tagging has focused on combining evidence from token-level annotations, like those produced by projection across parallel text, and type-level annotations, such as might be found in a dictionary [90,91]. We use the Posterior Regularization (PR) framework, which allows the model to flexibly incorporate information from these two types of sources, using the dictionary to prune the search space and the projections to create soft constraints on the tagging.

Whereas previous work has focused on projection from a single source language onto a single target language, we expand the problem to include joint projection with multiple source languages and multiple target languages. This setup adds additional constraints, preferring agreement not only between the target language and source language, but between all pairs of source and target languages and even between pairs of target languages. Figure 7.1.1

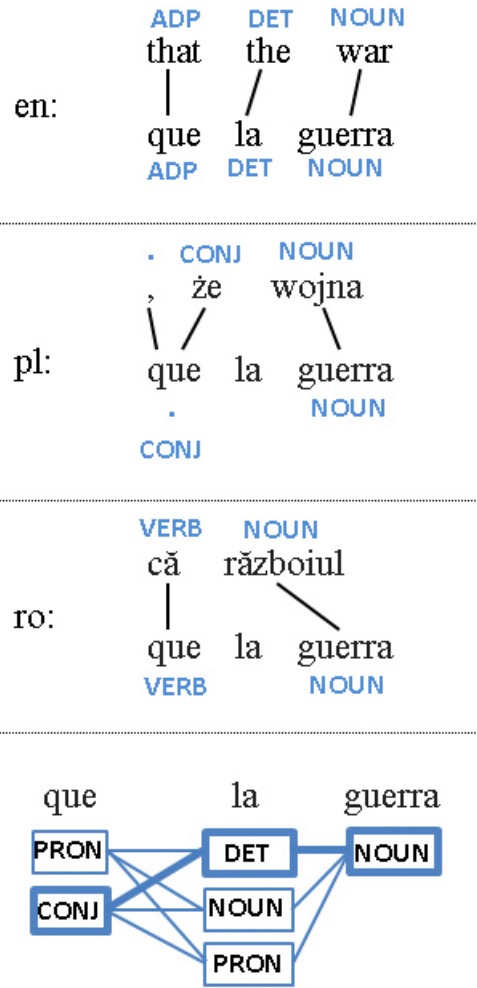


Figure 7.1.1: Examples of POS projection from three different source languages (English, Romanian, and Polish, respectively) onto the same Spanish text. Errors in tagging and word alignment, along with syntactic differences between languages mean that no single target language is able to correctly tag the text. Shown below them is the lattice allowed by the Spanish tagging dictionary with the correct path highlighted. By combining information from all these sources, the Spanish text can be correctly tagged.

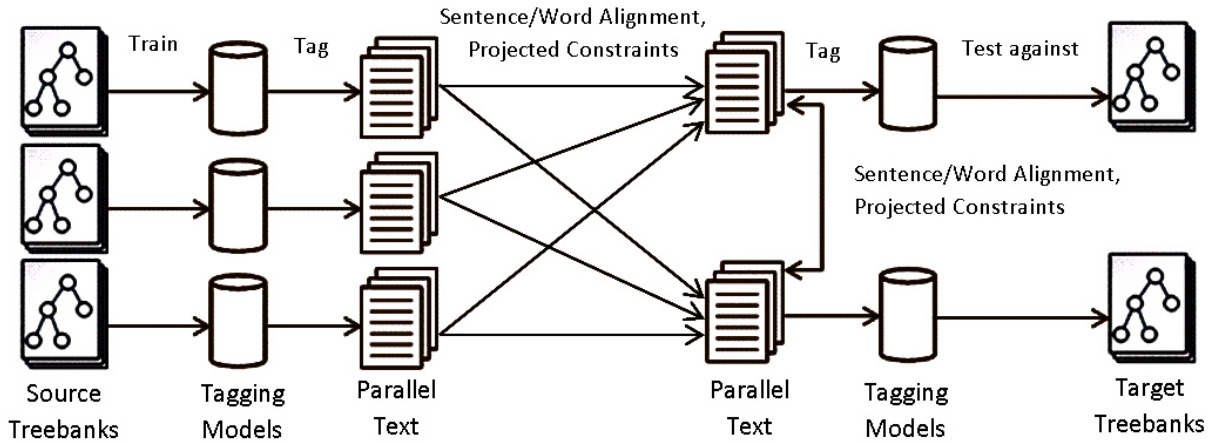


Figure 7.2.1: A diagram of information flow in this chapter’s system. This diagram shows three source languages and two target languages, but the method is applicable to any number of source and target languages.

shows an example of tag projection onto Spanish text. In order to correctly tag the text, information must be combined from multiple source languages and the tag dictionary.

We describe our model and learning algorithm in Section 7.2 and our data in Section 7.3. We report on experiments and results in Section 7.4, including a simulated low-resource experiment in Section 7.4.1. Finally we discuss our results, especially how they might apply in a real low-resource scenario in Section 7.5, and finally draw conclusions in Section 7.6.

7.2 Methods

This section describes our methods, starting with an overview of our approach. We then give a detailed description of the cross-lingual learning model and describe a baseline approach that we will compare against.

7.2.1 Overview

Our approach uses data from many different languages simultaneously, making a distinction between “source” languages and “target” languages. Source languages here are languages for which we are already able to train a supervised tagger – we do not learn anything further for these languages. Target languages are languages for which we want to train a tagger. This approach also utilizes a massively parallel corpus, a corpus in which the same text is translated into many different languages.

Figure 7.2.1 shows the flow of information through our system. We start with treebanks in the source languages, using them to train supervised taggers. These supervised taggers then tag the appropriate section of the massively parallel corpus, e.g. the English tagger is used to tag the English side of the corpus. For each pair of source language s and target language t , we perform sentence- and word-alignment between $corpus(s)$ and $corpus(t)$. The $s \rightarrow t$ word alignments are used to create constraints on the tagging that the model produces for t , preferring for a word in t to receive the same tag as its analogue is assigned in s .

We also perform the same type of alignment between each pair of target languages t and t' , setting up constraints that prefer that words aligned to each other in t and t' receive the same tag. Running our learning method produces a tagging model for each of the target languages. These tagging models are evaluated against treebanks in the target language.

We use four different terms for different variations of this model. **Single-source single-target** means that we use only one source language (English) and one target language. **Multi-source single-target** means that we project from multiple source languages (Bulgarian, Czech, English, French, Polish, Romanian, and Slovak) but a single target language with constraints preferring agreement with each of the source languages. In single-target

experiments, we build a tagger for each target language separately. **Single-source multi-target** means that only English is used as a source language. Each target language has constraints preferring agreement with English, but also has constraints preferring agreement with each of the other target languages. **Multi-source multi-target** uses all languages and constraints simultaneously. Multi-target experiments build taggers for all the target languages simultaneously.

7.2.2 Learning Model

Our model for weakly-supervised tagging in the target language is a linear chain conditional random fields (CRF) model trained using posterior regularization (PR). Recall that PR is a framework for learning weakly-supervised statistical machine learning models that constrains the space of possible posterior distributions for the model. In the case of tagging, this means putting constraints (e.g. each sentence must have at least one verb) on the tags produced by the model [171].

The objective function under PR with no labeled data is

$$\max_{\theta} -\mathbf{KL}(\mathcal{Q}||p_{\theta}(\mathbf{Y}|\mathbf{X})), \quad (7.2.1)$$

where $\mathbf{KL}(\mathcal{Q}||p) = \min_{q \in \mathcal{Q}} \mathbf{KL}(q||p)$ and p_{θ} represents the probability function of a linear chain CRF (see equation 2.3.2). This can be expressed in its dual form,

$$\max_{\theta} \min_{\lambda \geq 0} \mathbf{b} \cdot \lambda + \log \sum_{\mathbf{Y}} p_{\theta}(\mathbf{Y}|\mathbf{X}) e^{-\lambda \cdot \phi(\mathbf{X}, \mathbf{Y})}, \quad (7.2.2)$$

and optimized using stochastic gradient ascent. The gradient with respect to λ is $\mathbf{b} -$

$\mathbf{E}_{q_\lambda}[\mathbf{f}(\mathbf{X}, \mathbf{Y})]$ and the gradient with respect to θ is $\mathbf{E}_{q_\lambda}[\mathbf{f}(\mathbf{X}, \mathbf{Y})] - \mathbf{E}_{p_\theta}[\mathbf{f}(\mathbf{X}, \mathbf{Y})]$. We use a step size of 1.0 in all our experiments.

Single Source Projection In our application of this model to POS projection from a single source language s , to a single target language t , we use just one constraint feature function $\phi(\mathbf{X}, \mathbf{Y})$. This function takes a value of 1 if either (1) x_i 's projected tag from s matches its assigned tag in t and is allowed by the dictionary or (2) x_i is unaligned and its assigned tag is allowed by the dictionary. Otherwise ϕ is 0. This formulation is equivalent to the model presented in [91].

Multi-Source Projection The constraint functions of single-source projection naturally generalize to a setting using multiple source languages $s_1 \dots s_k$. In this case, we create an analogous constraint function for each pairing of a source language s and the target language t , with each token having as many as k constraints.

Multi-target Projection In addition to pairings between source languages and target languages, we also create constraint functions to enforce agreement between pairs of target languages t and t' . Broadly these take the same form as all the other ϕ constraints, except that unlike the source languages, the target languages have no single correct tagging. So to project from t' onto t and create a constraint, we use the current tags in t' (which can change from iteration to iteration) as the projected tags.

In single-target projection experiments, we build a tagger for each target language separately, but for multi-target experiments, all target languages are trained simultaneously. Because it can take a few iterations to arrive at a tagging that agrees well with the source-language

constraints, we train for 10 iterations before we add the constraints between target languages. This helps to ensure that the target-to-target constraints are propagating a more meaningful signal rather than simply noise.

7.2.3 Baseline: Pruning the Lattice

Täckström et al. [90] present a similar method for cross-lingual tag projection, which instead of using soft constraints on the CRF objective function, creates hard constraints by directly pruning the CRF lattice. The lattice is first pruned by removing the tags for each word that are not allowed by the tag dictionary. Then if a word’s projected tag is allowed by the dictionary, all tags but the projected tag are pruned. Otherwise, no further pruning takes place. This method allows the model to make use of both type-level knowledge from the tag dictionary and token-level knowledge from the projected tags, but minimizes projections of incorrect tags, which can result from syntactic divergences between the two languages.

If we let $\mathcal{Y}(\mathbf{x})$ be the first dictionary pruned lattice and $\tilde{\mathcal{Y}}(\mathbf{x})$ be the lattice further pruned by projection, CRF learning proceeds by maximizing the log-likelihood over each instance \mathbf{x} , which is defined to be

$$\log \sum_{\mathbf{y} \in \tilde{\mathcal{Y}}(\mathbf{x})} \theta \cdot \mathbf{f}(\mathbf{x}, \mathbf{y}) - \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} \theta \cdot \mathbf{f}(\mathbf{x}, \mathbf{y}). \quad (7.2.3)$$

Both the first and second terms of this equation can be efficiently computed using the forward-backward algorithm on the CRF lattice. As in [90], we optimize using L-BFGS [172].

7.2.4 Features

The models in the chapter use standard features that have been demonstrated as effective in previous work. For each word x_i , we use a bias feature, the full word x_i , suffixes up to length 3, whether it starts with a capital letter, whether it contains a digit, hyphen, or other punctuation, and its word cluster. Word cluster features are based on Brown clustering [96].

We also use features that make use of the word’s context, using a conjunction of the current word with the previous word, and a conjunction of the current word’s and previous word’s clusters.

To calculate clusters, we use Percy Liang’s implementation of Brown clustering [173] on approximately 400,000 sentences from Europarl in each language, creating 256 clusters. This differs from previous work, which has used significantly larger corpora for calculating clusters.

7.3 Data

7.3.1 Parallel Corpus

We use Europarl [119] as our massively parallel corpus. For each pair of languages, we perform automatic sentence alignment [174] and use the Berkeley aligner [175] to align the texts at the word level. To choose an appropriate subset, we first select the approximately 400,000 sentences in the corpus that are represented in the most languages. We then order sentences by the average proportion of tokens that are in a 1-to-1 alignment for each pairing, greedily selecting sentences until we reach 500,000 tokens.

The Bulgarian, Czech, English, French, Polish, Romanian, and Slovak sections of each corpus are treated as source languages and are automatically tagged using the MATE tools tagger [169] trained on the appropriate treebank. As in previous work, we use Danish, German, Greek, Spanish, Italian, Portuguese and Swedish as target languages.

7.3.2 Tagging Dictionaries

We use dictionaries derived from Wiktionary, a community-curated dictionary that covers a large number of languages. Since Wiktionary changes over time, dictionaries derived from it can vary based on when they were created. We use the dictionaries extracted by Li et al. [87].

7.3.3 Training and Testing Data

In this experimental setting, we need both training data to train taggers for the source languages and testing data to evaluate target language tagging performance. For both, we use treebanks from CoNLL X and CoNLL 2007 [29, 30], which provide tokenized sentences with POS tags. We also use the French treebank from Abeillé et al. [176], the Skladnica Polish treebank [177], and the Romanian Dependency Treebank [178]. These treebanks use their custom tagsets appropriate for their respective languages, but these tagsets are not compatible with one another. To allow projection of tags across languages, we convert the tags for each treebank to use a common tagset, the set of universal POS tags [89].

Language	Täckström et al.	Single source, single target	Multi source, single target	Single source, multi target	Multi source, Multi target
Danish	77.67	82.55	85.13*	82.64	83.37*
Dutch	84.28	83.92	85.25*	84.05	84.35
German	88.16	88.57	90.45*	88.84	90.02*
Greek	87.57	87.12	88.82*	86.70	87.01
Italian	86.73*	86.17	87.75*	85.82	84.54
Portuguese	84.71	88.19	88.31	82.19	86.69
Spanish	87.37	87.45	89.14*	86.93	87.72
Swedish	80.43	80.29	83.03*	82.43*	82.37*
<i>Average</i>	84.62	85.53	87.23*	84.95	85.76

Table 7.15: Accuracies of this chapter’s methods on each of the target languages. Bolded items represent the highest achieved accuracy for each language. A * indicates that an entry is statistically significantly better than the single-source single-target entry with $p < 0.01$.

7.4 Experiments and Results

The PR model with relaxed constraints has one parameter that must be tuned, the vector \mathbf{b} , which represents the strengths of the constraints, controlling how often the model uses the projected tag. If $\mathbf{b} = 0.3$, the model has the flexibility to ignore the projected tag (possibly including erroneous projected tags) 30% of the time in expectation, while a value of $\mathbf{b} = 0$ means that the model should, in expectation, never deviate from the projected tag.

In the case of single-source, single-target tagging, Ganchev and Das [91] found a strong relationship between the optimal value for \mathbf{b} and the average number of tags per token (TpT) allowed by the dictionary in the target language corpus. In fact, the average TpT for a corpus could be used to choose a good value for \mathbf{b} . In experiments using single-target projection, we hold out seven of the eight target languages and use them to learn a model

of \mathbf{b} and TpT, using this automatically-learned model to select a value of \mathbf{b} for the eighth language.

In experiments with multi-target projection, taggers for all target languages are being learned simultaneously, and we're not able to hold languages out. In these experiments, we use a uniform value of $\mathbf{b} = 0.1$ for all target languages.

Results of tag projection with the five methods mentioned are presented in Table 7.15. Across all eight languages, multi-source single-target projection produced the highest accuracy tagger of the methods compared, beating the previous state of the art (single-source, single-target) by a statistically significant margin in almost all cases.

7.4.1 Low-Resource Experiments

Techniques like cross-lingual tag projection have their most obvious application in building NLP tools for low-resource languages that might not have the POS-annotated text needed to train a supervised tagger. For most such languages it would be unreasonable to assume access to massively parallel corpus the size and quality of Europarl. To simulate a low-resource scenario, we use Bible translations for our massively parallel corpus. A complete translation of the Bible has approximately 30,000 sentences, which roughly matches the size of the Europarl sample used earlier. Complete translations are available for many languages, but some languages have only the New Testament portion translated, resulting in about 10,000 sentences. As before, we perform automatic sentence alignment and word alignment on each language pair. We also use the text of the target language Bible to create Brown clusters.

Many different translations of the Bible are often available, even within a single language

Language	Täckström	Single source, single target	Multi source, single target	Single source, multi target	Multi source, Multi target
Danish	77.12	78.90	80.35*	80.81*	79.31
Dutch	83.51*	82.29	83.26*	82.47	81.33
German	87.88	88.01	88.87*	86.31	88.46
Greek	80.56	80.27	83.85*	83.12*	82.37*
Italian	84.74	85.67	83.44	84.17	81.28
Portuguese	83.31	84.63	84.86	84.95	84.61
Spanish	85.48*	85.06	87.19*	84.88	82.37
Swedish	79.65	80.82	82.04*	81.74*	82.14*
<i>Average</i>	82.78	83.20	84.23*	83.56*	83.27
<i>Europarl Average</i>	84.62	85.53	87.23*	84.95	85.76

Table 7.16: Accuracies of this chapter’s methods on the various languages when applied to the low-resource setting using the Bible for parallel text. Bolded items represent the highest achieved accuracy for each language. A * indicates that an entry is statistically significantly better than the single-source single-target entry with $p < 0.01$.

Language	Bible	Language	Bible
Bulgarian	Veren Bulgarian Bible	Italian	Nuova Riveduta 2006
Czech	Český Studijní Překlad	Polish	NBG Vertaling
Danish	Bibelen på hverdagsdansk	Portuguese	A Biblia para Todos
Dutch	Die Nieuwe Bijbelvertaling	Romanian	Fidela 2013
English	English Standard Version	Slovenian	Jubilejni prevod Nove zaveze†
French	Nouvelle Bible Segond	Spanish	La Biblia Reina-Valera
German	Elberfelder Bibel	Swedish	Svenska Folkbibeln
Greek	Filos Pergamos		

Table 7.17: Translations of the Bible used for each language. Entries marked with † contain only a new testament translation.

(though this is less prevalent in low-resource languages). The number of translations available ranged from one in Slovenian to 40 in English. We automatically selected the translations used for each language by using a hill-climbing algorithm designed to maximize the number of 1-to-1 sentence alignments. The translations selected for each language are shown in Table 7.17.

Table 7.16 presents the results of the same experiment performed in a low-resource setting, using text of the Bible for a parallel corpus. Multi-source single-target projection still produces the best results on average, but the patterns from Table 7.15 are not nearly as pronounced.

7.5 Discussion

In terms of accuracy, either of the multi-source models (both single- and multi-target) represent a consistent improvement over the single-source, single target setup. We would recommend the multi-source, single-target setup to future researchers working on this task. This model is not only the most accurate among those tested, but also substantially faster to train than models with constraints between multiple target languages.

Compared to the large improvement that multiple source languages offered over a single source, the change to multiple target languages yielded relatively little improvement, and in fact hurt accuracy in some cases. We had waited until 10 iterations to start including constraints between target languages in order to avoid noise, but it seems that one clearer signal from a source language can be better than many noisy signals from other target languages.

Tagging accuracy was considerably lower in the simulated low-resource setting. There are

several factors that could contribute to this, including the Brown clusters (calculated over a smaller corpus), the fact that poorly-aligned Bible sentences are not filtered, and domain differences between scriptural text and the testing treebanks. We ran a set of experiments, varying the mentioned parameters and holding the rest constant, to determine the cause of the lower performance. The source of the Brown clusters makes very little difference, as does the filtering of alignments¹³. The only factor that led to any change in performance was the domain of the parallel text.

The domain of text is an issue that is often discussed in supervised settings [179–181], but not often in cross-lingual settings. The treebanks used for testing in this chapter’s target languages are built mostly from newspaper text, which is obviously much closer in nature to political speech as found in Europarl than to scriptural text. One of the treebanks (Greek) even includes tagged sentences from Europarl.

Based on our results, we believe that in any practical use of cross-lingual learning techniques, the domain of the tool’s application must be considered. For low-resource languages that may not have any parallel text outside of the Bible, multi-source and multi-target learning using Bible text can be used to create taggers with reasonable accuracies. However, when comparing the results of Table 7.16 with Table 7.15, it becomes clear that in-domain parallel text for even a single language pair can be better than an out-of-domain text covering many language pairs.

Some have argued that access to a tag dictionary for a low-resource language is an unrealistic assumption to make [95], as Wiktionary has much poorer coverage for low-resource languages like Hausa than for the languages evaluated in this chapter. Wiktionary however, is not the

¹³In the single-source, single-target setup, small gains were possible when selecting only the n best-aligned sentences. Settings with multiple source or target languages did not show any improvement. It seems that sentences that are aligned well in one language pair may be quite poorly aligned in other pairs.

Error type	Proportion
Dictionary has word, has tag	25.91%
Dictionary has word, missing tag	25.06%
Dictionary missing word, seen in training	14.72%
Dictionary missing word, unseen in training	34.31%

Table 7.18: The prevalence of four types of errors in the Europarl experiments. Rates of each type of error are quite similar in the Bible experiments.

only source for tagging dictionaries. Using Panlex [125], for example it’s possible to build tagging dictionaries for over 500 languages.

There are several types of errors that models using type and token knowledge can make. The prevalence of these different types of errors help illuminate which lines of research would be most productive to further reduce the error. Table 7.18 shows the proportions of each type of error. We divide the errors according to whether or not the tagging dictionary contains the mistagged word. If the dictionary has both the word and lists the correct tag as an option, then it is the projection model that failed and further research is needed to help it choose the correct tag. If the dictionary has the word, but not the correct tag, then the model in this chapter could not have possibly selected the right tag – research toward improving the tagging dictionary would be necessary.

If the tagging dictionary does not contain the mistagged word, then the only hope of correctly tagging it comes from using its context and emission features. Either more training data (i.e. a massively parallel corpus with more text) would be needed to better estimate feature weights or better features that correlate more strongly with the emitted tag would be needed. Based on the numbers in Table 7.18, there is room for improvement along any of these dimensions.

On the subject of features, it is worth noting that re-implementations of previous methods here achieve lower accuracies than are reported in the original works. The only difference between our implementations and the original implementations is in the Brown cluster features. In both [90] and [91], the Brown word clusters are described as being calculated by the method of Uszkoreit and Brants [182], presumably over very large corpora, likely larger than would be available for most low-resource languages.

In this chapter, we calculated the word clusters from Europarl (and from Bible text in the low-resource experiments). Because these clusters are more representative of low-resource settings, we believe that the results reported here are more in line with the accuracies that could be expected in a real application of these methods.

7.6 Conclusion

Previous work in cross-lingual projected learning has often run into problems with erroneously projected tags, whether from source-side tagging errors, word alignment errors, or simply from typological differences between the languages. By expanding this projection to multiple source languages and multiple simultaneous target languages, we improve redundancy and increase the range of syntactic phenomena, reducing errors. This allows us to make a statistically significant improvement on state of the art in cross-lingual POS tagging.

POS tags enable many higher level tasks in NLP. Examples of tasks that are often formulated to rely on POS tags include relation extraction, coreference resolution, semantic role labeling, and parsing. In the next chapter, we look at the task of dependency parsing low-resource languages. Such a system could conceivably be the next stage after POS tagging in a low-resource NLP pipeline.

CHAPTER VIII

Multi-Source Cross-lingual Dependency Parser Projection

8.1 Introduction

Just as POS projection systems had not made wide use of multiple source languages, dependency projection systems have also used only a single source language. With dependency parsing, syntactic mismatches and other sources of propagated error seem to be even more severe than with POS tagging. While projecting tags most often involves a tag for a single word being projected across a single word alignment pair onto a single word, the fundamental unit of dependency parsing is the edge, which involves two source language words, two word alignment pairs, and two target language words. This means that an error in one source language word can affect two arcs in the resulting parse tree. For this reason, multiple source languages should be especially beneficial for projected dependency parsing. As in the previous chapter, our approach requires having the same text parallel across all the source-target language pairs, so that annotations from multiple source languages can be

projected onto the same target language sentence, again making use of Europarl [119].

8.2 Methods

Recall that a conditional random field (CRF) parser models the probability of a parse tree \mathbf{y} given an input sentence \mathbf{x} as follows:

$$p_{\theta}(\mathbf{y}|\mathbf{x}) = \frac{\exp(\theta \cdot f(\mathbf{y}, \mathbf{x}))}{\sum_{\mathbf{y}' \in T(\mathbf{x})} \exp(\theta \cdot f(\mathbf{y}', \mathbf{x}))}, \quad (8.2.1)$$

where θ is the model’s parameter vector and $f(\mathbf{y}, \mathbf{x})$ is a feature function that converts a parse tree and sentence into a real feature vector. As in previous work [112], we assume that the tree’s probability factors over individual edges and denote the model’s weight function for each edge as follows:

$$w(e, \mathbf{x}) = \exp(\theta \cdot f(e, \mathbf{x})). \quad (8.2.2)$$

8.2.1 Single-source projected training

Smith and Eisner [183] propose a semi- and weakly-supervised training method for dependency parsers that reduces the need for labeled data. It does so by training over a set containing both weakly-labeled instances and unlabeled instances. In the case of cross-lingual projected parsing, our two sets are \mathbf{P} , a set of unlabeled instances with parallel text, and \mathbf{U} , a set of unlabeled instances with no parallel text. Smith and Eisner’s method maximizes *expected* log-likelihood of the model over both sets of instances:

$$\max_{\theta} \sum_{\mathbf{x} \in \mathbf{X}} \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} \tilde{p}(\mathbf{y}|\mathbf{x}) \log p_{\theta}(\mathbf{y}|\mathbf{x}), \quad (8.2.3)$$

where $p_{\theta}(\mathbf{y}|\mathbf{x})$ is the probability assigned to a parse \mathbf{y} of \mathbf{x} under the current target language model. $\tilde{p}(\mathbf{y}|\mathbf{x})$ is the probability under a *transferring distribution*, which in our case represents the probability of a source language model projected across the alignments. The authors show that under reasonable approximations¹⁴ [183], equation 8.2.3 reduces to

$$\max_{\theta} \sum_{\mathbf{x} \in \mathbf{P}} \tilde{p}(\mathbf{y}|\mathbf{x}) \log p_{\theta}(\mathbf{y}|\mathbf{x}) + \sum_{\mathbf{x} \in \mathbf{U}} H(p_{\theta, \mathbf{x}}). \quad (8.2.4)$$

This form yields some insights about the training method’s relationship to previous bootstrapping formulations. Specifically it is a generalization of Abney’s K objective function for bootstrapping [184], which is itself a generalization of Yarowsky’s methods [185]. Whereas Abney’s objective function regularizes over unlabeled examples using a uniform distribution over labelings, this method instead regularizes using cross-entropy, which has a strong correlation with accuracy, naturally assigning greater weights to unlabeled examples with higher confidence [183].

We let E represent our source language and F represent our target language. The parallel set $P = \{(\mathbf{x}^E, \mathbf{x}^F, a)\}$ is a set of target language instances x^F , each associated with a source language instance x^E , connected via a word alignment a . The transferring distribution \tilde{p} is defined in terms of a *transferring weight function*, \tilde{w}_E , as follows:

$$\tilde{p}(\mathbf{y}|\mathbf{x}) = \frac{\prod_{e \in \mathbf{y}} \tilde{w}_E(e, \mathbf{x})}{\sum_{\mathbf{y}'} \prod_{e \in \mathbf{y}'} \tilde{w}_E(e, \mathbf{x})}. \quad (8.2.5)$$

¹⁴specifically, that the transferring distribution $\tilde{p}(\mathbf{y}|\mathbf{x})$ for instances in \mathbf{U} is equal to $p_{\theta}(\mathbf{y}|\mathbf{x})$

Because the source language represents a language with adequate resources to train a dependency parser, we assume that the source language has two trained models, a standard lexicalized CRF parsing model p_{λ_E} and a delexicalized CRF model $p_{\lambda_E}^{delex}$ [70]. A common problem with projected parsing is that some target language edges do not align 1-to-1 with any source language edge. We define the transferring weight function by projecting edges when they are aligned and falling back to the delexicalized model otherwise:

$$\tilde{w}_E(e^F, \mathbf{x}^F) = \begin{cases} \theta_E \cdot f(e^E, \mathbf{x}^E) & \text{if } a(e^E, e^F) \\ \theta_E^{delex} \cdot f(e^F, \mathbf{x}^E) & \text{otherwise.} \end{cases} \quad (8.2.6)$$

8.2.2 Multi-source projected training

Because of syntactic mismatches between the source and target language as well as various errors in alignment and parsing, it may be desirable to project from multiple source languages simultaneously onto the same target language. As with multi-source POS projection, this requires a massively parallel corpus, with the same language translated into many languages and alignments between each pair of languages. As with single-source parsing, we require that each of the source languages has access to a supervised parser.

The method for single-source projection can be naturally adapted to make use of multiple source languages by modifying the transferring weight function. For each of the source languages E in the set of source languages \mathcal{E} , we can define a transferring weight function, \tilde{w}_E just as before. The multi-source weight function $\tilde{w}_{\mathcal{E}}$ then is simply the average of the individual source language weight functions:

$$\tilde{w}_{\mathcal{E}}(e^F, \mathbf{x}^F) = \frac{1}{|\mathcal{E}|} \sum_{E \in \mathcal{E}} \tilde{w}_E(e^F, \mathbf{x}^F). \quad (8.2.7)$$

8.3 Experiments

8.3.1 Data

To train and evaluate parsers, we use CoNLL style dependency treebanks. Our source languages are English [158] and French [176]. Our target languages are Danish [157], German [160], Greek [186], Spanish [163], Italian [161], Dutch [187], Portuguese [162], and Swedish [164]. To avoid problems with divergent treebank conventions, where possible, we make use of the normalized treebanks in the Google Universal Treebanks v2.0 collection [118].

For parallel data, we make use of the Europarl raw data [119], sentence- and word-aligning each language pair. Sentence alignment is done using the Gale-Church method [174] and word alignment using the Berkeley aligner [175], an aligner that enforces agreement between the two alignment directions.

For the overall corpus, we find the 350,000 sentences that are present in the most languages and align sentences and words in this subsection. We discard any sentences that are not in a 1-to-1 alignment and greedily select sentences with high-quality word alignments until we have a corpus of 500,000 tokens. For this greedy selection, we calculate each sentence’s proportion of words in 1-to-1 alignments [188] across all language pairs and select the remaining sentence with the highest average alignment score. From among these sentences, we choose 500 at random for each language.

8.3.2 Features

All the experiments described in this chapter use the same sets of features for the parser, one a delexicalized feature set, used for scoring delexicalized edges in the transferring weight function, and one a fully lexicalized feature set, used for scoring all edges. The delexicalized feature set is exactly the feature set *Delex* described in [103]. Our lexicalized feature set simply takes the *Delex* feature set and adds conjunctions with the word forms of both the parent and child of the edge.

8.4 Results and Discussion

Table 8.19 shows the results of running both single-source and multi-source projection on eight different target languages. The numbers reported are Unlabeled Attachment Scores (UAS) excluding punctuation tokens. In their paper describing the single-source dependency projection algorithm which we extend here, Ma and Xia [112] provide a thorough comparison of the method against many other cross-lingual parsing systems. Because our results compare favorably to theirs, we omit such a comparison and instead refer the reader to their publication.

On average multi-source projection outperforms single-source projection, though perhaps less than one would expect. Much of the error comes from attempting to score arc that are not aligned to a source arc. While delexicalized scoring does a decent job at giving a score to these edges, if an edge is unaligned in one language pair, it's likely to be unaligned in another (it probably represents some sort of idiosyncrasy of the target language). Thus, adding additional source languages doesn't tend to help much in these cases.

Target Language	Single-Source Projection	Multi-source Projection
Danish	41.29	45.15
Dutch	60.63	63.47
German	62.05	66.92
Greek	63.11	65.71
Italian	72.55	75.17
Portuguese	71.26	70.87
Spanish	72.46	73.51
Swedish	68.97	62.04
<i>Average</i>	64.04	65.36

Table 8.19: UAS for single-source and multi-source projection for dependency parsing on various languages.

Because the results of Chapter VII showed that joint projection onto multiple target languages, in which agreement between target languages is enforced, did not yield any improvement over single-source projection, we do not pursue multi-target projection here. We still feel that this concept has some merit as the true correct parses for the target language sentences do exhibit substantial agreement with one another. This would seem to be a case of non-convex optimization where the learner gets stuck in a poor local minimum. With better initialization and more careful guidance, we believe this technique could still be fruitful.

Having evaluated two different projection systems that attempt to deal with errors by projecting from multiple source languages at once, it is clear that while multiple source projection does help to reduce errors, there is still a considerable gap between the performance that is possible with projection systems and with supervised systems. While there is likely still more that can be done from a machine learning perspective, the most obvious gap in projection

systems is the lack of linguistic knowledge used. From examining the types of errors made with two different syntax projection systems (POS-tagging and dependency parsing), it is clear that some sort of linguistic manifest listing known patterns where languages express concepts differently syntactically would be enormously helpful. While not every case would be easily correctable, if several of the most prominent cases could be detected and corrected, it could represent a major improvement over current naïve projection systems.

CHAPTER IX

Conclusions and Future Directions

9.1 Summary of Contributions

Though the Internet continues to reach more of the world and gain speakers of more languages, NLP technologies have only been deployed for a small fraction of those languages. Tools such as parsers and POS taggers are traditionally built by running supervised machine learning methods on a large annotated corpus. There are usually not any such corpora for low-resource languages, and so rather than investigating what can be done without these annotated corpora, low-resource languages are typically ignored.

The main question we explored in this thesis is “What can be practically done to build NLP tools for low-resource languages with existing resources?” The chapters of the thesis move through the stages of an NLP pipeline for processing low-resource languages, starting with language identification, moving to POS tagging, and finishing with dependency parsing.

In Chapter III, we describe an actual NLP pipeline for low-resource language processing that we developed along with this thesis, which demonstrates the techniques described here on real

documents. It starts off knowing few words of a low-resource language, but uses these words to construct queries that can be sent to a Web search engine to retrieve more documents. After retrieving these documents, it is able to automatically segment text according to language and tag the words with a POS.

In Chapter IV, we introduce word-level language identification. Documents retrieved from the Web containing a low-resource language typically also contain text in other languages, so separating out which text belongs to which language is important and could not be done with existing language identification techniques, which assumed that documents were monolingual. We develop weakly supervised methods that require very little data in order to produce a highly accurate labeling of the words. The methods of this chapter are applicable in the case when the document contains exactly two languages and the identities of the languages are known beforehand.

In Chapter V, we investigate the problem of word-level language identification when the previous assumptions no longer hold, that is, when the document may contain any number of languages mixed together and the identities of the languages are unknown. We create a number of Bayesian models for word-level language identification. These models begin with a small amount of sample text for each language and iteratively narrow down the set of languages contained in each document, finally arriving at a highly accurate labeling of words.

Chapter VI transitions to looking at higher order NLP tasks, starting with POS tagging. Low-resource languages often lack annotated corpora for performing supervised POS tagging, so we investigate how to transfer a supervised POS tagger from a high-resource language to a low-resource target language. This chapter takes the approach of direct transfer, using a common POS tagset for both high-resource and low-resource languages and applying the

trained parser directly to the target language. We show that if direct transfer tagging is used as an initialization point for a type-supervised POS tagger, we can create higher accuracy POS taggers in the target language on average than we can without it.

In Chapter VII, we continue to study the problem of cross-lingual POS tagging, this time making use of parallel text. Using posterior regularization, we create methods to project POS taggers from multiple high-resource source languages onto both a single target language and onto multiple target languages simultaneously. These methods require that we have a massively parallel corpus, which is a corpus with the same text translated into many different languages. We find that multiple-source, single-target projection consistently outperforms the previous state of the art on the task of cross-lingual tag projection.

In Chapter VIII, we turn to the problem of parsing in low-resource languages. As in the previous chapter, we develop methods for projecting dependencies from multiple source languages onto a single target language. These methods again consistently improve on the performance of the previous state of the art in cross-lingual dependency parsing.

9.2 Future Directions

There are several directions in which this research could be taken. One obvious direction is to continue to apply these techniques, especially those of cross-lingual projected, toward higher-level NLP applications, such as information extraction or sentiment analysis. Likewise, the techniques described here could be applied as part of a pipeline that enables another higher-level task, such as machine translation. Almost all modern machine translation relies on large amounts of parallel text for training, texts that are not as widely available for low-resource languages. Making use, however, of linguistic knowledge and NLP tools such as

parsing allows for high quality machine translation with less example text. Areas such as information retrieval and health informatics could stand to benefit enormously from low-resource NLP as machine translation can help to enable access to the Web and healthcare.

This work also has natural synergies with the work of linguists, especially those studying endangered languages. The Human Language Project [25] proposes that descriptive and computational linguists collaborate more closely in creating documentation in an electronically readable format. In order to making use of language documentation and language experts, typical NLP approaches will need to be adapted to be able to as easily exploit partial annotations from experts as they are fully labeled data. While this thesis focuses on text from the Web and other relatively clean sources (like the Bible), there are likely still many languages for which these techniques would not be applicable: languages with no writing system, languages without standardized orthography, and languages that don't match other assumptions we've made here (e.g. can be tokenized using whitespace). Further research should focus on not only using existing NLP technologies such as the ones described in this paper for language description, but also on extending methods to be appropriate for larger numbers of languages.

Finally, there is still considerable room for improvement on the tasks studied in this thesis. All of the techniques described here are rather naïve linguistically. Certainly, much more can be gained by making use of linguistic knowledge, for example, building into a cross-lingual POS tagging system linguistic knowledge about why certain concepts are expressed differently syntactically in different languages. Such approaches are greatly lacking in NLP research today.

Bibliography

- [1] D. Nettle, “Explaining global patterns of language diversity,” *Journal of anthropological archaeology*, vol. 17, no. 4, pp. 354–374, 1998.
- [2] D. A. Wagner, R. L. Venezky, and B. V. Street, *Literacy: An international handbook*. Westview Press Boulder, 1999.
- [3] M. P. Lewis, “Ethnologue: Languages of the world sixteenth edition,” *Dallas, Tex.: SIL International. Online version: <http://www.ethnologue.com>*, 2009.
- [4] D. Zeman, D. Marecek, M. Popel, L. Ramasamy, J. Stepánek, Z. Zabokrtský, and J. Hajic, “Hamletd: To parse or not to parse?,” in *LREC*, pp. 2735–2741, 2012.
- [5] “Världens 100 största språk 2007 (the world’s 100 largest languages in 2007),” in *Nationalencyklopedin*.
- [6] “Human development report 2013,” tech. rep., United Nations Development Programme, 2013.
- [7] M. Krauss, “The world’s languages in crisis,” *Language*, vol. 68, no. 1, pp. 4–10, 1992.
- [8] O. Streiter, K. P. Scannell, and M. Stuflessner, “Implementing NLP projects for non-central languages: instructions for funding bodies, strategies for developers,” *Machine Translation*, vol. 20, no. 4, pp. 267–289, 2006.
- [9] D. W. Oard and F. J. Och, “Rapid-response machine translation for unexpected languages,” in *Proceedings of MT Summit IX*, Citeseer, 2003.
- [10] P. Resnik and N. A. Smith, “The Web as a parallel corpus,” *Computational Linguistics*, vol. 29, no. 3, pp. 349–380, 2003.
- [11] D. S. Munteanu and D. Marcu, “Improving machine translation performance by exploiting non-parallel corpora,” *Computational Linguistics*, vol. 31, no. 4, pp. 477–504, 2005.

- [12] E. M. Bender, “Linguistically naïve!= language independent: Why NLP needs linguistic typology,” in *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pp. 26–32, Association for Computational Linguistics, 2009.
- [13] K. Yamada and K. Knight, “A syntax-based statistical translation model,” in *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pp. 523–530, Association for Computational Linguistics, 2001.
- [14] G. Bārzdīns and N. Grūzītis, “Dependency-based hybrid model of syntactic analysis for the languages with a rather free word order,” 2007.
- [15] F. Tyers and K. Donnelly, “Apertium-cy-a collaboratively-developed free RBMT system for Welsh to English,” *The Prague Bulletin of Mathematical Linguistics*, vol. 91, no. 1, pp. 57–66, 2009.
- [16] M. Humayoun and A. Ranta, “Developing Punjabi morphology, corpus and lexicon.,” in *PACLIC*, pp. 163–172, 2010.
- [17] G. De Pauw, N. Maajabu, and P. W. Wagacha, “A knowledge-light approach to Luo machine translation and part-of-speech tagging,” in *Proceedings of the Second Workshop on African Language Technology (AfLaT 2010). Valletta, Malta: European Language Resources Association (ELRA)*, pp. 15–20, 2010.
- [18] A. Mayor, I. Alegria, A. D. De Ilarraza, G. Labaka, M. Lersundi, and K. Sarasola, “Matxin, an open-source rule-based machine translation system for basque,” *Machine translation*, vol. 25, no. 1, pp. 53–82, 2011.
- [19] A. Rios, “Spell checking an agglutinative language: Quechua,” in *5th Language and Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*, pp. 51–55, 2011.
- [20] G. De Pauw, G.-M. De Schryver, and J. van de Loo, “Resource-light Bantu part-of-speech tagging,” *Language Technology for Normalisation of Less-Resourced Languages*, p. 85, 2012.
- [21] K. P. Scannell, “The Crúbadán project: Corpus building for under-resourced languages,” in *Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop*, vol. 4, (Louvain-la-Neuve, Belgium), pp. 5–15, 2007.
- [22] K. Scannell, “Automatic thesaurus generation for minority languages: an irish example,” in *Actes de la 10e conférence TALNa Batz-sur-Mer*, vol. 2, pp. 203–212, Citeseer, 2003.

- [23] K. P. Scannell, “Statistical unicodification of African languages,” *Language resources and evaluation*, vol. 45, no. 3, pp. 375–386, 2011.
- [24] K. P. Scannell, “Machine translation for closely related language pairs,” in *Proceedings of the Workshop on Strategies for developing machine translation for minority languages at LREC*, Citeseer, 2006.
- [25] S. Abney and S. Bird, “The human language project: building a universal corpus of the world’s languages,” in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 88–97, Association for Computational Linguistics, 2010.
- [26] C. Biemann, G. Heyer, U. Quasthoff, and M. Richter, “The Leipzig corpora collection—monolingual corpora of standard size,” *Proceedings of Corpus Linguistic*, 2007.
- [27] D. Goldhahn, T. Eckart, and U. Quasthoff, “Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages.,” in *LREC*, pp. 759–765, 2012.
- [28] D. Yarowsky, G. Ngai, and R. Wicentowski, “Inducing multilingual text analysis tools via robust projection across aligned corpora,” in *Proceedings of the first international conference on Human language technology research*, pp. 1–8, Association for Computational Linguistics, 2001.
- [29] S. Buchholz and E. Marsi, “CoNLL-x shared task on multilingual dependency parsing,” in *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pp. 149–164, Association for Computational Linguistics, 2006.
- [30] J. Nilsson, S. Riedel, and D. Yuret, “The CoNLL 2007 shared task on dependency parsing,” in *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pp. 915–932, sn, 2007.
- [31] S. Bird and G. Simons, “Seven dimensions of portability for language documentation and description,” *Language*, vol. 79, no. 3, pp. pp. 557–582, 2003.
- [32] F. Lüpke, “Orthography development,” in *Handbook of endangered languages* (P. Austin and J. Sallabank, eds.), Cambridge University Press, 2011.
- [33] C. F. Voegelin and Z. S. Harris, “Methods for determining intelligibility among dialects of natural languages,” *Proceedings of the American Philosophical Society*, pp. 322–329, 1951.

- [34] K. R. Beesley, “Language identifier: A computer program for automatic natural-language identification of on-line text,” in *Proceedings of the 29th Annual Conference of the American Translators Association*, vol. 47, p. 54, 1988.
- [35] A. House and E. Neuburg, “Toward automatic identification of the language of an utterance. i. preliminary methodological considerations,” *The Journal of the Acoustical Society of America*, vol. 62, p. 708, 1977.
- [36] B. Hughes, T. Baldwin, S. Bird, J. Nicholson, and A. MacKinlay, “Reconsidering language identification for written language resources,” in *Proc. International Conference on Language Resources and Evaluation*, pp. 485–488, 2006.
- [37] T. Dunning, “Statistical identification of language,” tech. rep., 1994.
- [38] A. Poutsma, “Applying monte carlo techniques to language identification,” *Language and Computers*, vol. 45, no. 1, pp. 179–189, 2002.
- [39] W. B. Cavnar and J. M. Trenkle, “N-gram-based text categorization,” in *Proceedings of the Third Annual Symposium on Document Analysis and Information (SDAIR 94)*, (Las Vegas, Nevada), pp. 161–175, 1994.
- [40] F. Xia, W. Lewis, and H. Poon, “Language ID in the context of harvesting language data off the web,” in *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, (Athens, Greece), pp. 870–878, Association for Computational Linguistics, March 2009.
- [41] M. Lui and T. Baldwin, “Accurate language identification of twitter messages,” in *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)@EACL*, pp. 17–25, 2014.
- [42] A. K. Singh and J. Gorla, “Identification of languages and encodings in a multilingual document,” *Building and Exploring Web Corpora (WAC3-2007)*, vol. 4, p. 95, 2007.
- [43] J. M. Prager, “Linguini: Language identification for multilingual documents,” in *System Sciences, 1999. HICSS-32. Proceedings of the 32nd Annual Hawaii International Conference on*, pp. 11–pp, IEEE, 1999.
- [44] M. Lui, J. H. Lau, and T. Baldwin, “Automatic detection and language identification of multilingual documents,” *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 27–40, 2014.
- [45] H. Yamaguchi and K. Tanaka-Ishii, “Text segmentation by language using minimum description length,” in *Proceedings of the 50th Annual Meeting of the Association for*

- Computational Linguistics: Long Papers-Volume 1*, pp. 969–978, Association for Computational Linguistics, 2012.
- [46] D. Nguyen and A. S. Dogruöz, “Word level language identification in online multilingual communication,” *Proceedings of EMNLP2013, Seattle, USA*, 2013.
- [47] R. S. Roy, M. Choudhury, P. Majumder, and K. Agarwal, “Overview and datasets of FIRE 2013 track on transliterated search,” in *FIRE@ISM 2013*, 2013.
- [48] S. Gella, K. Bali, and M. Choudhury, “"ye word kis lang ka hai bhai?" testing the limits of word level language identification,” NLP AI, 2014.
- [49] T. Solorio, E. Blair, S. Maharjan, S. Bethard, M. Diab, M. Ghoneim, A. Hawwari, F. AlGhamdi, J. Hirschberg, A. Chang, and P. Fung, “Overview for the first shared task on language identification in code-switched data,” in *Proceedings of the First Workshop on Computational Approaches to Code Switching*, (Doha, Qatar), pp. 62–72, Association for Computational Linguistics, October 2014.
- [50] A. K. Joshi, “Processing of sentences with intra-sentential code-switching,” in *Proceedings of the 9th conference on Computational linguistics-Volume 1*, pp. 145–150, Academia Praha, 1982.
- [51] T. Solorio and Y. Liu, “Learning to predict code-switching points,” in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, (Honolulu, Hawaii), pp. 973–981, Association for Computational Linguistics, October 2008.
- [52] C.-L. Chu, D.-c. Lyu, and R.-y. Lyu, “Language identification on code-switching speech,” in *Proceedings of ROCLING*, 2007.
- [53] D.-C. Lyu and R.-Y. Lyu, “Language identification on code-switching utterances using multiple cues,” in *Ninth Annual Conference of the International Speech Communication Association*, 2008.
- [54] C. Sutton and A. McCallum, “An introduction to conditional random fields for relational learning,” *Introduction to statistical relational learning*, pp. 93–128, 2006.
- [55] A. Y. Ng and M. I. Jordan, “On discriminative vs. generative classifiers: A comparison of logistic regression and naïve Bayes,” *Advances in neural information processing systems*, vol. 14, p. 841, 2002.
- [56] X. Carreras, “Experiments with a higher-order projective dependency parser,” in *EMNLP-CoNLL*, pp. 957–961, 2007.

- [57] T. Koo and M. Collins, “Efficient third-order dependency parsers,” in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 1–11, Association for Computational Linguistics, 2010.
- [58] X. Ma and H. Zhao, “Fourth-order dependency parsing.,” in *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pp. 785–796, The COLING 2012 Organizing Committee, 2012.
- [59] J. M. Eisner, “Three new probabilistic models for dependency parsing: An exploration,” in *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pp. 340–345, Association for Computational Linguistics, 1996.
- [60] M. Collins and Y. Singer, “Unsupervised models for named entity classification,” in *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 100–110, 1999.
- [61] A. Haghighi and D. Klein, “Prototype-driven learning for sequence models,” in *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, (New York City, USA), pp. 320–327, Association for Computational Linguistics, June 2006.
- [62] M.-W. Chang, L. Ratinov, and D. Roth, “Guiding semi-supervision with constraint-driven learning,” in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, (Prague, Czech Republic), pp. 280–287, Association for Computational Linguistics, June 2007.
- [63] K. Bellare, G. Druck, and A. McCallum, “Alternating projections for learning with expectation constraints,” in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 43–50, AUAI Press, 2009.
- [64] G. Druck, G. Mann, and A. McCallum, “Learning from labeled features using generalized expectation criteria,” in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2008)*, pp. 595–602, ACM, 2008.
- [65] K. Ganchev, J. a. Graça, J. Gillenwater, and B. Taskar, “Posterior regularization for structured latent variable models,” *The Journal of Machine Learning Research*, vol. 11, pp. 2001–2049, 2010.
- [66] N. A. Smith and J. Eisner, “Contrastive estimation: Training log-linear models on unlabeled data,” in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, (Ann Arbor, Michigan), pp. 354–362, Association for Computational Linguistics, June 2005.

- [67] G. S. Mann and A. McCallum, “Generalized expectation criteria for semi-supervised learning of conditional random fields,” in *Proceedings of ACL-08: HLT*, (Columbus, Ohio), pp. 870–878, Association for Computational Linguistics, June 2008.
- [68] S. Singh, D. Hillard, and C. Leggetter, “Minimally-supervised extraction of entities from text advertisements,” in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, (Los Angeles, California), pp. 73–81, Association for Computational Linguistics, June 2010.
- [69] A. McCallum, G. Mann, and G. Druck, “Generalized expectation criteria,” tech. rep., University of Massachusetts Amherst.
- [70] R. McDonald, S. Petrov, and K. Hall, “Multi-source transfer of delexicalized dependency parsers,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 62–72, Association for Computational Linguistics, 2011.
- [71] O. Täckström, “Nudging the envelope of direct transfer methods for multilingual named entity recognition,” in *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pp. 55–63, Association for Computational Linguistics, 2012.
- [72] J. Hana, A. Feldman, and C. Brew, “A resource-light approach to Russian morphology: Tagging Russian using Czech resources,” in *EMNLP*, pp. 222–229, 2004.
- [73] A. Feldman, J. Hana, and C. Brew, “A cross-language approach to rapid creation of new morpho-syntactically annotated resources,” in *Proceedings of LREC*, pp. 549–554, 2006.
- [74] S. Reddy and S. Sharoff, “Cross language POS taggers (and other tools) for Indian languages: An experiment with Kannada using Telugu resources,” *Cross Lingual Information Access*, vol. 11, 2011.
- [75] J. Lafferty, A. McCallum, and F. C. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proceedings of the 18th International Conference on Machine Learning 2001 (ICML 2001)*, 2001.
- [76] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, “Feature-rich part-of-speech tagging with a cyclic dependency network,” in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pp. 173–180, Association for Computational Linguistics, 2003.

- [77] C. D. Manning, “Part-of-speech tagging from 97% to 100%: is it time for some linguistics?,” in *Computational Linguistics and Intelligent Text Processing*, pp. 171–189, Springer, 2011.
- [78] B. Merialdo, “Tagging english text with a probabilistic model,” *Computational linguistics*, vol. 20, no. 2, pp. 155–171, 1994.
- [79] N. A. Smith and J. Eisner, “Contrastive estimation: Training log-linear models on unlabeled data,” in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 354–362, Association for Computational Linguistics, 2005.
- [80] K. Toutanova and M. Johnson, “A Bayesian LDA-based model for semi-supervised part-of-speech tagging,” in *Advances in Neural Information Processing Systems*, pp. 1521–1528, 2007.
- [81] K. S. Hasan and V. Ng, “Weakly supervised part-of-speech tagging for morphologically-rich, resource-scarce languages,” in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 363–371, Association for Computational Linguistics, 2009.
- [82] S. Goldwater and T. Griffiths, “A fully Bayesian approach to unsupervised part-of-speech tagging,” in *Annual meeting-association for computational linguistics*, vol. 45, p. 744, 2007.
- [83] T. Moon, K. Erk, and J. Baldridge, “Crouching Dirichlet, hidden Markov model: Unsupervised POS tagging with context local tag generation,” in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 196–206, Association for Computational Linguistics, 2010.
- [84] S. Ravi and K. Knight, “Minimized models for unsupervised part-of-speech tagging,” in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 504–512, Association for Computational Linguistics, 2009.
- [85] S. Ravi, A. Vaswani, K. Knight, and D. Chiang, “Fast, greedy model minimization for unsupervised tagging,” in *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 940–948, Association for Computational Linguistics, 2010.
- [86] Y. Goldberg, M. Adler, and M. Elhadad, “EM can find pretty good HMM POS-taggers (when given a good start).,” in *ACL*, pp. 746–754, 2008.
- [87] S. Li, J. V. Graça, and B. Taskar, “Wiki-ly supervised part-of-speech tagging,” in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language*

- Processing and Computational Natural Language Learning*, pp. 1389–1398, Association for Computational Linguistics, 2012.
- [88] D. Das and S. Petrov, “Unsupervised part-of-speech tagging with bilingual graph-based projections.,” in *ACL*, pp. 600–609, 2011.
- [89] S. Petrov, D. Das, and R. McDonald, “A universal part-of-speech tagset,” in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)* (N. Calzolari, K. Choukri, T. Declerck, M. U. Doğan, B. Maegaard, J. Mariani, J. Odiijk, and S. Piperidis, eds.), (Istanbul, Turkey), pp. 2089–2096, European Language Resources Association (ELRA), May 2012. ACL Anthology Identifier: L12-1115.
- [90] O. Täckström, D. Das, S. Petrov, R. McDonald, and J. Nivre, “Token and type constraints for cross-lingual part-of-speech tagging,” *Transactions of the Association for Computational Linguistics*, vol. 1, pp. 1–12, 2013.
- [91] K. Ganchev and D. Das, “Cross-lingual discriminative learning of sequence models with posterior regularization.,” in *EMNLP*, pp. 1996–2006, 2013.
- [92] V. Fossum and S. Abney, “Automatically inducing a part-of-speech tagger by projecting from multiple source languages across aligned corpora,” in *Natural Language Processing-IJCNLP 2005*, pp. 862–873, Springer, 2005.
- [93] D. Garrette and J. Baldridge, “Learning a part-of-speech tagger from two hours of annotation,” in *Proceedings of NAACL-HLT*, pp. 138–147, 2013.
- [94] D. Garrette, J. Mielens, and J. Baldridge, “Real-world semi-supervised learning of pos-taggers for low-resource languages.,” in *ACL (1)*, pp. 583–592, 2013.
- [95] L. Duong, T. Cohn, K. Verspoor, S. Bird, and P. Cook, “What can we get from 1000 tokens? a case study of multilingual pos tagging for resource-poor languages,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 886–897, Association for Computational Linguistics, October 2014.
- [96] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, “Class-based n-gram models of natural language,” *Computational linguistics*, vol. 18, no. 4, pp. 467–479, 1992.
- [97] M. Johnson, “Why doesn’t EM find good HMM POS-taggers?,” in *EMNLP-CoNLL*, pp. 296–305, 2007.

- [98] C. Christodoulopoulos, S. Goldwater, and M. Steedman, “Two decades of unsupervised POS induction: How far have we come?,” in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 575–584, Association for Computational Linguistics, 2010.
- [99] M. Seraji, J. Nivre, *et al.*, “Bootstrapping a persian dependency treebank,” *Linguistic Issues in Language Technology*, vol. 7, no. 1, 2012.
- [100] D. Zeman and P. Resnik, “Cross-language parser adaptation between related languages,” in *IJCNLP*, pp. 35–42, 2008.
- [101] A. Søgaard, “Data point selection for cross-language adaptation of dependency parsers,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pp. 682–686, Association for Computational Linguistics, 2011.
- [102] A. Søgaard and J. Wulff, “An empirical study of non-lexical extensions to delexicalized transfer,” in *COLING (Posters)*, pp. 1181–1190, 2012.
- [103] O. Täckström, R. McDonald, and J. Nivre, “Target language adaptation of discriminative transfer parsers,” in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Atlanta, Georgia), pp. 1061–1071, Association for Computational Linguistics, June 2013.
- [104] M. Xiao and Y. Guo, “Distributed word representation learning for cross-lingual dependency parsing,” *CoNLL-2014*, p. 119, 2014.
- [105] R. Hwa, P. Resnik, A. Weinberg, C. Cabezas, and O. Kolak, “Bootstrapping parsers via syntactic projection across parallel texts,” *Natural language engineering*, vol. 11, no. 3, pp. 311–325, 2005.
- [106] D. A. Smith and J. Eisner, “Parser adaptation and projection with quasi-synchronous grammar features,” in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pp. 822–831, Association for Computational Linguistics, 2009.
- [107] K. Spreyer and J. Kuhn, “Data-driven dependency parsing of new languages using incomplete and noisy training data,” in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pp. 12–20, Association for Computational Linguistics, 2009.

- [108] K. Ganchev, J. Gillenwater, and B. Taskar, “Dependency grammar induction via bi-text projection constraints,” in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pp. 369–377, Association for Computational Linguistics, 2009.
- [109] K. Liu, Y. Lü, W. Jiang, and Q. Liu, “Bilingually-guided monolingual dependency grammar induction,” in *ACL (1)*, pp. 1063–1072, 2013.
- [110] W. Jiang and Q. Liu, “Dependency parsing and projection based on word-pair classification,” in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 12–20, Association for Computational Linguistics, 2010.
- [111] Z. Li, M. Zhang, and W. Chen, “Soft cross-lingual syntax projection for dependency parsing,” in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, (Dublin, Ireland), pp. 783–793, Dublin City University and Association for Computational Linguistics, August 2014.
- [112] X. Ma and F. Xia, “Unsupervised dependency parsing with transferring distribution via parallel guidance and entropy regularization,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, vol. 1, pp. 1337–1348, Association for Computational Linguistics, 2014.
- [113] F. Xia and W. D. Lewis, “Multilingual structural projection across interlinear text.,” in *HLT-NAACL*, pp. 452–459, 2007.
- [114] R. Georgi, F. Xia, and W. D. Lewis, “Improving dependency parsing with interlinear glossed text and syntactic projection.,” in *COLING (Posters)*, pp. 371–380, 2012.
- [115] R. Georgi, F. Xia, and W. D. Lewis, “Enhanced and portable dependency projection algorithms using interlinear glossed text.,” in *ACL (2)*, pp. 306–311, 2013.
- [116] W. D. Lewis and F. Xia, “Developing odin: A multilingual repository of annotated language data for hundreds of the world’s languages,” *Literary and Linguistic Computing*, vol. 25, no. 3, pp. 303–319, 2010.
- [117] J. Morsink, *The Universal Declaration of Human Rights: origins, drafting, and intent*. University of Pennsylvania Press, 1999.
- [118] R. T. McDonald, J. Nivre, Y. Quirnbach-Brundage, Y. Goldberg, D. Das, K. Ganchev, K. B. Hall, S. Petrov, H. Zhang, O. Täckström, *et al.*, “Universal dependency annotation for multilingual parsing.,” in *ACL (2)*, pp. 92–97, 2013.

- [119] P. Koehn, “Europarl: A parallel corpus for statistical machine translation,” in *MT summit*, vol. 5, pp. 79–86, 2005.
- [120] P. Resnik, M. B. Olsen, and M. Diab, “The Bible as a parallel corpus: Annotating the ‘book of 2000 tongues’,” *Computers and the Humanities*, vol. 33, no. 1-2, pp. 129–153, 1999.
- [121] P. A. Chew, S. J. Verzi, T. L. Bauer, and J. T. McClain, “Evaluation of the Bible as a resource for cross-language information retrieval,” in *Proceedings of the workshop on multilingual language resources and interoperability*, pp. 68–74, Association for Computational Linguistics, 2006.
- [122] D. T. Haug and M. Jøhndal, “Creating a parallel treebank of the old indo-european Bible translations,” in *Proceedings of the Language Technology for Cultural Heritage Data Workshop (LaTeCH 2008), Marrakech, Morocco, 1st June 2008*, pp. 27–34, 2008.
- [123] T. Mayer and M. Cysouw, “Creating a massively parallel Bible corpus,” *Oceania*, vol. 135, no. 273, p. 40, 2014.
- [124] E. A. Nida, “The nature of dynamic equivalence in translating.,” *Babel: International Journal of Translation*, 1977.
- [125] D. Kamholz, P. Pool, and S. Colowick, “Panlex: Building a resource for panlingual lexical translation,” in *Proceedings of the International Conference on Language Resources (LREC’14)*, 2014.
- [126] M. Baroni and S. Bernardini, “Bootcat: Bootstrapping corpora and terms from the web,” in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, vol. 4, (Lisbon, Portugal), pp. 1313–1316, 2004.
- [127] E. Charniak, “Statistical parsing with a context-free grammar and word statistics,” *AAAI/IAAI*, vol. 2005, pp. 598–603, 1997.
- [128] S. Clark, J. R. Curran, and M. Osborne, “Bootstrapping pos taggers using unlabelled data,” in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pp. 49–55, Association for Computational Linguistics, 2003.
- [129] D. McClosky, E. Charniak, and M. Johnson, “Effective self-training for parsing,” in *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*, pp. 152–159, Association for Computational Linguistics, 2006.
- [130] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell, “Toward an architecture for never-ending language learning.,” in *AAAI*, 2010.

- [131] E. Agichtein and L. Gravano, “Snowball: Extracting relations from large plain-text collections,” in *Proceedings of the fifth ACM conference on Digital libraries*, pp. 85–94, ACM, 2000.
- [132] M. Banko and O. Etzioni, “Strategies for lifelong knowledge extraction from the web,” in *Proceedings of the 4th international conference on Knowledge capture*, pp. 95–102, ACM, 2007.
- [133] J. Landsbergen, “The Rosetta project,” (Munich, Germany), pp. 82–87, 1989.
- [134] A. McCallum, “Mallet: A machine learning for language toolkit.” <http://mallet.cs.umass.edu>, 2002.
- [135] S. Sarawagi and W. W. Cohen, “Semi-markov conditional random fields for information extraction,” *Advances in Neural Information Processing Systems (NIPS 2004)*, vol. 17, pp. 1185–1192, 2004.
- [136] G. Druck, *Generalized Expectation Criteria for Lightly Supervised Learning*. PhD thesis, University of Massachusetts Amherst, 2011.
- [137] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.
- [138] G. S. Mann and A. McCallum, “Generalized expectation criteria for semi-supervised learning with weakly labeled data,” *The Journal of Machine Learning Research*, pp. 955–984, 2010.
- [139] E. F. Tjong Kim Sang and F. De Meulder, “Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition,” in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003* (W. Daelemans and M. Osborne, eds.), pp. 142–147, 2003.
- [140] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, “Indexing by latent semantic analysis,” *JASIS*, vol. 41, no. 6, pp. 391–407, 1990.
- [141] T. Hofmann, “Probabilistic latent semantic indexing,” in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 50–57, ACM, 1999.
- [142] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet allocation,” *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

- [143] T. L. Griffiths, M. I. Jordan, J. B. Tenenbaum, and D. M. Blei, “Hierarchical topic models and the nested chinese restaurant process,” *Advances in neural information processing systems*, vol. 16, pp. 106–114, 2004.
- [144] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth, “The author-topic model for authors and documents,” in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 487–494, AUAI Press, 2004.
- [145] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum, “Integrating topics and syntax,” in *Advances in neural information processing systems*, pp. 537–544, 2004.
- [146] H. M. Wallach, “Topic modeling: beyond bag-of-words,” in *Proceedings of the 23rd international conference on Machine learning*, pp. 977–984, ACM, 2006.
- [147] D. M. Blei and J. D. Lafferty, “Dynamic topic models,” in *Proceedings of the 23rd international conference on Machine learning*, pp. 113–120, ACM, 2006.
- [148] L. AlSumait, D. Barbará, and C. Domeniconi, “On-line LDA: Adaptive topic models for mining text streams with applications to topic detection and tracking,” in *Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on*, pp. 3–12, IEEE, 2008.
- [149] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, “Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora,” in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pp. 248–256, Association for Computational Linguistics, 2009.
- [150] T. L. Griffiths and M. Steyvers, “Finding scientific topics,” *Proceedings of the National academy of Sciences of the United States of America*, vol. 101, no. Suppl 1, pp. 5228–5235, 2004.
- [151] W. Li and A. McCallum, “Pachinko allocation: DAG-structured mixture models of topic correlations,” in *Proceedings of the 23rd international conference on Machine learning*, pp. 577–584, ACM, 2006.
- [152] A. Gruber, Y. Weiss, and M. Rosen-Zvi, “Hidden topic markov models,” in *International Conference on Artificial Intelligence and Statistics*, pp. 163–170, 2007.
- [153] A. Gruber and A. C. Popat, “Notes regarding computations in OpenHTMM,” 2007.
- [154] P. Liang, M. I. Jordan, and D. Klein, “Type-based MCMC,” in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 573–581, Association for Computational Linguistics, 2010.

- [155] D. Garrette and J. Baldridge, “Type-supervised hidden markov models for part-of-speech tagging with incomplete tag dictionaries,” in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 821–831, Association for Computational Linguistics, 2012.
- [156] K. Simov, P. Osenova, M. Slavcheva, S. Kolkovska, E. Balabanova, D. Doikoff, K. Ivanova, A. Simov, and M. Kouylekov, “Building a linguistically interpreted corpus of Bulgarian: the BulTreeBank,” in *LREC*, 2002.
- [157] M. T. Kromann, “The Danish dependency treebank and the DTAG treebank tool,” in *Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT)*, p. 217, 2003.
- [158] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, “Building a large annotated corpus of English: The Penn treebank,” *Computational linguistics*, vol. 19, no. 2, pp. 313–330, 1993.
- [159] K. Haverinen, T. Viljanen, V. Laippala, S. Kohonen, F. Ginter, and T. Salakoski, “Treebanking Finnish,” in *Proceedings of the Ninth International Workshop on Treebanks and Linguistic Theories (TLT)*, p. 79, 2010.
- [160] S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith, “The TIGER treebank,” in *Proceedings of the workshop on treebanks and linguistic theories*, pp. 24–41, 2002.
- [161] S. Montemagni, F. Barsotti, M. Battista, N. Calzolari, O. Corazzari, A. Lenci, A. Zampolli, F. Fanciulli, M. Massetani, R. Raffaelli, *et al.*, “Building the Italian syntactic-semantic treebank,” in *Treebanks*, pp. 189–210, Springer, 2003.
- [162] S. Afonso, E. Bick, R. Haber, and D. Santos, “Floresta Sintá (c) tica: A treebank for Portuguese,” in *LREC*, 2002.
- [163] M. Civit and M. A. Martí, “Building Cast3lb: A Spanish treebank,” *Research on Language and Computation*, vol. 2, no. 4, pp. 549–574, 2004.
- [164] J. Nivre, J. Nilsson, and J. Hall, “Talbanken05: A Swedish treebank with phrase structure and dependency annotation,” in *Proceedings of the fifth International Conference on Language Resources and Evaluation (LREC)*, pp. 1392–1395, 2006.
- [165] M. S. Dryer and M. Haspelmath, eds., *The World Atlas of Language Structures Online*. Munich: Max Planck Digital Library, 2011 ed., 2011.

- [166] P. P. Talukdar and K. Crammer, “New regularized algorithms for transductive learning,” in *Machine Learning and Knowledge Discovery in Databases*, pp. 442–457, Springer, 2009.
- [167] S. F. Chen and J. Goodman, “An empirical study of smoothing techniques for language modeling,” in *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pp. 310–318, Association for Computational Linguistics, 1996.
- [168] K. Hall, R. McDonald, J. Katz-Brown, and M. Ringgaard, “Training dependency parsers by jointly optimizing multiple objectives,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1489–1499, Association for Computational Linguistics, 2011.
- [169] A. Björkelund, B. Bohnet, L. Hafdell, and P. Nugues, “A high-performance syntactic and semantic dependency parser,” in *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, pp. 33–36, Association for Computational Linguistics, 2010.
- [170] R. McDonald, F. Pereira, K. Ribarov, and J. Hajič, “Non-projective dependency parsing using spanning tree algorithms,” in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 523–530, Association for Computational Linguistics, 2005.
- [171] K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar, “Posterior regularization for structured latent variable models,” *The Journal of Machine Learning Research*, vol. 11, pp. 2001–2049, 2010.
- [172] D. C. Liu and J. Nocedal, “On the limited memory BFGS method for large scale optimization,” *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.
- [173] P. Liang, *Semi-supervised learning for natural language*. PhD thesis, Massachusetts Institute of Technology, 2005.
- [174] W. A. Gale and K. W. Church, “A program for aligning sentences in bilingual corpora,” *Computational linguistics*, vol. 19, no. 1, pp. 75–102, 1993.
- [175] P. Liang, B. Taskar, and D. Klein, “Alignment by agreement,” in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pp. 104–111, Association for Computational Linguistics, 2006.
- [176] A. Abeillé and N. Barrier, “Enriching a French treebank.” in *LREC*, Citeseer, 2004.

- [177] M. Wolinski, K. Glowinska, and M. Swidzinski, “A preliminary version of Skladnica—a treebank of Polish,” in *Proceedings of the 5th Language and Technology Conference*, 2011.
- [178] M. Calacean, *Data-driven dependency parsing for Romanian*. PhD thesis, Master’s thesis, Uppsala University, 2008.
- [179] J. Blitzer, R. McDonald, and F. Pereira, “Domain adaptation with structural correspondence learning,” in *Proceedings of the 2006 conference on empirical methods in natural language processing*, pp. 120–128, Association for Computational Linguistics, 2006.
- [180] J. Blitzer, M. Dredze, F. Pereira, *et al.*, “Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification,” in *ACL*, vol. 7, pp. 440–447, 2007.
- [181] P. Koehn and J. Schroeder, “Experiments in domain adaptation for statistical machine translation,” in *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 224–227, Association for Computational Linguistics, 2007.
- [182] J. Uszkoreit and T. Brants, “Distributed word clustering for large scale class-based language modeling in machine translation.,” in *ACL*, pp. 755–762, 2008.
- [183] D. A. Smith and J. Eisner, “Bootstrapping feature-rich dependency parsers with entropic priors.,” in *EMNLP-CoNLL*, pp. 667–677, Citeseer, 2007.
- [184] S. Abney, “Understanding the Yarowsky algorithm,” *Computational Linguistics*, vol. 30, no. 3, pp. 365–395, 2004.
- [185] D. Yarowsky, “Unsupervised word sense disambiguation rivaling supervised methods,” in *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pp. 189–196, Association for Computational Linguistics, 1995.
- [186] P. Prokopidis, E. Desipri, M. Koutsombogera, H. Papageorgiou, and S. Piperidis, “Theoretical and practical issues in the construction of a Greek dependency treebank,” in *In Proc. of the 4th Workshop on Treebanks and Linguistic Theories (TLT)*, 2005.
- [187] L. Van der Beek, G. Bouma, R. Malouf, and G. Van Noord, “The Alpino dependency treebank,” *Language and Computers*, vol. 45, no. 1, pp. 8–22, 2002.
- [188] L. Duong, P. Cook, S. Bird, and P. Pecina, “Simpler unsupervised POS tagging with bilingual projections.,” in *ACL (2)*, pp. 634–639, 2013.