BILINGUAL AND CROSS-LINGUAL LEARNING OF SEQUENCE MODELS WITH BITEXT

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Mengqiu Wang
March 2014

This dissertation is online at: http://purl.stanford.edu/nq879qs3428

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Christopher Manning, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Daniel Jurafsky

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Percy Liang

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Abstract

Information extraction technologies such as detecting the names of people and places in natural language texts are becoming ever more prevalent, as the amount of unstructured text data grows exponentially. Tremendous progress has been made in the past decade in learning supervised sequence models for such tasks, and current state-of-the-art results are in the lower 90s in term of F1 score, for resource-rich languages like English and widely-studied datasets such as the CoNLL newswire corpus.

However, the performance of existing supervised methods lag by a significant margin when evaluated for non-English languages, new datasets, and domains other than newswire. Furthermore, for resource-poor languages where there is often little or no annotated training data, neither supervised nor existing unsupervised methods tend to work well.

This thesis describes a series of models and experiments in response to these challenges in three specific areas.

Firstly, we address the problem of balancing between feature weight undertraining and overtraining in learning log-linear models. We explore the use of two novel regularization techniques—a mixed $\ell_2\ell_1$ norm in a product of experts ensemble, and an adaptive regularization with feature noising—to show that they can be very effective in improving system performance.

Secondly, we challenge the conventional wisdom of employing a linear architecture and sparse discrete feature representation for sequence labeling tasks, and closely examine the connection and tradeoff between a linear versus nonlinear architecture, as well as a discrete versus continuous feature representation. We show that a nonlinear architecture enjoys a significant advantage over linear architecture when used with continuous feature vectors, but does not seem to offer benefits over traditional sparse features.

Lastly, we explore methods that leverage readily available unlabeled parallel text from translation as a rich source of constraints for learning bilingual models that transfer knowledge from English to resource-poor languages. We formalize the model as loopy Markov Random Fields, and propose a suite of approximate inference methods for decoding. Evaluated on standard test sets for five non-English languages, our semi-supervised models yield significant improvements over the state-of-the-art results for all five languages.

We further propose a cross-lingual projection method that is capable of learning sequence models for languages where there are no annotated resources at all. Our method projects model posteriors from English to the foreign side over word alignments on bitext, and handles missing and noisy labels via expectation regularization. Learned with no annotated data at all, our model attains the same accuracy as supervised models trained with thousands of labeled examples.

Acknowledgements

Words cannot express my gratitude enough for my advisor, Chris. Without a doubt, you are the best advisor I could ever hoped for. You took care of me all these years I was at Stanford. Time and time again you pulled me up when I was drowning in my own confusion, supported when I left to explore the world, and received me back when I was lost. The one thing that I appreciate the most is how you always wanted the betterment of me, even if my plans were not in the best interest of the lab or the project. You are a true educator, an inspiration that I'll always draw strength from. I would have never gotten where I am today in a million years without your support and trust. Thank you so much.

I also want to thank my other committee members, Dan and Percy. Dan, I can recall the first time we spoke, back in 2007 at EMNLP in Prague, I approached you to tell you about the work I was doing at the time, given how famous and busy you were, I expected some chitchat and maybe a hand shake, but instead you sat me down on a couch and heard me out for almost half an hour. I was thinking, how can someone so smart and so famous and yet be so humble and warm? After 7 years of working with you, I still have the same wonder of how smart and nice you are. What also amazes me is how quickly you can grasp the big picture of any new ideas, and immediately point out the merit or problem with the ideas. Your lifestyle is also amazing and inspiring, you always have amazing stories about music and food. Percy, you are hands down the smartest and most talented person I have ever worked with. I hope you don't mind me telling people stories of how when we first met you told me that you just moved to the bay area and was looking for an apartment stay, and in one afternoon you wrote a program that scrapes all of the listings on Craigslist and plotted them out on Google maps with pricing info, lease term, property attributes, etc., and you did that because it was easier than searching on Google or Craigslist. I always

wonder if the world would be better or worse if academia didn't captivate your intellectual interest — a million brilliant startups vs. breakthrough in machine learning and NLP, hmm . . . I would like to give special thanks to Wanxiang, who not only gave me most of the ideas for the bilingual NER work and co-authored three papers with me, but has also been a wonderful friend and mentor. Other than profs, I am also super grateful of how wonderful everyone in the NLP group is. I will not attempt to list names here, but thank you all for inspiring me, keeping me great company when I was whining about research problems, and putting up with my bad jokes and random stories at NLP lunch. And finally, I want to thank my family. I get overly emotional when I talk about my family, having been away from home for over 14 years now means a lot of sacrifice and commitment from my parents and my grandma. They have always been supportive even when I was having doubts about my career. Without you I could have never made it.

To my mom, dad, and my grandparents.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Humans are incredibly good at understanding the semantic relational information contained in unstructured texts, which is a class of problems commonly referred to as Information Extraction (IE). Given the raw text of a Wikipedia article that describes the biography of a person, almost all adult readers can effortlessly identify the birthplace, occupation, or educational credentials of the protagonist. For example, in an example sentence "Steve Jobs is the CEO of Apple Inc.", "Steve Jobs" is a person name and "Apple Inc." is an organization. The IE task here is to extract these two phrases as canonicalized entities from a piece of surface text with latent semantics.

However, the ability of a human reader to process a large collection of such articles is bounded by both reading speed and fatigue. Traditional information-processing professions such as intelligence analysts and journalists are facing increasing challenges as the amount of free text grows at an incredible exponential rate. And it is not just the growth rate but the volume at which new text data is being generated that is staggering. Over 168 million emails were sent world-wide and over 1.1 million conversations had taken place over instant-messengers every minute.[1] In 2012, Gartner predicted that the amount of data will grow by 800% over the next five years, and 80% of the growth will come from

[1] http://www.go-gulf.com/blog/60-seconds-v2

unstructured sources such as email, text, social media, websites, etc. [2]

Take the task of extracting biographical information from Wikipedia for example. There are 4,402,482 articles in the English wikipedia today.[3] It will take a human annotator years to perform the task over the entire collection, and in the meantime the data is growing much faster than the processing speed of human annotators. Even in disciplines such as scientific research, the growth of the volume of publications is posing new challenges to the scientific community. Figure 1.1 plots the number of peer-reviewed scientific publications by country and by year. It is estimated that PubMed — a large compendium of biomedical research articles — currently indexes over 23 million articles. The sheer volume of these articles and the amount of knowledge contained in them will quickly overwhelm researchers and make it impossible to keep up with the advance of a field just by reading.

Therefore, there is an increasingly urgent need for automated methods to aid or replace humans at information extraction tasks. In this thesis, our primary goal is to advance the current state-of-the-art in automatic information extraction methods. In particular, we focus on one specific IE task — named-entity recognition (NER). Given an input sentence, an NER tagger identifies words that are part of a named entity, and assigns the entity type and relative position information. The most commonly seen and well-studied case of NER is in the newswire domain, where we build systems to recognize all mentions of named entities, such as people, locations, and organizations, from news articles. However, the techniques and methods developed for the newswire domain can in fact be transferred to much broader domains. In recent years, we have seen NER systems blossoming in both research labs and real-world applications in domains such as legal studies (Surdeanu et al., 2010) and the life sciences (Settles, 2004).

NER is a foundational component for many higher-level NLP applications as well. For example, in factoid question answering, that is, answering simple who/what/when questions like "who is the 42nd president of United States", it is helpful to first identify whether it is the question is about *people* ("who"), *date/time* ("when") or other entity types such as an *organization* ("which company"). Then depending on the question type, we can use

[2] `http://www.forbes.com/sites/tomgroenfeldt/2012/01/06/big-data-big-money-says-it-is-a-paradigm-buster`

[3] As of Dec 16th, 2013, according to `http://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia`

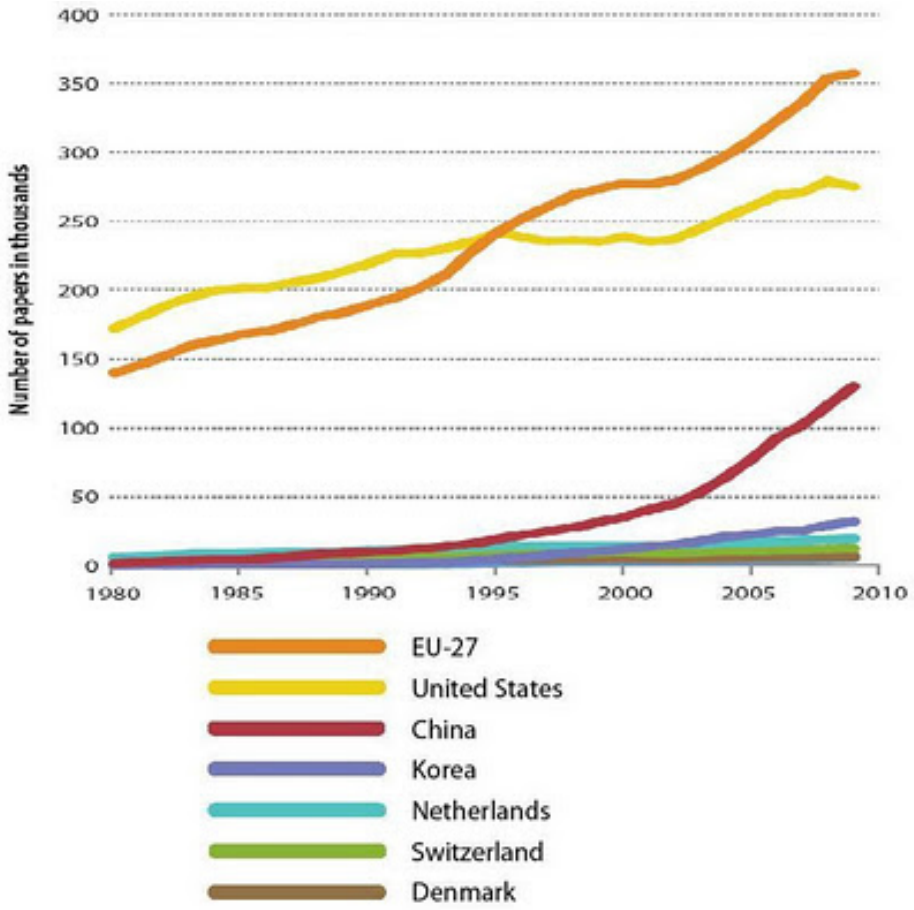


Figure 1.1: The growth of number of scientific publications by year 2010. The source of this plot is from `http://conversableeconomist.blogspot.com/2012/01/us-science-needs-to-look-beyond-our.html`.

NER to identify all the matching entities from documents and consider them as potential answer candidates.

NER can also be leveraged to improve the performance of applications such as statistical machine translation (SMT). Entities such as person names, locations, and organizations are content words that carry most of the information expressed in the source sentence. Knowing them can guide the MT system to weigh them with greater importance and lead the system to better preserve the meaning in the translation. Recognizing entities can also provide useful information for phrase detection and word sense disambiguation (e.g., "Melody" as in a female name has a different translation from the word "melody" in a musical sense), and can be directly leveraged to improve translation quality (Babych and Hartley, 2003). Another example where NER can help SMT is transliteration. Suppose our goal is to translate from Chinese to English using a phrase-based SMT system. The phrase table often has a high miss rate for rare terms such as Chinese people names; but if we could identify the name mentions using NER, we could employ a transliteration system to translate these words, and amend the shortcoming of the phrase-based system.

Over the years, the models and techniques for building effective NER systems have come a long way. The current state-of-the-art supervised NER system scores 90.8% in F1 measure (Ratinov and Roth, 2009) on the standard English CoNLL 2003 (Sang and Meulder, 2003) newswire corpus. One of the hallmarks of the advances in NER development is the use of supervised statistical learning methods, and in particular, discriminatively trained log-linear models that can incorporate a rich set of carefully engineered features, such as the maximum entropy (MaxEnt) model (Borthwick, 1999), and successors that extend MaxEnt into sequence models, such as the Maximum Entropy Markov Model (MEMM) (McCallum et al., 2000) and Conditional Random Fields (CRF) (Lafferty et al., 2001). We will give a more detailed introduction to these models in Chapter 2. This suite of models has enjoyed great success in NLP over the past decade, but there are still limitations and plenty of room for improvements. In this thesis, we closely examine several properties and limitations in the existing supervised learning approach that trains on labeled data with discriminative log-linear models, and conduct a three part investigation to search for new methods to improve system performance.

The first part of our investigation examines a property of log-linear models, which is

their ability to incorporate arbitrarily overlapping and inter-dependent features. On the one hand, this particular trait gives NLP practitioners great flexibility and power in designing features to capture various linguistic and statistical phenomena that are important for NER. However, on the other hand, complex dependencies among overlapping features can often give rise to in a phenomenon known as "feature co-adaptation" (Hinton et al., 2012) — some features become dependent on other features that they tend to co-occur with during training; this becomes a problem when the weaker features get neglected by the model because of the stronger features they co-occur with, but at test time the stronger features might be missing or do not fire frequently.

To mitigate this weight undertraining problem, we propose two new methods. Our first method is related to a popular method that attempts to counter weight undertraining, called logarithmic opinion pooling (LOP) (Heskes, 1998), which is a specialized form of product-of-experts model that automatically adjusts the weighting among experts. A major problem with LOP is that it requires significant amounts of domain expertise in designing effective experts. We propose a novel method that learns to induce experts — not just the weighting between them — through the use of a mixed $\ell_2\ell_1$ norm as previously seen in *elitist lasso* (Kowalski and Torrésani, 2009). Unlike its more popular sibling $\ell_1\ell_2$ norm (used in group lasso), which seeks feature sparsity at the group-level, the $\ell_2\ell_1$ norm encourages sparsity within feature groups. We demonstrate how this property can be leveraged as a competition mechanism to induce groups of diverse experts, and introduce a new formulation of elitist lasso MaxEnt in the FOBOS optimization framework (Duchi and Singer, 2009).

Our second method draws inspiration from the recently popular dropout training method pioneered by Hinton et al. (2012), which breaks feature co-adaptation during training by randomly omitting a subset of the features during each iteration of the learning procedure. A more recent discovery by Wager et al. (2013) showed that there is an interesting connection between what dropout training attempts to do and feature noising (adding noise into the original training data). In fact, a second-order approximation of the feature noising objective can be interpreted as applying adaptive regularization to the original objective. In this thesis, we show how to efficiently simulate training with artificially noised features in the context of *log-linear structured prediction*, without actually having to generate noised data.

A second aspect we investigate looks at the connection between model architecture and feature representation schemes. Traditionally, most statistical models in NLP employ a linear architecture,[4] and sparse discrete feature representations. Recently, Collobert et al. (2011) proposed "deep architecture" models for sequence labeling and showed promising results on a range of NLP tasks including NER. Two new changes were suggested: extending the model from a linear to a non-linear architecture; and replacing discrete feature representations with distributional feature representations in a continuous space. It has generally been argued that non-linearity between layers is vital to the power of neural models (Bengio, 2009). The relative contribution of these changes, however, is unclear, as is the question of whether gains can be made by introducing non-linearity to conventional feature-based models. In this thesis, we illustrate the close relationship between CRF and sentence-level neural networks, and conduct an empirical investigation of these questions.

Lastly, the main focus of this thesis is to investigate alternative sources of knowledge and signal for semi-supervised learning of NER models. It is well-known that the performance of supervised learners increases when more labeled training examples become available. In most application scenarios, however, manually labeled data are extremely limited in quantity and manual curation of annotated corpora is a costly and time consuming process. Another problem is that no other language has as many resources as English, and many languages have few or no supervised language resources available, which hinders the adoption of supervised learning methods in many multilingual environments.

A potential solution is to leverage the vast amount of freely available unannotated data we have. One would expect to greatly increase the coverage of a system if such large amounts of additional data can be incorporated in a judicious manner. Significant progress has been made in developing unsupervised and semi-supervised approaches to leverage unannotated data to improve system performances (Collins and Singer 1999; Klein 2005; Liang 2005; Smith 2006; Goldberg 2010; *inter alia*). More recent paradigms for semi-supervised learning allow modelers to directly encode knowledge about the task and the domain as constraints to guide learning (Chang et al., 2007; Mann and McCallum, 2010; Ganchev et al., 2010).

[4]When we speak of a linear architecture, we are referring to *generalized linear models* (GLM) (McCullagh, 1984) which includes log-linear models.

Most previous semi-supervised work, however, is situated in a monolingual setting where all unannotated data are available only in a single language. In a multilingual setting where we are learning NER models for many languages, coming up with effective constraints for multilingual semi-supervised learning requires the NLP practitioner to command extensive knowledge of each language, which is an infeasible feat for most.

Bitext — bilingual text that contains human-generated translations between two languages — lends itself as a valuable resource to transfer linguistic constraints from a resource-rich language like English to a foreign languages. Since human translation in general preserves semantic equivalence, bitext offers two different views on the same semantic content (Burkett et al., 2010b). As a result, we can find complementary cues in the two languages that help to disambiguate named entity mentions (Brown et al., 1991). For example, the English word "Jordan" could be referring to either a person name (e.g., *Michael Jordan*) or to a location (i.e., the *Hashemite Kingdom of Jordan*, an Arabic kingdom in the middle east). Without sufficient context it can be difficult to distinguish the two. But in Chinese, these two senses take distinct forms and are naturally disambiguated: "乔丹" as a last name, and "约旦" as a country name.

Thanks to advances in SMT research, a vast amount of translated parallel texts have been made publicly available in recent years. Although the primary use of bitext so far has been in training SMT systems, recent research has increasingly considered the possibilities of utilizing parallel corpora to improve systems outside of SMT. Previous work such as Li et al. (2012a) and Kim et al. (2012) successfully demonstrated that manually-labeled bilingual corpora can be used to improve monolingual system performance. This approach, however, encounters the difficulty that manually annotated bilingual corpora are even harder to come by than monolingual ones.

In this thesis, we first propose a semi-supervised learning scheme using unannotated *bitext*. For a given language pair (e.g., English-Chinese), we expect one language (e.g., English) to have more annotated training resources than the other (e.g., Chinese), and thus there exists a strong monolingual model (for English) and a weaker model (for Chinese). Since bitext contains translations across the two languages, an aligned sentence pair would exhibit some semantic and syntactic similarities. Thus we can constrain the two models to agree with each other by making joint predictions that are skewed towards the more

informed model. In general, errors made in the lower-resource model will be corrected by the higher-resource model, but we also anticipate that these joint predictions will have higher quality for both languages than the output of a monolingual model alone. We can then apply this bilingual annotation method to a large amount of unannotated bitext, and use the resulting annotated data as additional training data to train a new monolingual model with better coverage.[5] Formally, we develop a bilingual NER model by embedding two monolingual CRF-based NER models into a larger undirected graphical model, and introduce additional edge factors based on word alignment (WA) that captures agreements between the two languages. The new bilingual model contains many cyclic cliques, and poses a challenge in performing efficient decoding over the model, since exact inference is intractable.

To tackle this challenge, we first introduce a joint inference method that formulates the bilingual NER tagging problem as an Integer Linear Program (ILP), and solves it using a relaxed LP solver. Experimental results show that the new bilingual model significantly improves bilingual tagging performance, but the ILP decoding process is slow in practice. We further propose an approximate inference method based on Gibbs sampling that improves decoding speed significantly. We demonstrate that it is possible to further improve tagging performance by modeling document-level consistency in the Gibbs sampler, in light of Finkel et al. (2005).

Since the bilingual NER model relies greatly on automatically induced word alignments to enforce agreements across language boundaries, alignment errors committed by the word aligner have a direct impact over the tagging model performance. We observe that the entity span and type predictions given by the NER models actually contain complementary information that could be used to correct alignment errors. To capture this source of information, we present an extension that combines the bilingual NER model with two uni-directional HMM-based alignment models, and jointly improves both NER and word alignment performances using a new approximate inference method called dual decomposition.

A premise of the aforementioned semi-supervised learning method with bitext is that we have access to a pre-trained "weak" model for the target language. Consequently, this

[5]This training regimen is also referred to as "uptraining" (Petrov et al., 2010).

requirement prevents the application of this method to languages that have no available annotated training data. To overcome this limitation, we propose a new cross-lingual projection based method, similar to Yarowsky and Ngai (2001), but different in two important ways. First, we never explicitly project the labels. Instead, we project expectations over the labels. This pseudo-projection acts as a soft constraint over the labels, which allows us to transfer more information and uncertainty across language boundaries. Secondly, we encode the expectations as constraints and train a model by minimizing divergence between model expectations and projected expectations in a Generalized Expectation (GE) Criteria (Mann and McCallum, 2010) framework.

1.2 Contributions of this thesis

This thesis presents four main contributions to the field of natural language processing.

- We present two new methods that address the weight undertraining problem that commonly occur in log-linear model training, and demonstrate significant improvements over standard regularization methods.

- We conduct both theoretical and empirical investigations into the connection between model architecture and feature representation for sequence tagging tasks, and answer important questions such as whether linear models are sufficient for sequence labeling NLP problems, or are there benefits to be sought in moving to non-linear models.

- We propose a novel paradigm for using bilingual text to improve monolingual system performance, based on joint decoding of bilingual models. We propose a suite of agreement-based bilingual models under the new paradigm, and demonstrate superior performance to both past methods using bilingual text and state-of-the-art monolingual models. To address the computational challenge with the bilingual models, we present three approximate inference methods that makes efficient decoding feasible over this model.

- We address scenarios where we want to bootstrap NER models for new languages

where we do not have any annotated training by proposing a novel cross-lingual projection method. Our method projects soft labels instead of hard labels, and models uncertainties in a expectation regularization framework, yielding significantly improvements over past projection-based methods.

1.3 Structure of this thesis

Chapter 2 This chapter gives the necessary technical background for readers to be able to follow the remainder of the thesis. We first introduce the task of named-entity recognition as well as foundational log-linear models such as MaxEnt and Conditional Random Fields. We then describe the experiment setup, dataset, feature templates, and evaluation methods.

Chapter 3 This chapter first gives an introduction to the problem of feature weight undertraining, a common problem that affects log-linear model performance. We then review some of the past models that aim to mitigate this problem, including various product-of-experts models, and feature noising methods. We then present a novel use of a mixed $\ell_2\ell_1$ norm for automatically learning the structure of product-of-experts models, and how to efficiently simulate training with artificially noised features in the context of log-linear structured prediction, without actually having to generate noised data. The contents of this chapter are based on Wang and Manning (2013b) and Wang et al. (2013c).

Chapter 4 This chapter challenges the conventional wisdom of employing linear architecture and sparse discrete feature representation for sequence labeling tasks, and closely examines the connection and tradeoff between linear versus non-linear architectures, as well as discrete versus continuous feature representations. The contents of this chapter are based on Wang and Manning (2013a).

Chapter 5 This chapter aims to improve NER system performance from a new angle by asking the question—can we leverage the vast amount of unlabeled bilingual text to improve monolingual system performance? We show a factored bilingual NER model using two monolingual CRFs and many cross-lingual tag assignment agreement models

over automatically induced word alignments that can significantly improve bilingual tagging performance. Additionally, we show monolingual NER where decoded bitext is used as additional training data to obtain significant performance gain. We describe in detail three decoding methods—Integer Linear Programming (ILP), Gibbs sampling, and dual decomposition—for performing approximate inference over the bilingual NER model. The contents of this chapter are based on Wang et al. (2013a), Wang et al. (2013b) and Che et al. (2013).

Chapter 6 This chapter describes a cross-lingual projection method that can be applied to a minimally supervised setting where we only have annotated training data for a source language (e.g., English) but none for the target language. Most existing projection-based methods project hard labels across word alignment links over bitext, we instead project the source model's posteriors as soft labels; we then train the target side model under the expectation regularization framework to allow more uncertainties in model training, which has the same effect as marginalizing out unknown word positions. We evaluate this method in both a minimally supervised setting as well as the semi-supervised setting as in Chapter 4, and demonstrate its superior performance over strong baselines and previous methods. The contents of this chapter are based on Wang and Manning (2014).

Chapter 7 In this chapter we give a conclusion of the work conducted in this thesis, and provide points for several potential future work directions.

Chapter 2

Background

This chapter gives a background introduction to sequence labeling and the Named Entity Recognition task. We give definition and implementational details about the baseline MaxEnt and CRF models, as well as the evaluation method and dataset used for the experiments. The same experimental setup is used throughout most of this thesis. We will also explain the batch and stochastic optimization algorithms employed for training these models.

2.1 Named Entity Recognition

An important class of NLP problems take a raw sentence as input, and aim at assigning syntactic or semantic categorical tags to each word in the sentence. Such problems are commonly referred to as *sequence labeling*, and are important first steps in enabling downstream applications such as Question Answering and Relation Extraction. Examples of sequence labeling tasks include part-of-speed tagging, where words are assigned grammatical category tags such as NN (i.e., noun) and VB (i.e. verb), and semantic role labeling, which attempts to assign higher-level semantic tags such as AGENT, PATIENT, etc., [1] which marks the semantic roles of words.

Named Entity Recognition is another important sequence labeling task. Given an input sentence, an NER tagger identifies words that are part of a named entity, and assigns the

[1] The more commonly seen semantic roles are the PropBank (Palmer et al., 2005) style numbered arguments, such as ARG0, ARG1, etc..

Vice_{O} $\text{Foreign}_{B\text{-}ORG}$ $\text{Affairs}_{I\text{-}ORG}$ Minister_{O} $\text{Huaqiu}_{B\text{-}PER}$ $\text{Liu}_{I\text{-}PER}$ held_{O} talks_{O} with_{O} $\text{Kamyao}_{B\text{-}PER}$

外交部$_{B\text{-}ORG}$ 副部长$_{O}$ 刘华秋$_{B\text{-}PER}$ 与$_{O}$ 加米奥$_{B\text{-}PER}$ 举行$_{O}$ 了$_{O}$ 会谈$_{O}$

Figure 2.1: Examples of NER tagged sentences in English and Chinese.

entity type and relative position information. For example, in the commonly used `BIO` tagging scheme, a tag such as `B-PERSON` indicates the word is at the beginning position of a *person name* entity; and a `I-LOCATION` tag marks the word to be inside a *location* entity. All words marked with tag `O` are not part of any entity. Figure 2.1 illustrates two tagged sentences in English and Chinese. The `B-PER` tag in these examples is abbreviation for `B-PERSON`.

2.2 Evaluation

Evaluating NER systems is relatively straight-forward given gold-standard annotations, typically generated by human expert annotators. Since most of the words in a sentence do not have entity types, accuracy is not particularly suited to this task (simply predicting everything to be `O` would get a pretty high accuracy). Instead, we compute the precision, recall and F_1 measure. There are two levels of granularity at which we can compute these test statistics: word level or token level. At the word level, we count the number of words whose predicted tags are correct (*true positives*), the number of words with wrong prediction (*true negatives*), and the number of entity-typed words that the system missed (*false positives*). Precision is then computed as:

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{true negatives}}$$

which captures the percentage of predictions that are correct. Recall is computed as:

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

which characterizes the percentage of true entities the system finds. And the F_1 measure is the harmonic mean of precision and recall, formally shown as:

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

The word level measure is easy to understand and naturally suited to many sequence labeling tasks such as POS tagging. But for NER, it can be too lenient because it rewards partially correct assignments. For example, let us assume the correct assignment for an example phrase is the following:

$$\text{The}_{\text{O}}\ \text{inner}_{\text{B-LOC}}\ \text{Mongolia}_{\text{I-LOC}}\ \text{autonomous}_{\text{I-LOC}}\ \text{region}_{\text{I-LOC}}$$

And a system outputs the following assignments:

$$\text{The}_{\text{O}}\ \text{inner}_{\text{B-LOC}}\ \text{Mongolia}_{\text{O}}\ \text{autonomous}_{\text{I-LOC}}\ \text{region}_{\text{I-LOC}}$$

According to the word-level evaluation, this system would receive a precision of 1.00 and a recall of 0.75, which clearly does not reflect the quality of the system: not only does it miss the most important word (*Mongolia*), it also erroneously produces two location entities (*inner*, and *autonomous region*) instead of one.

A more appropriate and stricter set of measures are the entity-level metrics, where we still compute precision, recall and F_1 score, except that the calculation is per entity and an entity is considered correct if all words in that entity are labeled correctly. In this case, the previous example would be scored 0.00 for both precision and recall. We will employ entity-level evaluation for all experiments in this thesis.

Statistical significance tests for comparing outputs from different systems are done using the paired bootstrap resampling method (Efron and Tibshirani, 1993), where we repeatedly draw random samples with replacement from the output of the two systems, and compare the test statistics (e.g. absolute difference in F_1 score) of the new samples with the observed test statistics. We used 1000 sampling iterations in our experiments.

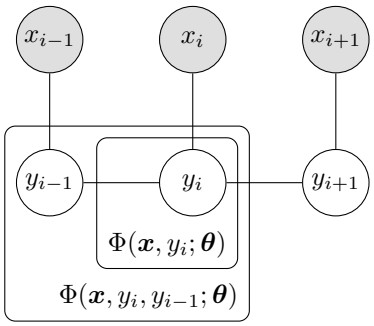


Figure 2.2: A simple linear-chain CRF with node and edge clique potentials.

2.3 Log-Linear Models for Structure Prediction

Current state-of-the-art supervised NER systems employ an undirected graphical model called as Conditional Random Field (CRF) (Lafferty et al., 2001).

Formally, a CRF models the conditional probability of structured output variables $\boldsymbol{x}$ given observations $\boldsymbol{x}$. In sequence modeling, the observations are typically words in a sentence, and the output variables are some syntactic or semantic tags we are trying to predict for each word (e.g., POS, named-entity tags, etc.). The most commonly used CRF model has a linear chain structure, where prediction y_i at position i is independent of other predictions, given its neighbors y_{i-1} and y_{i+1}. It is customary to describe the model as an undirected graphical model, as shown in Figure 2.2, with the following probability definition:

$$P_{CRF}(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\prod_{i=1}^{|\boldsymbol{x}|} \Phi(\boldsymbol{x}, y_i, y_{i-1}; \boldsymbol{\theta})}{Z(\boldsymbol{x}; \boldsymbol{\theta})} \tag{2.1}$$

$\Phi(\boldsymbol{x}, y_i, y_{i-1})$ denotes a clique potential function at position i, which defines the unnormalized probability mass assigned to state y_i and y_{i-1}. $Z(\boldsymbol{x}; \boldsymbol{\theta})$ is the partition function that

sums over all possible assignments of output variables in the entire sequence, defined as:

$$Z(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{\boldsymbol{y'}} \prod_{i=1}^{|\boldsymbol{x}|} \Phi(\boldsymbol{x}, y'_i, y'_{i-1}; \boldsymbol{\theta})$$

where y' represents all possible label sequence assignments. The partition function ensures CRFs are global normalized—assignments to parts of the sequence will influence the probability assignment of other parts of the sequence.

The clique potential function $\Phi(\boldsymbol{x}, y_i, y_{i-1}; \boldsymbol{\theta})$ is defined as the following:

$$\Phi(\boldsymbol{x}, y_i, y_{i-1}; \boldsymbol{\theta}) = \exp\{\boldsymbol{f}(y_i, y_{i-1}, \boldsymbol{x}) \cdot \boldsymbol{\theta}\} \tag{2.2}$$

$\boldsymbol{f}(y_i, y_{i-1}, \boldsymbol{x}) \in \mathbb{R}^d$ is a feature function that can utilize all or parts of the information contained in $(y_i, y_{i-1}, \boldsymbol{x})$ to compute a feature vector of dimension d for a clique, and $\boldsymbol{\theta} \in \mathbb{R}^d$ is the corresponding feature weight vector (i.e., parameters in the model). We use $f_j(y_i, y_{i-1}, \boldsymbol{x})$ and θ_j to denote the jth feature value and weight.

Note that this definition of first-order linear-chain CRF can be easily generalized to higher-order CRF models, by extending the size of the clique modeled in potential function Φ. Although higher-order CRFs have been proven useful in many applications such as computer vision (Kohli et al., 2009), its contribution on top of the utility already offered by linear-chain CRFs in tasks such as NER is only marginal. But a significant drawback of higher-order CRFs is that they significantly increase the difficulty of performing learning and inference over these models. In this thesis, we adopt the linear-chain CRFs as basic building blocks of our models, and focus our exploration in areas other than the Markov order properties of the models.

Another important note is that in practice, many of the features in $\boldsymbol{f}(y_i, y_{i-1}, \boldsymbol{x})$ are defined over only the node label (y_i) but not the edge label (pair y_i and y_{i-1}). At times it is desirable to make a distinction between the node features and edge features (*cf.* Section 4.3). We will denote the node features as $\boldsymbol{g}(y_i, \boldsymbol{x})$ and the edge features as $\boldsymbol{h}(y_i, y_{i-1}, \boldsymbol{x})$, where $\boldsymbol{f}(y_i, y_{i-1}, \boldsymbol{x}) = \boldsymbol{g}(y_i, \boldsymbol{x}) + \boldsymbol{h}(y_i, y_{i-1}, \boldsymbol{x})$. An alternative formulation to explain this distinction between node and edge features is by explicitly separating out the node cliques

from the edge cliques:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{\prod_{i=1}^{|\boldsymbol{x}|} \Psi(\boldsymbol{x}, y_i; \boldsymbol{\theta}) \times \Phi(\boldsymbol{x}, y_i, y_{i-1}; \boldsymbol{\theta})}{Z(\boldsymbol{x})} \tag{2.3}$$

where

$$\Psi(\boldsymbol{x}, y_i; \boldsymbol{\theta}) = \exp\{\boldsymbol{g}(y_i, \boldsymbol{x}) \cdot \boldsymbol{\theta}\}$$
$$\Phi(\boldsymbol{x}, y_i, y_{i-1}; \boldsymbol{\theta}) = \exp\{\boldsymbol{h}(y_i, y_{i-1}, \boldsymbol{x}) \cdot \boldsymbol{\theta}\}$$

This formulation will become handy when we explore the linearity property of CRF architecture in Chapter 4.

If there are no edge features in the model, then a linear-chain CRF reduces down to a product of MaxEnt (i.e., logistic regression) classifiers, one at each position in the sequence. The following derivation illustrates this connection: Under the assumption, we have $\boldsymbol{f}(y_i, y_{i-1}, \boldsymbol{x}) = \boldsymbol{g}(y_i, \boldsymbol{x})$, then:

$$Z(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{\boldsymbol{y'}} \prod_{i=1}^{|\boldsymbol{x}|} \Phi(\boldsymbol{x}, y'_i; \boldsymbol{\theta})$$

And then we have:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{\prod_{i=1}^{|\boldsymbol{x}|} \Phi(\boldsymbol{x}, y_i)}{\prod_{i=1}^{|\boldsymbol{x}|} \sum_{y'_i} \Phi(\boldsymbol{x}, y'_i)} = \prod_{i=1}^{|\boldsymbol{x}|} \frac{\Phi(\boldsymbol{x}, y_i)}{\sum_{y'_i} \Phi(\boldsymbol{x}, y'_i)}$$

which is a product of locally normalized MaxEnt models whose predictions are independent of each other.

2.4 Parameter Training

Training a CRF model results in finding the feature weights $\boldsymbol{\theta}$ that maximize the conditional probabilities of observed label sequences $\boldsymbol{y}$. One nice property of log-linear models like CRFs is that computation is much easier in the log-domain. Instead of maximizing the original likelihood objective, we opt to maximize the log-likelihood instead:

$$\begin{aligned}\mathcal{L}(\boldsymbol{y}|\boldsymbol{x}) &= \sum_{i=1}^{|\boldsymbol{x}|} \log \Phi(\boldsymbol{x}, y_i, y_{i-1}; \boldsymbol{\theta}) - \log(Z(\boldsymbol{x})) \\ &= \sum_{i=1}^{|\boldsymbol{x}|} \boldsymbol{f}(y_i, y_{i-1}, \boldsymbol{x}) \cdot \boldsymbol{\theta} - \log(Z(\boldsymbol{x}))\end{aligned}$$

The second term in the log-likelihood objective involves taking the log of the partition function, which is intractable if computed naively by enumerating over all possible label sequences. Fortunately, by exploring the Markov assumptions in the model, we can devise efficient dynamic-programming algorithms to compute this term, similar to the forward-backward algorithms developed for the hidden Markov models (Rabiner and Juang, 1986) in the 80's. Details of the algorithm are well-understood and widely available, we will omit them here for brevity.[2]

To prevent the model from overfitting, it is standard practice to apply a Gaussian prior (i.e. L2 regularization) to the objective, which basically states that feature weights are expected to be zero a priori, and large feature weights will be penalized more heavily according to a Gaussian distribution. The updated objective function with Gaussian prior looks like the following:

$$\mathcal{L}(\boldsymbol{y}|\boldsymbol{x}) = \sum_{i=1}^{|\boldsymbol{x}|} \boldsymbol{f}(y_i, y_{i-1}, \boldsymbol{x}) \cdot \boldsymbol{\theta} - \log(Z(\boldsymbol{x})) - \frac{\boldsymbol{\theta}^2}{2\sigma^2}$$

where σ is the variance parameter that controls the regularization strength. Note that we assign a global σ to all parameters and do not attempt to set a separate σ_j for each weight parameter θ_j. However, L2 regularization is not the most effective regularization techniques

[2]Michael Collins gave a more recent tutorial on forward-backward algorithms, accessible from here: `www.cs.columbia.edu/~mcollins/fb.pdf`.

in some settings, especially in dealing with a phenomena known as feature co-adaptation. In Chapter 3, we will explore the use of more sophisticated regularization methods such as mixed L2/L1 norms and feature noising regularization.

Learning the parameter values $\boldsymbol{\theta}$ does not have an analytic solution. The values are found by numerical optimization methods, which require computing the partial derivative of this objective function with respect to each parameter [3]:

$$\begin{aligned}\frac{\partial}{\partial\theta_j}\mathcal{L}(\boldsymbol{y}|\boldsymbol{x}) &= \sum_{i=1}^{|\boldsymbol{x}|} f_j(y_i, y_{i-1}, \boldsymbol{x}) - \frac{\partial}{\partial\theta_j}\log(Z(\boldsymbol{x})) - \frac{\theta_j}{\sigma^2} \\ &= \sum_{i=1}^{|\boldsymbol{x}|} f_j(y_i, y_{i-1}, \boldsymbol{x}) - \mathbb{E}_{\boldsymbol{\theta}}[f_j|\boldsymbol{x}] - \frac{\theta_j}{\sigma^2}\end{aligned}$$

Update rules based on this partial derivative have a very intuitive interpretation: the update we should apply to a feature amounts to the difference between the observed counts and the expected counts under the current model, subject to the Gaussian prior regularization. Once we have the partial derivatives, we can choose among a great variety of numerical optimization techniques, including L-BFGS (Liu and Nocedal, 1989) and the Adaptive Sub-gradient method (Duchi et al., 2011).

At test time, we need to perform decoding to find the best label sequence under the current model for a given input sentence. This can also be done efficiently using Viterbi algorithm (Viterbi, 1967).

2.5 Dataset

There are three different experimental settings explored in this thesis.

2.5.1 Monolingual NER

The first experimental setting only concerns monolingual NER for English (Chapter 3, 4). Under this setting, we evaluate system performance over two standard benchmark datasets.

[3] A detailed derivation can be found in Noah Smith's notes: `http://www.cs.cmu.edu/~nasmith/papers/smith.tut04.pdf`

The first dataset is the CoNLL-03 English shared task benchmark dataset (Sang and Meulder, 2003), which is a collection of documents from Reuters newswire articles, annotated with four entity types: *Person*, *Location*, *Organization*, and *Miscellaneous*. We adopt the `BIO2` annotation standard (Sang and Veenstra, 1999). Beginning and intermediate positions of an entity are marked with *B-* and *I-* tags, and non-entities with *O* tag. The training set contains 204K words (14K sentences), the development set contains 51K words (3.3K sentences), and the test set contains 46K words (3.5K sentences).

To evaluate out-of-domain performance, we run the models trained on CoNLL-03 training data on two additional test datasets. The first dataset (ACE) is taken from the ACE Phase 2 (2001-02) and ACE-2003 data (Doddington et al., 2004). Although the ACE dataset also consists of newswire text and thus is not strictly out-of-domain, there is a genre or dialect difference in that it is drawn from mostly American news sources, whereas CoNLL is mostly from British news sources. The test portion of this dataset contains 63K words, and is annotated with 5 original entity types: *Person*, *Location*, *Organization*, *Facility*, and *GPE*. Notice that this label set does not fully agree with the label set from the CoNLL-03 dataset from which the models in question are trained. In order to unify the two label sets, we remove all entities of type *Facility* and *GPE* by relabeling them as *O* during preprocessing, and discard entity tags of type *Miscellaneous* in the output of the models.

The second dataset is from the 6th and 7th meeting of the Message Understanding Conferences (MUC-6/7) (Grishman and Sundheim, 1996; Chinchor, 1998). The combined dataset contains 59K words for testing, all drawn from American newswire text. The MUC data is annotated with 7 entity types. It is missing the *Miscellaneous* entity type, but includes 4 additional entity types that do not occur in CoNLL-2003: *Date*, *Time*, *Money*, and *Percent*. We converted the data to CoNLL-03 type format using the same method applied to the ACE data.

2.5.2 Bilingual NER

Our second set of experiments takes a turn from monolingual NER settings towards a bilingual setting, where the system performs NER over a pair of sentences written in two languages that are translations of each other. The purpose of this design is to explore the

cross-lingual constraints encoded in translations to improve the performance of monolingual models, which we will explain in detail later in Chapter 5.

For bilingual tagging experiments, we need to evaluate on an annotated parallel corpus that contains named entity annotation in both Chinese and English. The corpus we use is the latest version (v4.0) of the OntoNotes corpus (Hovy et al., 2006), which is a large, manually annotated corpus containing 401 pairs of Chinese and English documents (chtb_0001-0325, ectb_1001-1078). We use odd-numbered documents as the development set and even-numbered documents as the blind test set.

Out of the 18 named entity types that are annotated in OntoNotes, which include person, location, date, money, and so on, we select the four most commonly seen named entity types for evaluation. They are *Person*, *Location*, *Organization* and *GPE*. All entities of these four types are converted to the standard BIO format, and background tokens and all other entities types are marked with tag `O`.

These document pairs are aligned at the document level, but not at sentence or word level. To obtain sentence alignment, we use the Champollion Tool Kit (CTK).[4] After discarding sentences with no aligned counterpart, a total of 8,249 sentence pairs were retained. We induce automatic word alignment using the BerkeleyAligner toolkit (Liang et al., 2006).[5] The aligner outputs the posterior probability for each aligned word pair. To increase efficiency, we prune away all alignments that have probability less than 0.1.

At test time, we re-group the sentences that belong to the same original document into documents, and treat a whole document as an input "sentence" to the model. The reason for this particular setup is because we would like to evaluate the effect of modeling global document-level consistency (*cf.* Section 5.3.3).

We train two CRF baseline models on all portions of the OntoNotes corpus that are annotated with named entity tags, except the parallel-aligned portion which we reserve for development and test purposes. In total, there are about 660 documents ($\sim$16k sentences) and 1,400 documents ($\sim$39k sentences) for Chinese and English, respectively. Although the English portion of OntoNotes's training set is almost three times the size of the CoNLL-03 training set, the test portion of OntoNotes is harder than CoNLL-03. A baseline English

[4] `champollion.sourceforge.net`

[5] `code.google.com/p/berkeleyaligner`

CRF model trained and tested on OntoNotes dataset gives F_1 score of 78.11%, whereas on CoNLL-03 the same model achieves F_1 score of 85.82%.

2.5.3 Semi-supervised NER

Building on top of the bilingual tagging experiments described in the previous setting, we are also interested in a semi-supervised learning setting where we take the output of the bilingual tagger over a large unannotated bilingual corpus, and use the tagged sentences as additional training data to retrain a monolingual tagger.

To facilitate the Chinese-English semi-supervised experiments, we use the Foreign Broadcast Information Service corpus (FBIS, LDC2003E14) as unannotated bitext. This dataset is very large, processing the whole dataset will take a very long time. Therefore, we randomly sample a subset of 80,000 sentences.

In addition to the Chinese-English experiments, we add a new language pair of German-English. We used the German-English portion of the *News Commentary*[6] data from the 13th Workshop on Statistical Machine Translation (WMT-13) as unannotated bitext, from which we also sample a 80,000 sentences subset. The retrained monolingual German tagger is then evaluated on the German portion of the CoNLL-03 corpus.

Unlike the experiments described in the previous section, for semi-supervised NER experiments, we use each individual sentence as input to NER model, instead of whole documents.

2.5.4 Summary

Table 2.1 gives a summary of the datasets used in this thesis.

2.6 Feature Templates

The feature templates used in our English and Chinese NER experiments are summarized in Table 2.2. $\circ$ means string concatenation and y_i is the named entity label of the i^{th} word

[6] `http://www.statmt.org/wmt13/training-parallel-nc-v8.tgz`

Monolingual

Corpus	Lang	# Train Tokens	# Dev Tokens	# Test Tokens
CoNLL-03	en	203K	51K	46K
	de	207K	51K	52K
MUC-6/7	en	-	-	59K
ACE	en	-	-	63K

Bilingual

Corpus	Lang	# Train Tokens	# Dev Tokens	# Test Tokens
OntoNotes	en	706K	156K	165K
	zh	313K	120K	126K

Semi-Supervised

Corpus	Lang	# Bitext Tokens
FBIS	en	3300K
	zh	2535K
WMT-13	en	2022K
	de	2061K

Table 2.1: Summary statistics for the datasets used in this thesis

w_i. $\mathsf{len}(w_i)$ is the number of Chinese characters in w_i. $\mathsf{radical}(w_i, k)$ denotes the radical of the k^{th} Chinese character. These features can be grouped into three categories: generic features, language-specific features, and distributional similarity features.

2.6.1 Generic Features

The most basic features for NER are lexical features that examine the current word and neighboring words (01 in Table 2.2). These features give the model the ability to memorize word-label pairs seen in training data, as well as contextual clues such as "the word that appears before phrase *met with* is likely to be person". We also add a class bias feature that captures the relative likelihood that a particular label occurs (00).

	English Templates	Chinese Templates
00:	1 (class bias param)	1 (class bias param)
01:	$w_{i+k}, -1 \leq k \leq 1$	$w_{i+k}, -1 \leq k \leq 1$
02:	$\mathsf{shape}(w_{i+k}), -1 \leq k \leq 1$	$w_{i+k-1} \circ w_{i+k}, 0 \leq k \leq 1$
03:	$w_i \circ \mathsf{shape}(w_{i-1})$	$\mathsf{shape}(w_{i+k}), -4 \leq k \leq 4$
04:	$w_{i-1} \circ w_{i-2} \circ \mathsf{shape}(w_{i-1})$	$\mathsf{prefix}(w_i, k), 1 \leq k \leq 4$
05:	$\{w_{i+k}\}, -4 \leq k \leq 4$	$\mathsf{prefix}(w_{i-1}, k), 1 \leq k \leq 4$
06:	$\mathsf{cap}(w_i - 1) \circ \mathsf{cap}(w_i) \circ \mathsf{cap}(w_i + 1)$	$\mathsf{suffix}(w_i, k), 1 \leq k \leq 4$
07:	$\mathsf{title}(w_i, k), -1 \leq k \leq 0$	$\mathsf{suffix}(w_{i-1}, k), 1 \leq k \leq 4$
08:	$\mathsf{distsim}(w_{i+k}), -1 \leq k \leq 1$	$\mathsf{radical}(w_i, k), 1 \leq k \leq \mathsf{len}(w_i)$
09:		$\mathsf{distsim}(w_{i+k}), -1 \leq k \leq 1$
Node Features $y_i \circ$ 00 – 09		
Edge Features $y_{i-1} \circ y_i \circ$ 00 – 09		

Table 2.2: NER feature templates.

2.6.2 Language Specific Features

Each language has its own idiosyncrasies that cannot be captured by simple lexical features. We adopt a specific set of features that aim to provide better generalization over morphological and orthographical variations based on (Collins, 2002). The $\mathsf{shape}(w_i)$ feature morphs a word into a more abstract shape form, and was proposed in (Finkel et al., 2005). For example, we replace all lowercase letters with the letter "x", and all uppercase letters with the letter "X", then words such as "Tony" and "John" will have the same shape "Xxxx". We can also replace all digits with the letter "d", then two different dates such as "2001-10-23" and "2012-01-03" would have the same shape "dddd-dd-dd". Since capitalization provides a very strong clue for English NER, we design a specific feature just to capture capitalization ($\mathsf{cap}(w_i)$). Other English-specific features include $\mathsf{title}(w_i)$ features that match the current word w_i against a list of name titles (e.g., Mr., Mrs., Dr., etc.). For Chinese, we add features to examine the character prefix and suffix pattern of a word, since there are strongly indicative patterns such as "局" and "先生". Other Chinese specific features include radical features, which are quite limited in their utility but experimentally give minor improvements on development set performance.

Overall, a total number of 507,151 features were generated on the CoNLL-2003 English training data, and 765,289 features were generated on the OntoNotes Chinese training data. The total number of model parameters for English and Chinese is and 7,880,933 and 11,596,945, respectively.

2.6.3 Distributional Similarity Features

Distributional similarity features based on word clusters have been proven to be very effective in improving model performance for a variety of tasks, including NER (Miller et al., 2004; Finkel et al., 2005; Turian et al., 2010) and dependency parsing (Koo et al., 2008). Finkel et al. (2005) showed a 10%–25% reduction in error when adding word cluster features to a standard CRF-based NER tagger.

In our experiments, we adopt the word clusters used by (Finkel et al., 2005) for English. The clusters were trained using the clustering algorithm from (Clark, 2003) on a 275 million word corpus, and contains 200 clusters.

For Chinese, we used a C++ implementation of the Brown word clustering algorithms (Brown et al., 1992) from (Liang, 2005)[7] for inducing word clusters. Raw text was obtained from the fifth edition of Chinese Gigaword (LDC2011T13). One million paragraphs from the Xinhua news section were randomly selected, and the Stanford Word Segmenter with LDC standard was applied to segment Chinese text into words.[8] About 46 million words were obtained which were clustered into 1,000 word classes.

[7] github.com/percyliang/brown-cluster

[8] nlp.stanford.edu/software/segmenter.shtml

Chapter 3

Addressing Weight Undertraining

3.1 Introduction

Conditionally trained discriminative models like logistic regression (a.k.a., MaxEnt) models and CRFs gain power from their ability to incorporate a large number of arbitrarily overlapping features. In comparison, generative models like Naive Bayes does not work well when the features overlap a lot and we need to weight each feature's contribution to maximize likelihood. But such models also exhibit complex feature dependencies and therefore are susceptible to the problem of feature co-adaptation—the contributions of certain features are overlooked because of features they co-occur with (Sutton et al., 2007; Hinton et al., 2012). The problem of weight undertraining occurs as a consequence of feature co-adaptation in discriminatively trained classifiers when the strongly indicative features that overruled the weak features during training do not fire at test time. Because of the presence of such strong features during training, weaker features that occur at both training and test time do not receive enough weight, and thus impede the classifier from reaching its full potential at test time. Sutton et al. (2007) gave an excellent demonstration of this problem using a synthetic logistic regression model. In their example, the output function is the softmax of the linear sum of a set of weights x_1 to x_n, where each of the x_i acts as a "weak" predictor. A "strong" feature which takes the form of $x' = \sum_{i=1}^{n} x_i + \mathcal{N}$ (where $\mathcal{N}$ is an added Gaussian noise) is then added to the classifier during training but removed at test time, which simulates a feature co-adaptation scenario. Simulation results show that

classifier accuracy can drop by as much as 40% as a result of weight undertraining.

This problem is difficult to combat because we do not know a priori what set of features will be present at test time. Also, balancing the tradeoff between overfitting and undertraining is also challenging.

Over the years, a number of methods including product-of-experts (e.g., feature bagging (Sutton et al., 2007) and logarithmic opinion pooling (Heskes, 1998; Smith et al., 2005)) and feature noising (Hinton et al., 2012; Wang and Manning, 2013c; Wager et al., 2013) have been introduced as potential ways to regulate model learning and alleviate weight undertraining.

This chapter makes two main contributions. First, we observe that performance of existing product-of-experts methods such as logarithmic opinion pooling (LOP) are directly impacted by the quality and diversity of the experts; however, existing expert induction procedures remain knowledge-intensive and manual. We present one attempt at designing a principled way to automatically induce experts, through the use of a novel mixed $\ell_2\ell_1$ regularizer. Second, we introduce some very recent new methods for training log-linear models with artificially corrupted features as a way to regularize weight learning (Wager et al., 2013). We then show how to extend this approach to efficiently simulate training with noise in the context of structure prediction, specifically for linear-chain CRF models.

We show that both of these methods are effective at reducing the bad effects of weight undertraining, and do so more automatically than previous approaches.

3.2 Automatic Learning of Products of Experts

One popular method that attempts to mitigate weight undertraining is logarithmic opinion pooling (LOP) (Heskes, 1998; Smith et al., 2005; Sutton et al., 2007). LOP works in a similar fashion to a product of experts model, in which a number of experts are individually assembled first, and then their predictions are combined multiplicatively to create an ensemble model. Experts are generated by first partitioning the feature space into subsets, then an independent model (expert) is trained on each subset of features (Sutton et al., 2007). Under the assumption that strongly correlated features are partitioned into separate subsets, this method effectively forces the models to learn how to make predictions with

each subset of the features independently. It also helps in scenarios where some strong features stop other features from getting weight, a problem known as weight undertraining. The theoretical justification for favoring a product ensemble over an additive ensemble is that inference with a product of log-linear models is significantly easier than inference with a sum. Sutton et al. (2007) also suggests that product ensembles give better results than additive mixtures on sequence labeling tasks.

Smith et al. (2005) and Sutton et al. (2007) showed that the quality of experts and diversity among them have a direct impact on the final performance of the ensemble. Designing effective and diverse experts was left as an art, and requires a great deal of domain expertise.

It is not hard to see that the more accurate each expert, the more accurate the overall ensemble. It is less obvious that diversity among the experts also helps to improve the ensemble, as was illustrated in Smith and Osborne (2007). In particular, hand-crafted experts that exhibit more diversity are superior to experts created by randomly or sequentially partitioning the feature space. However, one major problem with this approach is that it requires significant amounts of domain expertise and manual effort to come up with effective and diverse experts.

In this section, we directly learn to induce diverse experts by using a mixed $\ell_2\ell_1$ norm known as *elitist lasso* in the context of generalized linear models (Kowalski and Torrésani, 2009). We design a novel competition mechanism to encourage experts to use non-overlapping feature sets, effectively learning a partition of the feature space. Efficient optimization of a maximum conditional likelihood objective with respect to the $\ell_2\ell_1$ norm is non-trivial and has not been previously studied. We propose a novel formulation to incorporate the $\ell_2\ell_1$ norm into the FOBOS optimization framework (Duchi and Singer, 2009). Experiments on our Named Entity Recognition task suggest that our proposed method gives consistent improvements over a baseline MaxEnt model and conventional LOP experts. In particular, our method gives the most improvement in recognizing entity types for out-of-vocabulary words.

3.2.1 Logarithmic Opinion Pooling

The idea behind logarithmic opinion pooling (LOP) is that instead of training one classifier, we can train a set of classifiers with non-overlapping or competing feature sets. At test time, we can combine these classifiers and average their results. By creating a diverse set of classifiers and training each of them separately, we can potentially reduce the effect of weight undertraining. For example, when two strongly correlated features are partitioned into different classifiers, they do not overshadow each other. Unlike voting or feature bagging where an additive ensemble of experts is used, LOP derives a joint model by taking a product of experts.[1]

The conditional probability of a LOP ensemble can be expressed as:

$$P_{LOP}(\boldsymbol{y}|\boldsymbol{x}) = \frac{\prod_{i=1}^{n} [P_i(\boldsymbol{y}|\boldsymbol{x})]^{\alpha_i}}{\sum_{\boldsymbol{y}'} \prod_{i=1}^{n} [P_i(\boldsymbol{y}|\boldsymbol{x})]^{\alpha_i}}$$

$$P_i(\boldsymbol{y}|\boldsymbol{x}) \ \propto \ \sum_{j=1}^{d} \theta_j^i f_j(\boldsymbol{x})$$

where n is the total number of experts.

$P_i(\boldsymbol{y}|\boldsymbol{x})$ is the probability assignment of expert i, in this case we choose each expert to take the form of a MaxEnt classifier (*cf.* Section 2.3). The experts are typically selected so that they capture different signals from the training data (e.g., via feature partitioning). Each expert is trained separately, and their model parameters θ_j^i (denotes the jth parameter of the ith expert) are fixed during LOP combination.

α_i is the mixing coefficient for this expert. Early work on LOPs either fixed the expert mixing coefficients to some uniformly assigned value (Hinton, 1999), or used some arbitrary hand-picked values (Sutton et al., 2007). In both of these two cases, no new parameter values need to be learned, therefore no extra training is required. Smith et al. (2005) proposed to learn the weights by maximizing log-likelihood of the ensemble product model, and showed that the learned weights work better than uniformly assigned values.

We can visualize each individual MaxEnt classifier and the LOP using a neural network style diagram, as shown in the top two parts of Figure 3.1. The top-most figure shows a

[1] A product in the raw probability space is equivalent to a sum in the logarithmic space, and hence the name "logarithmic" opinion pooling.

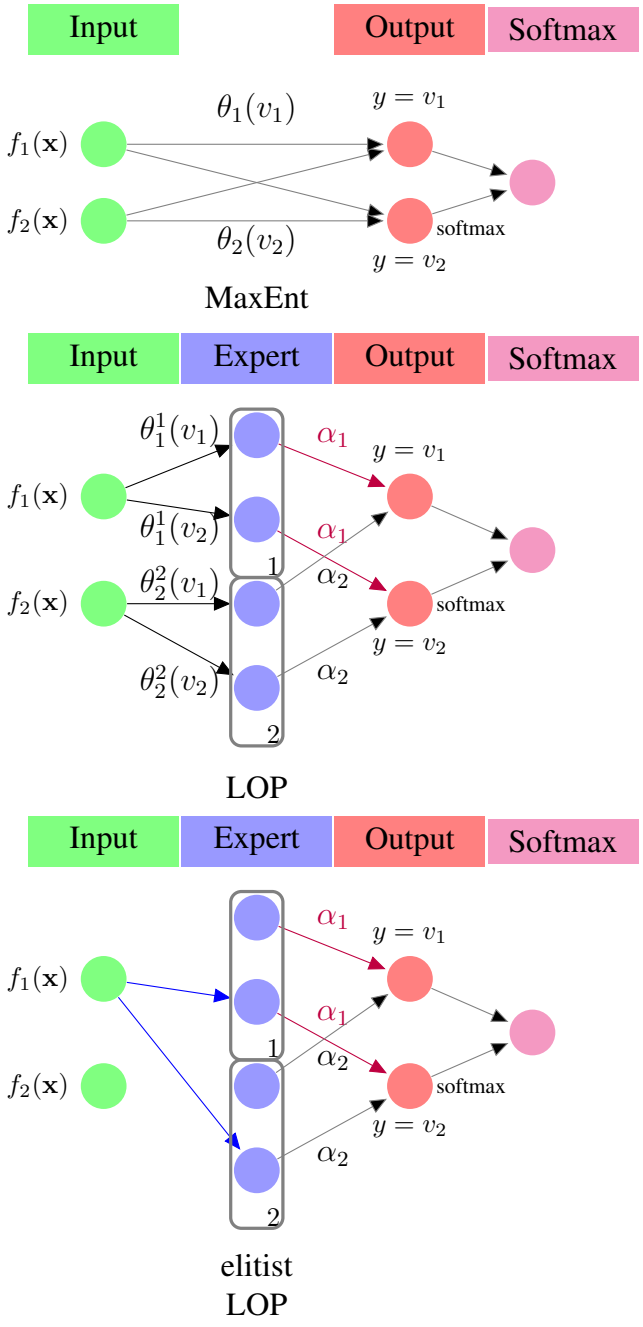


Figure 3.1: LOP and MaxEnt models drawn with neural network style network architectures.

regular MaxEnt model with 2 features and 2 output classes. The value at each node is computed as the sum of all incoming edge weights (parameters in the model) multiplied by the values of nodes on the originating end of edges. In this example, $d = 2$ and we have two input features ($f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$); there are two possible value assignments for y (v_1 and v_2, we use k to denote the index of output assignment). Correspondingly, there are four weight parameters ($\theta_j(v_k)$, for $j = 1, 2, k = 1, 2$) as shown by the directed edge going from "Input" layer nodes to "Output" layer nodes. We apply a softmax function to the "Output" layer to produce the probability assignment of each output class.

The middle figure shows a LOP with two experts. For each feature weight parameter θ, the superscript denotes which feature group it belongs to (e.g., θ^1 belongs to expert 1); the subscript denotes which input feature it belongs to (e.g., θ_1 means it applies to input feature $f_1(\boldsymbol{x})$); and the value in parentheses denotes which output class this feature is associated with (e.g., $\theta(v_1)$ applies to output class $y = v_1$). Features that belong to the same group are shown in the same color. α_1 and α_2 are the expert mixing coefficients to be learned in the LOP model. Two experts are shown in this diagram, each represented by a plate in the "Expert" layer. This example illustrates an expert generation scheme by partitioning the features into non-overlapping subsets – expert 1 has feature function $f_1(\boldsymbol{x})$ while expert 2 has feature function $f_2(\boldsymbol{x})$. The weights of the two experts are pre-trained and remain fixed. Training the LOP model involves only learning the α mixing parameters.

3.2.2 Automatic Induction of Elitist LOP

A major concern in adopting existing LOP methods is that there are no straightforward guidelines on how to create experts. Different feature partition schemes can result in dramatically different performance (Smith et al., 2005). A successfully designed LOP expert ensemble typically involves non-trivial engineering effort and a significant amount of domain expertise. Furthermore, the improvements shown in one application domain by a well-designed feature partition scheme do not necessarily carry over to other domains.

Therefore, it is our goal to investigate and search for effective methods to automatically induce LOP experts. We would like to find a set of experts that are diverse (i.e., having non-overlapping feature sets) and accurate. The main idea here is that we can train multiple

copies of MaxEnt models together to optimize training likelihood, but at the same time we encourage the MaxEnt models to differ as much as possible in their choice of features to employ. To meet this end, we use a $\ell_2\ell_1$ mixed norm to regularize a LOP, and denote the new model as *elitist LOP*.

Before we introduce the $\ell_2\ell_1$ norm, which has not been frequently used in NLP research, let us revisit a closely related but much better known sibling – the $\ell_1\ell_2$ norm used in group lasso.

For a set of model parameters $\boldsymbol{\theta}$, assume we have some feature grouping that partitions the features into G sub-groups. For simplicity, let us further assume that each feature group has the same number of features (M). We denote the index of each feature by g (for "group") and m (for "member"), and use $\boldsymbol{\theta}_{\hat{g}}$ to denote the feature group of index g.

The $\ell_1\ell_2$ mixed norm used in group lasso takes the following form:

$$\begin{aligned}\|\boldsymbol{\theta}\|_{1,2} &= \left(\sum_{g=1}^{G}\left(\sum_{m=1}^{M}|\theta_{g,m}|^2\right)^{1/2}\right) \\ &= \|(\|\boldsymbol{\theta}_{\hat{1}}\|_2, \cdots, \|\boldsymbol{\theta}_{\hat{G}}\|_2)\|_1\end{aligned}$$

This norm applies ℓ_2 regularization to each group of features, but enforces ℓ_1 regularization at the group level. Since the ℓ_1 norm encourages sparse solutions, therefore the effect of the mixed norm is to sparsify features at the group level by eliminating a whole group of features at a time.

It is easy to see how this relates to the $\ell_2\ell_1$ mixed norm, which is given as:

$$\begin{aligned}\|\boldsymbol{\theta}\|_{2,1} &= \left(\sum_{g=1}^{G}\left(\sum_{m=1}^{M}|\theta_{g,m}|\right)^{2}\right)^{1/2} \\ &= \|(\|\boldsymbol{\theta}_{\hat{1}}\|_1, \cdots, \|\boldsymbol{\theta}_{\hat{G}}\|_1)\|_2\end{aligned}$$

Like $\ell_1\ell_2$, this is also a group-sensitive sparsity-inducing norm, but instead of encouraging sparsity across feature groups, it promotes sparsity within each feature group.

We illustrate how we apply the $\ell_2\ell_1$ norm to induce diverse experts in the bottom diagram in Figure 3.1. Similar to the case of LOP, we create two "Expert" layer plates, one for

each expert (1 and 2). But unlike traditional LOP, where each expert only gets a subset of the input features (e.g., $f_1(\boldsymbol{x}) \mapsto 1$ and $f_2(\boldsymbol{x}) \mapsto 2$), each input feature is fully connected to each expert plate (i.e., there are four edges with the same expert superscript from the "Input" layer going into each plate in the "Expert" layer, shown in different colors).

We group every pair of feature weights that originate from the same input feature and end at the same output class into a separate feature group (e.g., $\theta_1^1(v_1)$ and $\theta_1^2(v_1)$ belong to one group, while $\theta_2^1(v_1)$ and $\theta_2^2(v_1)$ belong to another group, etc.). In total we have four feature groups, each shown in a different color.

When we apply the $\ell_2\ell_1$ norm to this feature grouping while learning the parameter weights in the LOP model, the regularization will push most of the features that belong to the same group towards 0. But since we are optimizing a likelihood objective, if a particular input feature is indeed predictive, the model will also try to assign some weights to the weight parameters associated with this feature. These two opposite forces create a novel competition mechanism among features that belong to the same group.

This competition mechanism encourages the experts to use different sets of non-overlapping features, thus achieving the diversity effect we want. But because we are also optimizing the likelihood of the ensemble, each expert is encouraged to use parameters that are as predictive as possible. Learning all the experts together can also be seen as a form of *feature sharing*, as suggested by Ando and Zhang (2005a) for multi-task learning.

In our automatic expert induction model with $\ell_2\ell_1$ norm, the overall objective is given as:

$$\mathcal{L} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left(-\log \left(\frac{\prod_{j=1}^{n} [P_j(y|\boldsymbol{x})]^{\alpha_j}}{\sum_{y'} \prod_{j=1}^{n} [P_j(y|\boldsymbol{x})]^{\alpha_j}} \right) + \lambda \|(\|\boldsymbol{\theta}_{\hat{1}}\|_1, \cdots, \|\boldsymbol{\theta}_{\hat{G}}\|_1)\|_2 \right)$$

where λ is a parameter that controls the regularization strength.

The feature groups are given as:

$$
\begin{aligned}
\boldsymbol{\theta}_{\hat{1}} &= \{\theta_1^1(v_1), \theta_1^2(v_1), \cdots, \theta_1^{e^n}(v_1)\} \\
\boldsymbol{\theta}_{\hat{2}} &= \{\theta_2^1(v_1), \theta_2^2(v_1), \cdots, \theta_2^{e^n}(v_1)\} \\
&\cdots \\
\boldsymbol{\theta}_{\hat{K}} &= \{\theta_K^1(v_1), \theta_K^2(v_1), \cdots, \theta_K^{e^n}(v_1)\} \\
\boldsymbol{\theta}_{\hat{K+1}} &= \{\theta_1^1(v_2), \theta_1^2(v_2), \cdots, \theta_1^{e^n}(v_2)\} \\
\boldsymbol{\theta}_{\hat{K+2}} &= \{\theta_2^1(v_2), \theta_2^2(v_2), \cdots, \theta_2^{e^n}(v_2)\} \\
&\cdots \\
\boldsymbol{\theta}_{\hat{G}} &= \{\theta_K^1(v_V), \theta_K^2(v_V), \cdots, \theta_K^{e^n}(v_V)\}
\end{aligned}
$$

There is a total of $G = K \times V$ feature groups, and each feature group has $M = n$ features.

The key difference here from LOP is that the $\boldsymbol{\theta}$ parameters are not pre-trained and fixed, but jointly learned with respect to $\ell_2\ell_1$ regularization. In the LOP case, learning the mixing coefficients α_i ($i = \{1, \ldots, n\}$) can be done by taking the gradients of α_i with respect to the training objective and directly plugging it into a gradient-based optimization framework, such as the *limited memory variable metric* (LMVM) used by Smith et al. (2005) or Stochastic Gradient Descent. However, learning the $\boldsymbol{\theta}$ parameters in our case is not as straightforward, since the objective is non-differentiable at many points due to the $\ell_2\ell_1$ norm. We cannot simply throw in an off-the-shelf gradient-based optimizer.

To alleviate the problem of non-differentiability, the recently proposed Forward-Backward Splitting (FOBOS) optimization framework (Duchi and Singer, 2009) comes to mind. FOBOS is a (sub-)gradient-based framework that solves the optimization problem iteratively in two steps: in each iteration, we first take a sub-gradient step, and then take an analytical minimization step that incorporates the regularization term, which can often be solved with a closed form solution. Formally, each iteration at timestamp t in FOBOS

consists of the following two steps:

$$\boldsymbol{\theta}_{t+\frac{1}{2}} = \boldsymbol{\theta}_t - \eta_t \boldsymbol{g}_t \tag{3.1}$$

$$\boldsymbol{\theta}_{t+1} = \operatorname*{argmin}_{\boldsymbol{\theta}} \left\{ \frac{1}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_{t+\frac{1}{2}}\|^2 + \eta_t \varphi(\boldsymbol{\theta}) \right\} \tag{3.2}$$

$\boldsymbol{g}_t$ is the subgradient of $-\log(P_{LOP}(y|\boldsymbol{x}))$ at θ_t. Step (3.1) is just a regular gradient-based step, where η_t is the step size and $\boldsymbol{g}_t$ is the sub-gradient vector of parameter vector $\boldsymbol{\theta}$. Step (3.2) finds a new vector that stays close to the interim vector after step (3.1), but also has a low penalty score according to the regularization term $\varphi(\boldsymbol{\theta})$ (in our case, $\varphi(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_{2,1}$). In the case of ℓ_1 and ℓ_2 regularization, each feature θ_i is independent in step (3.2), and thus can be solved separately.[2]

Duchi and Singer (2009) gave closed form solutions for solving the minimization problem in step (3.2) for ℓ_1, ℓ_2, and $\ell_1\ell_2$ norms, but no past research has demonstrated how to solve it for the $\ell_2\ell_1$ norm. We leverage the results given by Kowalski and Torrésani (2009) for elitist lasso, and show that parameter $\theta_{t+1,g,m}$ (the m^{th} feature of group g at timestamp $t+1$) can be solved analytically as follows:

$$\theta_{t+1,g,m} = \operatorname{sign}(\theta_{t+\frac{1}{2},g,m}) \left(|\theta_{t+\frac{1}{2},g,m}| - \tau_g \right)^+$$

$$\tau_g = \frac{\lambda'}{1 + \lambda' M_{\hat{g}}(\lambda')} \left\| \theta_{t+\frac{1}{2},g,1:M_{\hat{g}}(\lambda')} \right\|_1$$

$\operatorname{sign}(x)$ is the mathematical sign function. For $x \in \mathcal{R}$, we have $x^+ = x$ if $x \geq 0$ and $x^+ = 0$ if $x \leq 0$. $\lambda' = \eta_t \lambda$ is the regularization weight adjusted by the current step size.

Let $\bar{\boldsymbol{\theta}}_{\hat{g}}$ denote a new vector that holds the positive absolute value of $\boldsymbol{\theta}_{\hat{g}}$, sorted in descending order, i.e., $\bar{\theta}_{g,1} \geq \bar{\theta}_{g,2} \geq \cdots \bar{\theta}_{g,M}$. $M_{\hat{g}}(\lambda')$ is a positive integer given by the following definition:

$$\bar{\theta}_{g,M_{\hat{g}}(\lambda')+1} \leq \sum_{m=1}^{M_{\hat{g}}(\lambda')+1} \left(\bar{\theta}_{g,m} - \bar{\theta}_{g,M_{\hat{g}}(\lambda')+1} \right)$$

[2]This property turns out to be of great importance in real-world large scale scenarios, since it allows the optimization of high-dimensional feature vectors to be parallelized.

and

$$\bar{\theta}_{g,M_{\hat{g}}(\lambda')} > \lambda' \sum_{m=1}^{M_{\hat{g}}(\lambda')} \left(\bar{\theta}_{g,m} - \bar{\theta}_{g,M_{\hat{g}}(\lambda')}\right)$$

$\theta_{t+\frac{1}{2},g,1:M_{\hat{g}}(\lambda')}$ is then the subset of features from 1 to $M_{\hat{g}}(\lambda')$ – in descending order of their absolute value – in group g at timestamp $t + \frac{1}{2}$.

One problem with finding the exact solution of elitist lasso as illustrated above is that computing the value $M_{\hat{g}}(\lambda')$ involves sorting the parameter vector $\boldsymbol{\theta}$, which is a $O(n \log n)$ operation. When the number of features within each group is large, this could be prohibitively expensive. An approximate solution simply replaces $M_{\hat{g}}(\lambda')$ with the size of the group M. Kowalski and Torrésani (2009) found that approximated elitist lasso works nearly as well as the exact version, but is faster and easier to implement. We adopt this approximation in our experiments.

With this, we have described a general-purpose method for solving any convex optimization problem with $\ell_2\ell_1$ norm.

3.2.3 Results

We evaluate the performance of the proposed automatic expert induction method on NER over the same dataset as described in Section 2.5.1. The main difference here is that the baseline model is a MaxEnt classifier instead of a CRF.

Experimental settings specific to this section are explained below. We tuned the regularization parameters in the ℓ_1, ℓ_2 and $\ell_1\ell_2$ norms on the development dataset via grid search. We used a tuning procedure similar to the one used in Turian et al. (2010) where we evaluate results on the development set after each optimization iteration, and terminate the procedure after not observing a performance increase on the development set in 25 continuous iterations. We found 150 to be a good λ value for ℓ_2, 0.1 for ℓ_1 and 0.1 for $\ell_2\ell_1$. Similarly, we tuned the number of experts for both LOP baselines and the elitist LOP induction scheme. All model parameters were initialized to a random value in $[-0.1, 0.1]$.[3]

[3]Since the objective function in LOP is non-convex, random start typically works better than starting at 0 in practice.

			Dev Set			Test Set		
	Models	#E	Precision	Recall	F_1	Precision	Recall	F_1
CoNLL	MaxEnt-ℓ_2	1	88.46	87.73	88.09	81.42	81.64	81.53
	MaxEnt-ℓ_1	1	88.46	89.63	89.04	82.20	84.24	83.21†
	LOP random	2	88.75	90.79	89.76	82.73	85.84	84.25†‡
	LOP sequential	2	88.68	90.79	89.72	83.03	86.03	84.50†‡
	LOP-ℓ_1	5	88.76	90.41	89.58	82.90	85.94	84.40†‡
	elitist LOP	3	89.65	91.08	90.36	83.96	86.67	85.29†‡
	S05 simple	2	-	-	90.26	-	-	84.22
	S05 positional	3	-	-	90.35	-	-	84.71
	S05 label	5	-	-	89.30	-	-	83.27
MUC	MaxEnt-ℓ_2	1	91.47	86.20	88.76	89.06	80.44	84.53
	MaxEnt-ℓ_1	1	92.56	85.50	88.89	89.61	79.09	84.02
	LOP random	2	93.77	84.95	89.14	91.05	78.89	84.54
	LOP sequential	2	94.34	84.18	88.97	91.71	77.42	83.96
	LOP-ℓ_1	5	93.25	86.09	89.53	90.70	79.57	84.78‡
	elitist LOP	3	93.47	86.48	89.84	91.22	80.15	85.33†‡

Table 3.1: Results of NER on CoNLL-03 and MUC-6/7 datasets.

We compare our results against a MaxEnt baseline model with both ℓ_1 and ℓ_2 regularization, as well as three manually specified expert induction schemes. Two commonly used LOP expert induction schemes were also compared. In the first scheme (*LOP random*), experts are constructed by randomly partitioning the features into n subsets, where n is the number of experts. Each expert therefore has $\frac{1}{n}$th of the full feature set. The second scheme (*LOP sequential*) works in a similar way, but instead of random partition, we create feature subsets sequentially and preserve the relative order of the features. We also list results reported in Smith et al. (2005) on the same dataset for comparison. They experimented with three manually crafted expert induction schemes (*S05* {*simple;positional;label*}) for LOP with a Conditional Random Field (CRF), which is a more powerful sequence model than our MaxEnt baseline.

Results on the CoNLL-03 and MUC-6/7 dataset are shown in Table 3.1. † and ‡ indicate statistically significantly better F_1 scores than the MaxEnt-ℓ_2 and MaxEnt-ℓ_1 baselines, respectively.

The proposed automatic expert induction scheme (row *elitist LOP*) gives consistent and statistically significant improvements over the MaxEnt baselines on both the CoNLL-03

		IV			OOV		
	Models	Precision	Recall	F_1	Precision	Recall	F_1
CoNLL-03	MaxEnt-ℓ_2	89.21	86.81	88.00	85.69	80.61	83.08
	elitist LOP	90.25	89.85	90.05†	86.30	86.67	86.49†
MUC-6/7	MaxEnt-ℓ_2	90.21	80.88	85.29	84.90	78.80	81.74
	elitist LOP	92.27	80.02	85.71	87.49	80.62	83.91†

Table 3.2: OOV and IV results breakdown.

and MUC-6/7 test sets. In particular, on the CoNLL-03 test set, we observe an improvement of 3.7% absolute F_1 over *Maxent*-ℓ_2. The gains are particularly large in recall (by 5% absolute score), although there is also a 2.5% improvement in precision. The two conventional LOP induction schemes also show significant improvements over the MaxEnt baselines on this dataset, with the sequential feature partition scheme working slightly better than the random feature partition. However, elitist LOP outperforms the two LOP schemes by as much as 1% in absolute F_1, and also gives better results than manually crafted experts for CRFs in Smith et al. (2005).

On the MUC-6/7 test set, while elitist LOP still outperforms the MaxEnt baselines, the LOP schemes do not help or in some cases hurt performance. This suggests the lack of robustness of LOP which was discussed in Section 3.2.2. The automatically learned LOP experts are more robust on this data set, and gives a 0.8% improvement measured in absolute F_1 score over the *MaxEnt*-ℓ_2 baseline.

The elitist LOP model is more expressive than MaxEnt since it has more parameters to be combined. To understand whether the performance gain is obtained by the expressiveness or by the regularization of the $\ell_2\ell_1$ norm, we compare against an elitist LOP model with just ℓ_1 regularization (*LOP*-ℓ_1). Results show that the $\ell_2\ell_1$ norm is a better regularizer for avoiding overfitting with these models. We also see a significant gain from *LOP*-ℓ_1 over *MaxEnt*-ℓ_1, suggesting that the expressiveness of the model also helps.

To further understand *how* our method improves over the MaxEnt baseline, we break down the test results into two subsets: words that were seen in the training dataset (in-vocabulary, or IV), versus words that were not (out-of-vocabulary, or OOV). We removed the entity boundary tags (e.g. "I-ORG" becomes "ORG") so that each word token is evaluated separately. The division of IV and OOV subsets can be done by checking each word

against a lexicon compiled from training dataset.

If our automatic expert induction LOP method does successfully mitigate the weight undertraining problem, we would expect to see an improvement in OOV recognition. The result of this analysis is shown in Table 3.2. † indicates statistically significantly better F_1 scores than the MaxEnt-ℓ_2 baseline at 95% confidence level.

On the CoNLL-03 dataset, our method gives significant improvements over MaxEnt in both OOV and IV category. In particular, improvement from OOV subset (3.4% in F_1) is larger than the improvement from IV subset (2% in F_1). This matches our hypothesis that our method mitigates the weight undertraining problem and thus gives a stronger boost in OOV recognition.

This effect is even more pronounced on the MUC-6/7 dataset. The improvement in IV subset in this case is actually quite modest (0.5%), but we get a significant 2.2% improvement from the OOV subset.

3.2.4 Related Work

The use of group-sparsity norms in NLP research is relatively rare. Martins et al. (2011) proposed the use of structure-inducing norms, in particular $\ell_1\ell_2$ norm (group lasso), for learning the structure of classifiers. A key observation is that during testing, most of the runtime is consumed in the feature instantiation stage. Since the $\ell_1\ell_2$ norm discards whole groups of features at a time, there are fewer feature templates that need to be instantiated, therefore it could give a significant runtime speedup.

Our use of the $\ell_2\ell_1$ norm takes a different flavor in that we explore the internal structure of the model. There is, however, one recent paper that also makes use of the $\ell_2\ell_1$ norm for a similar reason. Das and Smith (2012) employed the $\ell_2\ell_1$ norm for learning sparse structure in a network formed by lexical predicates and semantic frames. Their results show that $\ell_2\ell_1$ norm yields much more compact models than the ℓ_1 or ℓ_2 norms, with superior performance in learning to expand lexicons.

However, the optimization problem in Das and Smith (2012) was much simpler since all parameters can only take on positive values, and therefore they could directly use a gradient-descent method with specialized edge condition checks. In our work, we have

shown a general-purpose method in the FOBOS framework for optimizing any convex function with $\ell_2\ell_1$ norm.

A related ℓ_1-ℓ_2 mixed norm is called the *elastic net* (Zou and Hastie, 2005). It was proposed to overcome a problem of the lasso (i.e., ℓ_1 regularization), which occurs when there is a group of correlated predictors, and the lasso tends to pick one and ignore all the others. Elastic net takes the form of a sum of a ℓ_1 and a ℓ_2 norm. It was used in Lavergne et al. (2010) for learning large-scale Conditional Random Fields.

3.2.5 Summary

Experimental results show that previous automatic expert definition methods used in LOP lack robustness across different tasks and data sets. Instead of manually defining good (i.e., diverse) experts, we demonstrated an effective way to induce experts automatically, by using a sparsity-inducing mixed $\ell_2\ell_1$ norm inspired by elitist lasso. We proposed a novel formulation of the optimization problem with the $\ell_2\ell_1$ norm in the FOBOS framework. Our method gives consistent and significant improvements over a MaxEnt baseline with fine-tuned ℓ_2 regularization on two NER datasets. The gains are most evident in recognizing entity types for out-of-vocabulary words.

3.3 Feature Noising for Structure Prediction

In the last section, we proposed a mixed $\ell_2\ell_1$ regularization for a product-of-experts model, and contrasted it with the traditional ℓ_2 and ℓ_1 norm to demonstrate its effectiveness. However, that specific form of mixed $\ell_2\ell_1$ regularization is specific to the product-of-experts model, and does not apply generally to all log-linear models.

In this section, we aim to provide an alternative approach that can be applied to all generalized linear models, including log-linear models for structure prediction, such as CRFs. Recall that traditional ℓ_2 or ℓ_1 regularization penalize all features in a uniform way without taking into account the properties of the actual model. An alternative approach to improving generalization while avoiding overfitting is to generate fake training data by adding random noise to the input features of the original training data. Intuitively, this can

be thought of as simulating missing features, whether due to typos or use of a previously unseen synonym. The effectiveness of this technique is well-known in machine learning (Abu-Mostafa, 1990; Burges and Schölkopf, 1997; Simard et al., 2000; Rifai et al., 2011a; van der Maaten et al., 2013), but working directly with many corrupted copies of a dataset can be computationally prohibitive.

In the recently proposed *dropout* learning scheme, each data sample is not just noised, but randomly "dropped out" during each iteration of parameter training (Hinton et al., 2012; Wang and Manning, 2013c). This effectively corresponds to an "expert" formation strategy by one-off feature bagging. It breaks co-adaptation in exactly the same way sought in the previous section (expert induction for product-of-expert ensemble), but now dynamically, by trying many experts rather than finding one good set of experts.

Fortunately, feature noising ideas often lead to tractable deterministic objectives that can be optimized directly. Sometimes, training with corrupted features reduces to a special form of regularization (Matsuoka, 1992; Bishop, 1995; Rifai et al., 2011b; Wager et al., 2013). For example, Bishop Bishop (1995) showed that training with features that have been corrupted with additive Gaussian noise is equivalent to a form of ℓ_2 regularization in the low noise limit. In other cases it is possible to develop a new objective function by marginalizing over the artificial noise (Wang and Manning, 2013c; van der Maaten et al., 2013).

We show how to efficiently simulate training with artificially noised features in the context of log-linear structured prediction, without actually having to generate noised data. We focus on dropout noise (Hinton et al., 2012), a recently popularized form of artificial feature noise where a random subset of features is omitted independently for each training example. Figure 3.2 illustrates dropout in two training iterations. Conceptually, given a training example, we sample some features to ignore (generate fake data) and make a parameter update. Our goal is to train with a roughly equivalent objective, without actually sampling.

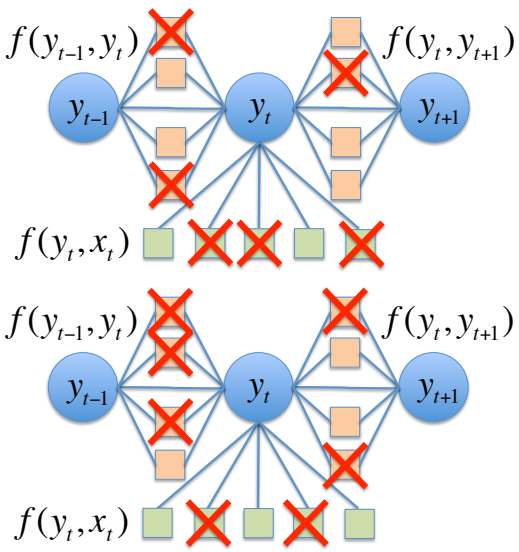


Figure 3.2: An illustration of dropout feature noising in linear-chain CRFs with only transition features and node features.

3.3.1 Feature Noising for MaxEnt

Recall the standard definition of a MaxEnt model that models the conditional probability of an output $\boldsymbol{y} \in \mathcal{Y}$ (e.g., a tag sequence) given some input $\boldsymbol{x} \in \mathcal{X}$ (e.g., a sentence):

$$P(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta}) = \exp\{\boldsymbol{f}(\boldsymbol{y},\boldsymbol{x}) \cdot \boldsymbol{\theta} - \log Z(\boldsymbol{x})\} \tag{3.3}$$

$$= \exp\{s_{\boldsymbol{y}} - A(\boldsymbol{s})\} \tag{3.4}$$

where we define $s_{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y},\boldsymbol{x}) \cdot \boldsymbol{\theta}$ and $A(\boldsymbol{s}) = \log Z(\boldsymbol{x})$; $\boldsymbol{s} = (s_1, \ldots, s_{|\mathcal{Y}|})$ is a vector of scores for each output sequence.

The key idea behind feature noising is to artificially corrupt the feature vector $\boldsymbol{f}(\boldsymbol{y},\boldsymbol{x})$ randomly into some $\tilde{\boldsymbol{f}}(\boldsymbol{y},\boldsymbol{x})$ and then to maximize the average log-likelihood of $\boldsymbol{y}$ given these corrupted features—the motivation is to choose predictors $\boldsymbol{\theta}$ that are robust to noise (missing words for example). Let $\tilde{\boldsymbol{s}}$, $\tilde{P}(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta})$ be a *randomly* perturbed version corresponding to $\tilde{\boldsymbol{f}}(\boldsymbol{y},\boldsymbol{x})$. We will also assume the feature noising preserves the mean: $\mathbb{E}[\tilde{\boldsymbol{f}}(\boldsymbol{y},\boldsymbol{x})] = \boldsymbol{f}(\boldsymbol{y},\boldsymbol{x})$, so that $\mathbb{E}[\tilde{\boldsymbol{s}}] = \boldsymbol{s}$.

It is useful to view feature noising as a form of regularization. Since feature noising

preserves the mean, the feature noising objective can be written as the sum of the original log-likelihood plus the difference in log-normalization constants:

$$\mathbb{E}[\log \tilde{P}(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta})] = \log P(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta}) - R(\boldsymbol{\theta},\boldsymbol{x}), \tag{3.5}$$

$$R(\boldsymbol{\theta},\boldsymbol{x}) \stackrel{\text{def}}{=} \mathbb{E}[A(\tilde{\boldsymbol{s}})] - A(\boldsymbol{s}). \tag{3.6}$$

Since $A(\cdot)$ is convex, $R(\boldsymbol{\theta},\boldsymbol{x})$ is always positive by Jensen's inequality and can therefore be interpreted as a regularizer. Note that $R(\boldsymbol{\theta},\boldsymbol{x})$ is in general non-convex.

Computing the regularizer (3.6) requires summing over all possible noised feature vectors, which can imply exponential effort in the number of features. This is intractable even for flat classification. Following Bishop Bishop (1995) and Wager et al. Wager et al. (2013), we approximate $R(\boldsymbol{\theta},\boldsymbol{x})$ using a second-order Taylor expansion, which will allow us to work with only means and covariances of the noised features. We take a quadratic approximation of the log-partition function $A(\cdot)$ of the noised score vector $\tilde{\boldsymbol{s}}$ around the the unnoised score vector $\boldsymbol{s}$:

$$A(\tilde{\boldsymbol{s}}) \approxeq A(\boldsymbol{s}) + \nabla A(\boldsymbol{s})^\top(\tilde{\boldsymbol{s}} - \boldsymbol{s}) + \frac{1}{2}(\tilde{\boldsymbol{s}} - \boldsymbol{s})^\top \nabla^2 A(\boldsymbol{s})(\tilde{\boldsymbol{s}} - \boldsymbol{s}).$$

Plugging (3.7) into (3.6), we obtain a new regularizer $R^{\text{q}}(\boldsymbol{\theta},\boldsymbol{x})$, which we will use as an approximation to $R(\boldsymbol{\theta},\boldsymbol{x})$:

$$R^{\text{q}}(\boldsymbol{\theta},\boldsymbol{x}) = \frac{1}{2}\mathbb{E}[(\tilde{\boldsymbol{s}} - \boldsymbol{s})^\top \nabla^2 A(\boldsymbol{s})(\tilde{\boldsymbol{s}} - \boldsymbol{s})] \tag{3.7}$$

$$= \frac{1}{2}\operatorname{tr}(\nabla^2 A(\boldsymbol{s}) \operatorname{Cov}(\tilde{\boldsymbol{s}})). \tag{3.8}$$

This expression still has two sources of potential intractability, a sum over an exponential number of noised score vectors $\tilde{\boldsymbol{s}}$ and a sum over the $|\mathcal{Y}|$ components of $\tilde{\boldsymbol{s}}$.

Multiclass classification If we assume that the components of $\tilde{\boldsymbol{s}}$ are independent, then $\operatorname{Cov}(\tilde{\boldsymbol{s}}) \in \mathbb{R}^{|\mathcal{Y}|\times|\mathcal{Y}|}$ is diagonal, and we have

$$R^{\text{q}}(\boldsymbol{\theta},\boldsymbol{x}) = \frac{1}{2}\sum_{\boldsymbol{y}\in\mathcal{Y}} \mu_{\boldsymbol{y}}(1-\mu_{\boldsymbol{y}}) \operatorname{Var}[\tilde{s}_{\boldsymbol{y}}] \tag{3.9}$$

where the mean $\mu_{\boldsymbol{y}} \stackrel{\text{def}}{=} p_\theta(\boldsymbol{y}|x)$ is the model probability, the variance $\mu_{\boldsymbol{y}}(1-\mu_{\boldsymbol{y}})$ measures model uncertainty, and

$$\text{Var}[\tilde{s}_{\boldsymbol{y}}] = \theta^\top \text{Cov}[\tilde{\boldsymbol{f}}(\boldsymbol{y}, \boldsymbol{x})]\boldsymbol{\theta} \tag{3.10}$$

measures the uncertainty caused by feature noising.[4] The regularizer $R^{\text{q}}(\boldsymbol{\theta}, \boldsymbol{x})$ involves the product of two variance terms, the first is non-convex in $\boldsymbol{\theta}$ and the second is quadratic in $\boldsymbol{\theta}$. Note that to reduce the regularization, we will favor models that (i) predict confidently and (ii) have stable scores in the presence of feature noise.

For multiclass classification, we can explicitly sum over each $\boldsymbol{y} \in \mathcal{Y}$ to compute the regularizer, but this will be intractable for structured prediction.

To specialize to multiclass classification for the moment, let us assume that we have a separate weight vector for each output $\boldsymbol{y}$ applied to the same feature vector $\boldsymbol{h}(\boldsymbol{x})$; that is, the score $s_{\boldsymbol{y}} = \boldsymbol{\theta}_{\boldsymbol{y}} \cdot \boldsymbol{h}(\boldsymbol{x})$. Further, assume that the components of the noised feature vector $\tilde{\boldsymbol{h}}(\boldsymbol{x})$ are independent. Then we can simplify (3.10) to the following:

$$\text{Var}[\tilde{s}_{\boldsymbol{y}}] = \sum_j \text{Var}[g_j(x)]\theta_{yj}^2. \tag{3.11}$$

Noising schemes We now give some examples of possible noise schemes for generating $\tilde{\boldsymbol{f}}(\boldsymbol{y}, \boldsymbol{x})$ given the original features $\boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x})$. This distribution affects the regularization through the variance term $\text{Var}[\tilde{s}_{\boldsymbol{y}}]$.

- *Additive Gaussian*:

  $\tilde{\boldsymbol{f}}(\boldsymbol{y}, \boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x}) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0,\, \sigma^2 I_{d\times d})$.

  In this case, the contribution to the regularizer from noising is $\text{Var}[\tilde{s}_{\boldsymbol{y}}] = \sum_j \sigma^2 \theta_{yj}^2$.

- *Dropout*:

  $\tilde{\boldsymbol{f}}(\boldsymbol{y}, \boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x}) \odot \boldsymbol{z}$, where $\odot$ takes the elementwise product of two vectors. Here, $\boldsymbol{z}$ is a vector with independent components which has $z_i = 0$ with probability δ, $z_i = \frac{1}{1-\delta}$ with probability $1-\delta$. In this case, $\text{Var}[\tilde{s}_{\boldsymbol{y}}] = \sum_j \frac{g_j(x)^2 \delta}{1-\delta}\theta_{yj}^2$.

[4] Here, we are using the fact that the first and second derivatives of the log-partition function yield the mean and variance.

- *Multiplicative Gaussian*:

  $\tilde{f}(\boldsymbol{y}, x) = f(\boldsymbol{y}, x) \odot (1 + \varepsilon)$, where $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_{d\times d})$. Here, $\mathrm{Var}[\tilde{s}_{\boldsymbol{y}}] = \sum_j g_j(x)^2 \sigma^2 \theta_{yj}^2$. Note that under our second-order approximation $R^{\mathrm{q}}(\boldsymbol{\theta}, \boldsymbol{x})$, the multiplicative Gaussian and dropout schemes are equivalent, but they differ under the original regularizer $R(\boldsymbol{\theta}, \boldsymbol{x})$.

3.3.2 Feature Noising for CRFs

So far, we have developed a regularizer that works for all log-linear models, but—in its current form—is only practical for multiclass classification. We now exploit the decomposable structure in CRFs to define a new noising scheme which does not require us to explicitly sum over all possible outputs $\boldsymbol{y} \in \mathcal{Y}$. The key idea will be to noise each local feature vector (which implicitly affects many $\boldsymbol{y}$) rather than noise each $\boldsymbol{y}$ independently.

Assume that the output $\boldsymbol{y} = (y_1, \ldots, y_T)$ is a sequence of T tags. In linear chain CRFs, the feature vector $\boldsymbol{f}$ decomposes into a sum of local feature vectors $\boldsymbol{h}_t$:

$$\boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x}) = \sum_{t=1}^{T} \boldsymbol{h}_t(y_{t-1}, y_t, \boldsymbol{x}), \tag{3.12}$$

where $\boldsymbol{h}_t(a, b, \boldsymbol{x})$ is defined on a pair of consecutive tags a, b for positions $t-1$ and t.

Rather than working with a score $s_{\boldsymbol{y}}$ for each $\boldsymbol{y} \in \mathcal{Y}$, we define a collection of *local scores* $\boldsymbol{s} = \{s_{a,b,t}\}$, for each tag pair (a, b) and position $t = 1, \ldots, T$. We consider noising schemes which independently set $\tilde{\boldsymbol{h}}_t(a, b, x)$ for each a, b, t. Let $\tilde{\boldsymbol{s}} = \{\tilde{s}_{a,b,t}\}$ be the corresponding collection of noised scores.

We can write the log-partition function of these local scores as follows:

$$A(\boldsymbol{s}) = \log \sum_{y\in\mathcal{Y}} \exp\left\{\sum_{t=1}^{T} s_{y_{t-1},y_t,t}\right\}. \tag{3.13}$$

The first derivative yields the edge marginals under the model $\mu_{a,b,t} = p_\theta(y_{t-1} = a, y_t = b|x)$, and the diagonal elements of the Hessian $\nabla^2 A(\boldsymbol{s})$ yield the marginal variances.

Now, following (3.8) and (3.9), we obtain the following regularizer:

$$R^{\mathrm{q}}(\boldsymbol{\theta}, \boldsymbol{x}) = \frac{1}{2} \sum_{a,b,t} \mu_{a,b,t}(1 - \mu_{a,b,t}) \operatorname{Var}[\tilde{s}_{a,b,t}], \tag{3.14}$$

where $\mu_{a,b,t}(1 - \mu_{a,b,t})$ measures model uncertainty about edge marginals, and $\operatorname{Var}[\tilde{s}_{a,b,t}]$ is simply the uncertainty due to noising. Again, minimizing the regularizer means making confident predictions and having stable scores under feature noise.

Computing partial derivatives So far, we have defined the regularizer $R^{\mathrm{q}}(\boldsymbol{\theta}, \boldsymbol{x})$ based on feature noising. In order to minimize $R^{\mathrm{q}}(\boldsymbol{\theta}, \boldsymbol{x})$, we need to take its derivative.

First, note that $\log \mu_{a,b,t}$ is the difference of a restricted log-partition function and the log-partition function. So again by properties of its first derivative, we have:

$$\begin{aligned} \nabla \log \mu_{a,b,t} = \mathbb{E}_{P_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}, y_{t-1}=a, y_t=b)}[\boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x})] \\ - \mathbb{E}_{P_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})}[\boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x})]. \end{aligned} \tag{3.15}$$

Using the fact that $\nabla \mu_{a,b,t} = \mu_{a,b,t} \nabla \log \mu_{a,b,t}$ and the fact that $\operatorname{Var}[\tilde{s}_{a,b,t}]$ is a quadratic function in $\boldsymbol{\theta}$, we can simply apply the product rule to derive the final gradient $\nabla R^{\mathrm{q}}(\boldsymbol{\theta}, \boldsymbol{x})$.

A Dynamic Program for the Conditional Expectation

A naive computation of the gradient $\nabla R^{\mathrm{q}}(\boldsymbol{\theta}, \boldsymbol{x})$ requires a full forward-backward pass to compute $\mathbb{E}_{P_{\boldsymbol{\theta}}(\boldsymbol{y}|y_{t-1}=a, y_t=b, \boldsymbol{x})}[\boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x})]$ for each tag pair (a, b) and position t, resulting in a $\mathcal{O}(K^4 T^2)$ time algorithm.

In this section, we reduce the running time to $\mathcal{O}(K^2 T)$ using a more intricate dynamic program. By the Markov property of the CRF, $\boldsymbol{y}_{1:t-2}$ only depends on (y_{t-1}, y_t) through y_{t-1} and $\boldsymbol{y}_{t+1:T}$ only depends on (y_{t-1}, y_t) through y_t.

First, it will be convenient to define the partial sum of the local feature vector from positions i to j as follows:

$$G_{i:j} = \sum_{t=i}^{j} \boldsymbol{h}_t(y_{t-1}, y_t, \boldsymbol{x}). \tag{3.16}$$

Consider the task of computing the feature expectation $\mathbb{E}_{P_{\boldsymbol{\theta}}(\boldsymbol{y}|y_{t-1}=a,y_t=b)}[\boldsymbol{f}(\boldsymbol{y},\boldsymbol{x})]$ for a fixed (a,b,t). We can expand this quantity into

$$\sum_{\boldsymbol{y}:y_{t-1}=a,y_t=b} P_{\boldsymbol{\theta}}(\boldsymbol{y}_{-(t-1:t)}|y_{t-1}=a,y_t=b)G_{1:T}.$$

Conditioning on y_{t-1}, y_t decomposes the sum into three pieces:

$$\sum_{\boldsymbol{y}:y_{t-1}=a,y_t=b} [\boldsymbol{h}_t(y_{t-1}=a,y_t=b,\boldsymbol{x}) + F_t^a + B_t^b],$$

where

$$F_t^a = \sum_{\boldsymbol{y}_{1:t-2}} P_{\boldsymbol{\theta}}(\boldsymbol{y}_{1:t-2}|y_{t-1}=a)G_{1:t-1}, \tag{3.17}$$

$$B_t^b = \sum_{\boldsymbol{y}_{t+1:T}} P_{\boldsymbol{\theta}}(\boldsymbol{y}_{t+1:T}|y_t=b)G_{t+1:T}, \tag{3.18}$$

are the expected feature vectors summed over the prefix and suffix of the tag sequence, respectively. Note that F_t^a and B_t^b are analogous to the forward and backward messages of standard CRF inference, with the exception that they are vectors rather than scalars.

We can compute these messages recursively in the standard way. The forward recurrence is

$$F_t^a = \sum_b p_\theta(y_{t-2}=b|y_{t-1}=a)\left[\boldsymbol{h}_t(y_{t-2}=b,y_{t-1}=a,\boldsymbol{x}) + F_{t-1}^b\right]$$

The backward recurrence is

$$B_t^b = \sum_a p_\theta(y_{t+1}=a|y_t=b)\left[\boldsymbol{h}_t(y_{t+1}=a,y_t=b,\boldsymbol{x}) + B_{t+1}^a\right]$$

Running the resulting dynamic program takes $\mathcal{O}(K^2Tq)$ time and requires $\mathcal{O}(KTq)$ storage, where K is the number of tags, T is the sequence length and q is the number of active features. Note that this is the same order of dependence as normal CRF training, but there is an additional dependence on the number of active features q, which makes training

	Dev			Test		
	P	R	F_1	P	R	F_1
None	89.84	88.97	89.40	84.28	85.06	84.67
ℓ_2	91.11	90.58	90.84	86.10	85.36	85.73
Dropout	93.85	89.96	91.86	88.40	86.45	87.42

Table 3.3: CoNLL-03 summary of results.

slower.

Fast Gradient Computations

In this section, we provide two ways to further improve the efficiency of the gradient calculation based on ignoring long-range interactions and based on exploiting feature sparsity.

Exploiting Feature Sparsity and Co-occurrence In each forward-backward pass over a training example, we need to compute the conditional expectations for all features active in that example. Naively applying the dynamic program in Section 3.3.2 is $\mathcal{O}(K^2T)$ for each active feature. The total complexity has to factor in the number of active features, q. Although q only scales linearly with sentence length, in practice this number could get large pretty quickly. For example, in the NER experiments (*cf.* Section 3.3.3), the average number of active features per token is about 20, which means $q \simeq 20T$; this term quickly dominates the computational costs. Fortunately, in sequence tagging and other NLP tasks, the majority of features are sparse and they often co-occur. That is, some of the active features would fire and only fire at the same locations in a given sequence. This happens when a particular token triggers multiple rare features.

We observe that all indicator features that only fired once at position t have the same conditional expectations (and model expectations). As a result, we can collapse such a group of features into a single feature as a preprocessing step to avoid computing identical expectations for each of the features. Doing so on the same NER tagging experiments cuts down q/T from 20 to less than 5, and gives us a 4 times speed up at no loss of accuracy. The exact same trick is applicable to the general CRF gradient computation as well and gives similar speedup.

3.3.3 Results

We evaluate the quadratic dropout regularizer in linear-chain CRFs on the CoNLL-03 NER dataset. The $F_{\beta=1}$ results are summarized in Table 3.3. NONE is the results of a CRF with no regularization; ℓ_2 is a CRF with ℓ_2 reguarlization; Dropout is the quadratic dropout regularization (3.14) described in this paper.

We observe a performance obtain of 1.7% and 1.0% absolute points in F_1 on the test and dev set, respectively.

3.4 Summary

In this chapter, we address a common problem in training log-linear models—feature weight undertraining. We first reviewed a strategy to counter this problem by generating a product-of-experts model where the input feature space are partitioned among experts in order to break feature co-adaptation. We then proposed a new method that automatically learns to induce a set of diverse experts, which is more robust across datasets and achieves higher accuracy than manually specified expert creation schemes. We then examine a new method based on "dropout" learning, which seeks to create many experts during different iterations of training, instead of just one set of experts. We present a new approach that formulates the dropout learning scheme as a form of feature noising regularization, based on second-order approximation. This new regularization technique favors models that predict confidently and are robust to noise in the data. We further extend this method to the case of structured prediction in CRFs. We tackle the key challenge of dealing with feature correlations that arise in the structured prediction setting in several ways. Finally, we applied our methods to the CoNLL-03 NER dataset and demonstrate significant gains over standard ℓ_2 regularization for CRF.

Chapter 4

Investigating Non-linear Architectures

4.1 Introduction

In the last chapter we considered methods for balancing the problem of overfitting and underfitting in training a log-linear model. In this chapter, we turn our attention to the connection between the log-linear model architecture and feature representation. Most methods developed so far for sequence labeling employ generalized linear statistical models, meaning methods that describe the data as a combination of linear basis functions, either directly in the input variables space (e.g., SVM) or through some transformation of the probability distributions (e.g., "log-linear" models). More broadly speaking, one rarely finds examples of non-linear models in the field of NLP as a whole, especially if compared to neighboring fields like Computer Vision and Audio Signal Processing. Recently, Collobert et al. (2011) proposed "deep architecture" models for sequence labeling (named Sentence-level Likelihood Neural Nets, abbreviated as SLNN henceforth), and showed promising results on a range of tasks (POS tagging, NER, Chunking, and SRL). Two new changes were suggested: extending the model from a linear to a non-linear architecture; and replacing discrete feature representations with distributional feature representations in a continuous space. It has generally been argued that non-linearity between layers is vital to the power of neural models (Bengio, 2009). The relative contribution of these changes, however, is unclear, as is the question of whether gains can be made by introducing non-linearity to conventional feature-based models.

In this chapter, we first illustrate the close relationship between CRF and SLNN models, and conduct an empirical investigation of the effect of non-linearity with different feature representations. Experiments on Named Entity Recognition (NER) suggest that non-linear models are highly effective in low-dimensional distributed feature space, but offer no benefits in high-dimensional discrete space. Furthermore, both linear and non-linear models improve when we combine the discrete and continuous feature spaces, but a linear model still outperforms the non-linear one.

4.2 Related Work

Most discussions of non-linearity in NLP come up in the consideration of different choices of statistical classifiers. It is well known that linear classifiers such as the Perceptron and Support Vector Machines (SVM) are not capable of approximating a non-linear function such as the *XOR* function. There exist methods that enhance the aforementioned linear classifiers with non-linear prediction power, such as the extension of the Perceptron to the Multilayer Perceptron (Rumelhart and McClelland, 1986), and the use of Kernel Methods in SVMs (Schölkopf et al., 1997).

Other classic non-linear models include *Nearest Neighbor*, *Guassian Kernels* and *Radial Basis Function Networks*. These models are often termed *local* non-linear models, whereas *global* non-linear models commonly refer to neural networks with hidden layers. Invented in the early 60's and popular in the 80's, neural network models faded from the public view of mainstream NLP research, but re-emerged recently. There has been a wave of new and exciting results coming out of a line of research now referred to as "deep learning". Upon first glance, deep learning resembles the same class of multi-layer (hence the name "deep") neural nets; but the term actually encapsulates a wider range of new advances in neural network research, especially in representation learning.

The recent surge of interest and activity in deep learning research can be traced back to Restricted Boltzmann Machines (RBM), and Stacked RBMs (also known as Deep Belief Networks) (Hinton and Salakhutdinov, 2006), which can be thought of as the undirected counterpart of directed graphical models like the Multi-layer Perceptron. Perhaps more important is the discovery of new training procedures to efficiently learn these complex

models. For example, the layer-wise unsupervised pre-training scheme proposed by Hinton et al. (2006), Autoencoders (Bengio et al., 2006; LeCun and Bengio, 2007), and Contrastive Estimation methods (Collobert et al., 2011; Rifai et al., 2012). Deep architecture models trained in an unsupervised fashion using these techniques are able to learn multi-level distributional representations for a variety of datasets. The best results from these models match or beat the state-of-the-art in Computer Vision (Lee et al., 2009; Le et al., 2012b), Speech Processing (Mikolov et al., 2011; Dahl et al., 2012), and NLP (Socher et al., 2011a,b,c; Collobert, 2011; Le et al., 2012a).

In sequence labeling, Collobert and Weston (2008) demonstrated that a single-layer neural network, using continuous distributional features pre-trained on a large amount of text, rivals the state-of-the-art in a series of tasks. Collobert et al. (2011) discussed the importance of "word embedding" feature vectors pre-trained in an unsupervised fashion, and found that the performance of the model drops sharply if the word embedding was directly learned from supervised data. The same effect was also studied in Erhan et al. (2010). Their findings suggest that unsupervised pre-training provides better initialization points, and guides the learning algorithm towards local minima with better generalization. Empirical results from Turian et al. (2010) also support the argument that distributional representations generalize better than discrete representations over out-of-domain or unseen data.

However, the question remains whether it is necessary to employ a non-linear architecture in order to profit from a pre-trained distributional feature representation. How would a non-linear architectures help with a traditional discrete feature space? Our experimental results show that non-linear deep architecture does not necessarily improve performance on sequence labeling tasks in high-dimensional discrete space, but gains are realized through the use of distributional representations in continuous space. Next, we will introduce the models we discussed here in Section 4.3, and test our hypotheses in Section 4.5.

4.3 From CRFs To SLNNs

Recall the CRF model definition in Eq. 2.3 of Section 2.3. Let us first focus our discussion on the node clique potentials Ψ for now and temporarily shelve the edge potentials Φ, which

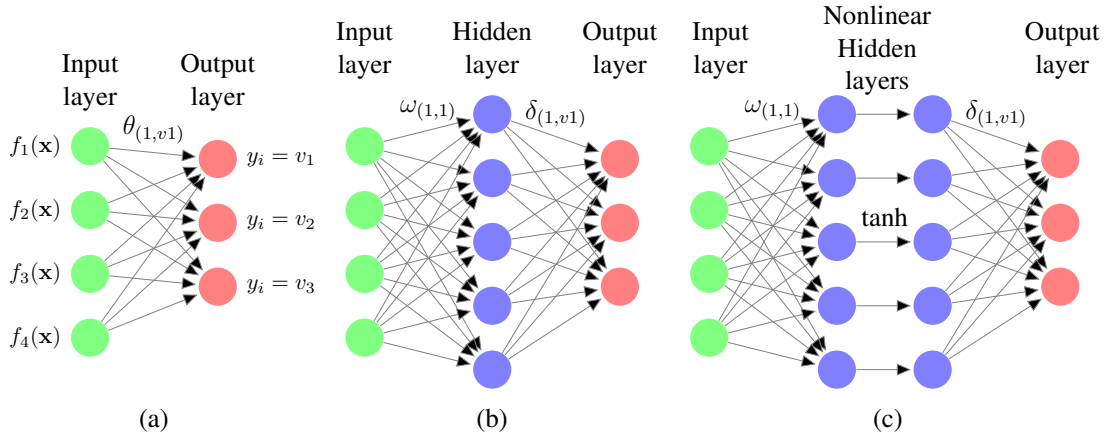


Figure 4.1: In this diagram, we assume the random variable y_i has three possible value assignments (v_1, v_2, v_3). On the left side (a) is the linear potential function ψ in CRF, illustrated as a single-layer Input-Output Neural Network. In the middle (b) is a potential function as a two-layer Linear Neural Network; on the right side (c) is a two-layer Deep Neural Network.

we will return to later. We call the operand of the exponentiation operator in Ψ a *potential function* ψ. In a CRF, this can be expressed in matrix notation as:

$$\psi(\boldsymbol{x}, y_i; \boldsymbol{\theta}) = |\boldsymbol{\theta}^{\mathsf{T}} \mathbf{g}(\boldsymbol{x})|_{\widehat{y}_i 1}$$

We use the notation $\widehat{y}_i$ to denote the ordinal index of the value assigned to y_i. This linear potential function ψ can be visualized using a neural network diagram, shown in the left plot in Figure 4.1. In this diagram, we assume the random variable y_i has three possible value assignments (v_1, v_2, v_3). On the left side (a) is the linear potential function ψ in CRF, illustrated as a single-layer Input-Output Neural Network. In the middle (b) is a potential function as a two-layer Linear Neural Network; on the right side (c) is a two-layer Deep Neural Network. Each edge in the graph represents a parameter weight $\theta_{(k,\widehat{y}_i)}$, for feature $f_k(\boldsymbol{x})$ and a variable assignment of y_i. In neural network terminology, this architecture of (b) is called a single-layer Input-Output Neural Network (IONN). [1] Normalizing locally in a logistic regression is equivalent to adding a *softmax* layer to the output layer of the IONN,

[1] The bias parameter "*b*" commonly seen in Neural Network convention can be encoded as an "always on" feature in the input layer.

which was commonly done in neural networks, such as in Collobert et al. (2011).

We can add a hidden linear layer to this architecture to formulate a two-layer Linear Neural Network (LNN), as shown in the middle diagram of Figure 4.1. The value of the node z_j in the hidden layer is computed as $z_j = \sum_k \omega_{(k,j)} f_k(\boldsymbol{x})$. The value y_i for nodes in the output layer is computed as: $y_i = \sum_j \delta_{(j,i)} z_j = \sum_j \delta_{(j,i)} \sum_k \omega_{(k,j)} f_k(\boldsymbol{x})$. where $\omega_{(k,j)}$ and $\delta_{(j,i)}$ are new parameters introduced in the model. In matrix form, it can be written as $\boldsymbol{y} = \Delta^\intercal \mathbf{z} = \Delta^\intercal \Omega^\intercal \mathbf{g}(\boldsymbol{x})$. The node potential function now becomes:

$$\psi'(\boldsymbol{x}, y_i; \Omega, \Delta) = |\Delta^\intercal \Omega^\intercal \mathbf{g}(\boldsymbol{x})|_{\widehat{y}_i 1}$$

This two-layer network is actually no more powerful than the previous model, since we can always compile it down to a single-layer IONN by making $\Theta = \Omega\Delta$. In the next step, we take the output of the hidden layer in the LNN, and send it through a non-linear activation function, such as a *sigmoid* or *tanh*, then we arrive at a two-layer Neural Network (TLNN) model. Unlike the previous two models, the TLNN is non-linear, and thus capable of representing a more complex decision surface.

So far we have extended the potential function used in node cliques of a CRF to a non-linear TLNN. And if we keep the potential function for edge cliques the same as before, then in fact we have arrived at an identical model to the SLNN in Collobert et al. (Collobert et al., 2011). The difference between a SLNN and an ordinary TLNN model is that we need to take into consideration the influence of edge cliques, and therefore we can no longer normalize the clique factors at each position to calculate the local marginals, as we would do in a logistic regression. The cardinality of the output variable vector $\boldsymbol{y}$ grows exponentially with respect to input sequence length. Fortunately, we can use *forward-backward* style dynamic programming to compute the marginal probabilities efficiently.

It is also worth pointing out that this model has in fact been introduced a few times in prior literature. It was termed *Conditional Neural Fields* by Peng et al. (2009), and later *Neural Conditional Random Fields* by Do and Artieres (2010). Unfortunately, the connection to Collobert and Weston (2008) was not recognized in either of these two studies; vice versa, neither of the above were referenced in Collobert et al. (2011). This model also appeared previously in the speech recognition literature in Prabhavalkar and Fosler-Lussier

(2010).

4.4 Parameter Learning

Supervised conditional training of the SLNN model amounts to maximizing the log-likelihood objective function $\mathcal{L}$, given by the sum of log-probabilities over training examples, similar to a CRF:

$$\mathcal{L}(\boldsymbol{y}^*|\boldsymbol{x}) = \sum_{l=1}^{|\boldsymbol{x}|} \Big(\sum_{i=1}^{|\boldsymbol{x}^l|} \psi'(\boldsymbol{x}^l, \boldsymbol{y}^l{}_i^*) + \sum_{j=1}^{|\boldsymbol{x}^l|} \phi(\boldsymbol{x}^l, \boldsymbol{y}^l{}_j^*, \boldsymbol{y}^l{}_{j-1}^*) \Big) - \sum_{l=1}^{|\boldsymbol{x}|} \log Z(\boldsymbol{x})$$

The change in node potential function from ψ to ψ' does not affect the inference procedure, and thus we can employ the same dynamic programming algorithm as in a CRF to calculate the log sum over $Z(\boldsymbol{x})$ and the expectation of feature parameters.

We adopted the simple L-BFGS algorithm for training weights in this model (Liu and Nocedal, 1989). Although L-BFGS is in general slower than mini-batch SGD – another common optimization algorithm used to train neural networks (Bengio et al., 2006, *inter alia*), it has been found to be quite stable and suitable for learning neural networks (Socher et al., 2011c). The gradient of a parameter $\omega_{(k,j)}$ is calculated as the following:

$$\frac{\partial \mathcal{L}}{\partial \omega_{(k,j)}} = \sum_{l=1}^{|\boldsymbol{x}|} \sum_{i=1}^{|\boldsymbol{x}_l|} \left(\frac{\partial \psi'(\boldsymbol{x}^l, \boldsymbol{y}^l{}_i)}{\partial \omega_{(k,j)}} - \mathbb{E}_{P(\boldsymbol{y}^l|\boldsymbol{x}^l)} \left[\frac{\partial \psi'(\boldsymbol{x}^l, \boldsymbol{y}^l{}_i)}{\partial \omega_{(k,j)}} \right] \right)$$

The partial derivative of the potential function $\frac{\partial \psi'(\boldsymbol{x}^l, \boldsymbol{y}^l{}_i)}{\partial \omega_{(k,j)}}$ can be calculated using the back-propagation procedure, identical to how gradients of a standard Multilayer Perceptron are calculated. The gradient calculation for output layer parameters Δ and edge parameters Λ follow the same form. We apply ℓ_2-regularization to prevent overfitting.

4.5 Experiments and Results

We evaluate the CRF and SLNN models on the Named Entity Recognition (NER) task.

In all experiments, we used the development portion of the CoNLL-03 data (*cf.* Section 2.5.1) to tune the ℓ_2-regularization parameter σ (variance in Gaussian prior), and found 20 to be a stable value. Overall tuning σ does not affect the qualitative results in our experiments. We terminate L-BFGS training when the average improvement is less than 1e-3. All model parameters are initialized to a random value in $[-0.1, 0.1]$ in order to break symmetry. We did not explicitly tune the features used in the CRF to optimize for performance, since feature engineering is not the focus of this study. However, overall we found that the feature set we used is competitive with CRF results from earlier literature (Turian et al., 2010; Collobert et al., 2011). For models that embed hidden layers, we set the number of hidden nodes to 300. [2] Results are reported on the standard evaluation metrics of entity/chunk precision, recall and F_1 measure.

For experiments with continuous space feature representations (a.k.a., word embedding), we took the word embedding (130K words, 50 dimensions) used in Collobert et al. (2011), which were trained for 2 months over Wikipedia text. [3] All sequences of numbers are replaced with `num` (e.g., "PS1" would become "PS`num`"), sentence boundaries are padded with token `PAD`, and unknown words are grouped into `UNKNOWN`. This is a different set of word embedding from the one reported in Turian et al. (2010). We attempt to replicate the model described in Collobert et al. (2011) without task-specific fine-tuning, with a few exceptions: 1) we used the *soft tanh* activation function instead of *hard tanh*; 2) we use the `BIO2` tagging scheme instead of `BIOES`; 3) we use L-BFGS optimization algorithm instead of stochastic gradient descent; 4) we did not use Gazetteer features; 5) Collobert et al. (2011) mentioned 5 binary features that look at the capitalization pattern of words to append to the embedding as additional dimensions, but only 4 were described in the paper, which we implemented accordingly.

For both the CRF and SLNN models, we experiment with both the discrete binary valued feature representation used in a regular CRF, and the word embedding described.

[2] We tried varying the number of hidden units in the range from 50 to 500, and the main qualitative results remain the same.

[3] Available at `http://ml.nec-labs.com/senna/`.

	CRF			SLNN		
	P	R	F_1	P	R	F_1
$CoNLL_d$	91.1	90.6	**90.8**	89.3	89.7	89.5
$CoNLL_t$	86.1	85.4	**85.7**	83.3	83.9	83.6
ACE	81.0	74.2	**77.4**	80.9	74.0	77.3
MUC	72.5	74.5	**73.5**	71.1	74.1	72.6

Table 4.1: Results of CRF versus SLNN, over discrete feature space. $CoNLL_d$ stands for the CoNLL development set, and $CoNLL_t$ is the test set. Best F_1 score on each dataset is highlighted in bold.

Unless otherwise stated, the set of edge features is limited to pairs of predicted labels at the current and previous positions, i.e., (y_i, y_{i-1}). The same edge features were used in Collobert et al. (2011) and were called "transition scores" ($[A]_{i,j}$).

4.5.1 Results of Discrete Representation

The first question we address is the following: in the high-dimensional discrete feature space, would the non-linear architecture in SLNN model help it to outperform CRF?

Results from Table 4.1 suggest that SLNN does not seem to benefit from the non-linear architecture. In particular, on the CoNLL and MUC dataset, SLNN resulted in a 1% performance drop, which is significant for NER. The specific statistical properties of this dataset that lead to the performance drop are hard to determine, but we believe it is partially because the SLNN has a much harder non-convex optimization problem to solve – on this small dataset, the SLNN with 300 hidden units generates a shocking number of 100 million parameters (437905 features times 300 hidden dimensions), due to the high dimensionality of the input feature space.

To further illustrate this point, we also compared the CRF model with its Linear Neural Network (LNN) extension, which has exactly the same number of parameters as the SLNN but does not include the non-linear activation layer. Although this model is identical in representational power to the CRF as we discussed in Section 4.3, the optimization problem here is no longer convex (Ando and Zhang, 2005a). To see why, consider applying a linear scaling transformation to the input layer parameter matrix Ω, and apply the inverse scaling to output layer Δ matrix. The resulting model has exactly the same function values. We

	CRF			LLN		
	P	R	F_1	P	R	F_1
$CoNLL_d$	91.1	90.6	**90.8**	89.5	90.6	90.0
$CoNLL_t$	86.1	85.4	**85.7**	83.1	84.7	83.9
ACE	81.0	74.2	**77.4**	80.7	74.3	77.3
MUC	72.5	74.5	73.5	72.3	75.2	**73.7**

Table 4.2: Results of CRF versus LNN, over discrete feature space.

can see from Table 4.2 that there is indeed a performance drop with the LNN model as well, likely due to difficulty with optimization. By comparing the results of LNN and SLNN, we see that the addition of a non-linear activation layer in SLNN does not seem to help, but in fact further decreases performance in all cases.

A distinct characteristic of NLP data is its high dimensionality. The vocabulary size of a decent sized text corpus is already in the tens of thousands, and bigram statistics are usually an order of magnitude larger. These basic information units are typically very informative, and there is not much structure in them to be explored. Although some studies argue that non-linear neural nets suffer less from the curse of dimensionality (Attali and Pagés, 1997; Bengio and Bengio, 2000; Pitkow, 2012), counter arguments have been offered (Camastra, 2003; Verleysen et al., 2003). The empirical results from our experiment seems to support the latter. Similar results have also been found in other NLP applications such as Text Classification. Joachims concluded in his seminal work: "non-linear SVMs do not provide any advantage for text classification using the standard kernels" (Joachims, 2004, p. 115). If we compare the learning curve of CRF and SLNN (Figure 4.2), where we vary the amount of binary features available in the model by random sub-sampling, we can further observe that SLNNs enjoy a small performance advantage in lower dimensional space (when less than 30% of features are used), but are quickly outpaced by CRFs in higher dimensional space as more features become available.

Another point of consideration is whether there is actually much non-linearity to be captured in sequence labeling. While in some NLP applications like grammar induction and semantic parsing, the data is complex and rich in statistical structures, the structure of data in sequence labeling is considerably simpler. This contrast is more salient if we compare with data in Computer Vision tasks such as object recognition and image segmentation.

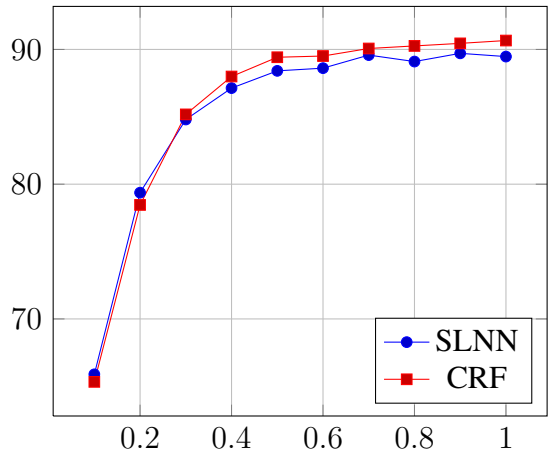


Figure 4.2: The learning curve of SLNN vs. CRF on CoNLL-03 dev set, with respect to the percentage of discrete features used (i.e., size of input dimension). Y-axis is the F_1 score (out of 100), and X-axis is the percentage of features used.

The interactions among local variables there are much stronger and more likely to be non-linear. Lastly, models like a CRF actually already capture some of the non-linearity in the input space through the interactions of latent variables (Liang et al., 2008), and it is unclear how much additional gain we would get by explicitly modeling the non-linearity in local inputs.

4.5.2 Results of Distributional Representation

For the next experiment, we replace the discrete input features with a continuous space representation by looking up the embedding of each word, and concatenate the embedding of a five word window centered around the current position. Four binary features are also appended to each word embedding to capture capitalization patterns, as described in Collobert et al. (2011). Results of the CRF and SLNN under this setting for the NER task is show in Table 4.3.

With a continuous space representation, the SLNN model works significantly better than a CRF, by as much as 7% on the CoNLL development set, and 3.7% on ACE dataset. This suggests that there exist statistical dependencies within this low-dimensional (300) data that cannot be effectively captured by linear transformations, but can be modeled in the

	CRF			SLNN		
	P	R	F_1	P	R	F_1
$CoNLL_d$	80.7	78.7	79.7	86.1	87.1	**86.6**
$CoNLL_t$	76.4	75.5	76.0	79.8	81.7	**80.7**
ACE	71.5	71.1	71.3	75.8	74.1	**75.0**
MUC	65.3	74.0	69.4	65.7	76.8	**70.8**

Table 4.3: Results of CRF versus SLNN, over continuous space feature representations.

	$CoNLL_d$	$CoNLL_t$	ACE	MUC
$CRF_{discrete}$	90.8	85.7	77.4	73.5
CRF_{join}	92.4	87.7	82.2	81.1
$SLNN_{continuous}$	86.6	80.7	75.0	70.8
$SLNN_{join}$	91.9	87.1	81.2	79.7

Table 4.4: Results of CRF and SLNN when word embedding is appended to the discrete features. Numbers shown are F_1 scores.

non-linear neural nets. This perhaps coincides with the large performance improvements observed from neural nets in handwritten digit recognition datasets as well (Peng et al., 2009; Do and Artieres, 2010), where dimensionality is also relatively low.

4.5.3 Combine Discrete and Distributional Features

When we join word embedding with discrete features, we see further performance improvements, especially in the out-of-domain datasets. The results are shown in Table 4.4.

A similar effect was also observed in Turian et al. (2010). The performance of both the CRF and SLNN increases by similar relative amounts, but the CRF model maintains a lead in overall absolute performance.

It is also worth pointing out that the word embedding used here is not crafted from any specific NER dataset, but built on top of the generic Wikipedia and Reuters data. It is quite impressive how the generic embedding matches the performance of a CRF with tens of thousands of discrete features specifically extracted from in-domain data, especially in the out-of-domain evaluation.

4.5.4 Influence of Edge Cliques

		CRF		SLNN	
		none	edge	none	edge
CoNLL_d	discrete	89.1	90.8	87.9	89.5
	continuous	72.3	79.7	78.7	86.6
CoNLL_t	discrete	82.2	85.7	81.5	83.6
	continuous	68.6	76.0	72.6	80.7
ACE	discrete	76.7	77.4	75.8	77.3
	continuous	67.4	71.3	71.6	75.0
MUC	discrete	72.8	73.5	72.2	72.6
	continuous	62.6	69.4	65.8	70.8

Table 4.5: With or without edge cliques features (none, edge) for CRF and SLNN. F_1 scores are shown for both discrete and continuous feature representation.

To understand the influence of edge cliques, we also compared results with edge features removed, which makes the CRF and SLNN equivalent to a logistic regression and a regular Multilayer Perceptron, respectively. Results of the CRF and SLNN models with and without edge features are shown in Table 4.5.

We can see that removing edge features in the discrete feature case resulted in a significant decrease of about 2% in performance in both CRF and SLNN. This degradation is even more severe in the continuous feature case, where a 5-7% decrease was observed. One explanation is related to the insight drawn by Liang et al. (2008) that CRFs exhibit non-linearity in terms of their marginal predictions, through the propagation of information via nearby hidden variables. In this way the use of edge cliques captures more non-linear effects in low-dimensional continuous feature space.

4.6 Summary

We carefully compared and analyzed the non-linear neural networks used in Collobert et al. (2011) and the widely adopted CRF, and revealed their close relationship. Through extensive experiments on NER and Syntactic Chunking, we have shown that non-linear architectures are effective in low dimensional continuous input spaces, but that they are not better

suited for conventional high-dimensional discrete input spaces. Furthermore, both linear and non-linear models benefit greatly from the combination of continuous and discrete features, especially for out-of-domain datasets. This finding confirms earlier results that distributional representations can be used to achieve better generalization. Also, the findings of this chapter helps us to choose to continue using discrete feature-based log-linear models in the bilingual NER experiments in the upcoming chapters.

Chapter 5

Semi-supervised Learning with Bitext

5.1 Introduction

In the previous two chapters, we examined new methods for better regularizing weight training of log-linear models, as well as the implications of the linearity in model architecture under various feature representation schemes. These techniques were presented in the context of improving model training and model properties under a *fully supervised* setting. At a high level, if we view the NER model as a black-box system, and conceptualize the supervised training data as input signals, the previous two chapters can be thought of as improvements from within the black-box. Another obvious direction to improve the output quality of this whole system is to increase the amount of input training signals. It is well-known that the performance of supervised learners increases when more labeled training examples become available.

The problem is that in most application scenarios, including NER, manually labeled data are extremely limited in quantity and costly to produce. On the other hand, we live in an age of abundance of unannotated data—as regards NLP, there has been an explosion in the amount of freely available web and news texts. Clearly unannotated data does not provide as direct a training signal as the labeled data, but they still contain valuable information that could help guide the learning of the system. A straight-forward example of such useful information is word context or word co-occurrence statistics. To illustrate, imagine a simple setting where from the annotated training data, we have observed word

context C occurring quite a few times, and each time the word immediately following C was labeled as a PERSON type entity. Our model may infer that C is a strong predictor for PERSON entities; now suppose at test time, we are predicting the entity type of a phrase A, which has never appeared in our training data, and is not proceeded by C: We may not have a strong clue as to what type is A; but we do have access to a large corpus of unannotated data, and let us assume C and A do appear together very often in this collection of data, even though A is never explicitly marked as a PERSON entity. By leveraging our knowledge about context C, we can inform the model about the high likelihood of phrase A being a PERSON entity, even if we have never directly observed A in supervised training.

A number of semi-supervised techniques have been introduced to tackle this problem, such as bootstrapping (Yarowsky, 1995; Collins and Singer, 1999; Riloff and Jones, 1999), multi-view learning (Blum and Mitchell, 1998; Ganchev et al., 2008) and structural learning (Ando and Zhang, 2005b).

However, most previous semi-supervised work is situated in a monolingual setting where all unannotated data are available only in a single language. In this chapter, we explore another source of unlabeled data for training, but instead of exploiting regularities in a single language, we study the use of bilingual translated parallel text, or *bitext* for short.

In recent years, a vast amount of bitext has been generated in our increasingly connected multilingual world. Examples include parliamentary proceedings in multilingual nations or regions such as Canada and Hong Kong, legal proceedings from international corporate lawsuits involving companies from different countries, translated bilingual news, and many others.

While such bitexts have primarily been leveraged to train statistical machine translation (SMT) systems, contemporary research has increasingly considered the possibilities of utilizing parallel corpora to improve systems outside of SMT. For example, Yarowsky and Ngai (2001) projects the part-of-speech labels assigned by a supervised model in one language (e.g. English) onto word-aligned parallel text in another language (e.g. Chinese) where less manually annotated data is available. Similar ideas were also employed by Das and Petrov (2011) and Fu et al. (2011). As a result, we can find complementary cues in the two languages that help to disambiguate named entity mentions (Brown et al., 1991). For example, the English word “Jordan” can be either a last name or a country. Without

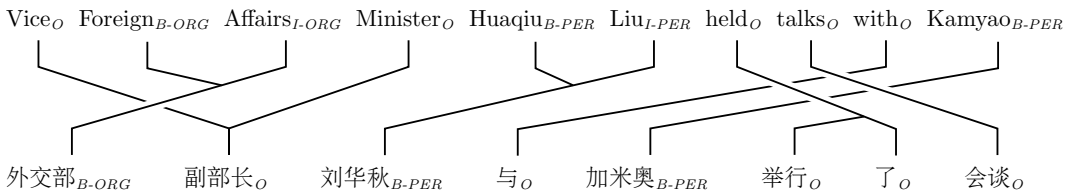


Figure 5.1: Example of NER labels between two word-aligned bilingual parallel sentences.

sufficient context it can be difficult to distinguish the two; however, in Chinese, these two senses are disambiguated: "乔丹" as a last name, and "约旦" as a country name.

A severe limitation of methods employing bilingual projection is that they can only be applied to test scenarios where parallel sentence pairs are available. It is more desirable to improve monolingual system performance, which is more broadly applicable.

In this chapter, we consider a semi-supervised learning scheme using unannotated bitext. For a given language pair (e.g., English-Chinese), we expect one language (e.g. English) to have more annotated training resources than the other (e.g. Chinese), and thus there exists a strong monolingual model (for English) and a weaker model (for Chinese). Since bitext contains translations across the two languages, an aligned sentence pair would exhibit some semantic and syntactic similarities. Thus we can constrain the two models to agree with each other by making joint predictions that are skewed towards the more informed model. In general, errors made in the lower-resource model will be corrected by the higher-resource model, but we also anticipate that these joint predictions will have higher quality for both languages than the output of a monolingual model alone. We can then apply this bilingual annotation method to a large amount of unannotated bitext, and use the resulting annotated data as additional training data to train a new monolingual model with better coverage.[1]

A core contribution of this chapter is a new factored probabilistic model that chains together two CRF monolingual models with a multi-component cross-lingual factored model based on word alignments. The latter component encourages soft label agreement across the two CRF models, and significantly improves the performance of the weaker of the two.

The challenge in working with this new model is that it is computationally intractable to perform exact inference over it. Therefore, we propose a series of approximate inference

[1]This training regimen is also referred to as "uptraining" (Petrov et al., 2010).

methods which includes a MCMC sampler, an ILP-formulation, and a dual-decomposition algorithm. We can further improve the quality of bilingual prediction by incorporating an additional model, expanding upon Finkel et al. (2005), that enforces global label consistency for each language, as well as joint inference over NER and word alignment.

Experiments show that our bilingual method yields significant improvements over the state-of-the-art monolingual NER system as well as past methods for bilingual NER.

5.2 Bilingual Factored Model with Constraints

A pair of aligned sentences in two languages contains complementary cues to aid the analysis of each other. For example, in Figure 5.1, it is not immediately obvious whether the phrase "Foreign Affairs" on the English side refers to an organization (Ministry of Foreign Affairs), or general foreign affairs. But the aligned word on the Chinese side is a lot less ambiguous, and can be easily identified as an ORGANIZATION entity.

Another example is that in the Chinese training data we have never seen the translation of the name "Kamyao". As a result, the tagger cannot make use of lexical features, and so has to rely on less informative contextual features to predict if it is a GPE (Geo-Political Entity) or a PERSON. But we have seen the aligned word on the English side being tagged as PERSON, and thus can infer that the Chinese aligned entity should also be a PERSON.

It is straight-forward to see that accurate word alignment is essential in such an analysis. Fortunately, there are automatic word alignment systems used in MT research that produce robust and accurate alignment results, and our method will use the output of one (Liang et al., 2006).

We present a probabilistic formulation for bilingual NER, and show how it leads to an optimization problem with hard and soft bilingual constraints.

The new probability objective function that we are trying to optimize is the joint probability of label sequences $\boldsymbol{y}^e$ (English) and $\boldsymbol{y}^c$ (Chinese)[2] given input sentences $\boldsymbol{x}^e$ (English)

[2] $\boldsymbol{y}^c$ is also used to denote other foreign languages such as German when English-German language pair is used.

and $\boldsymbol{x}^c$ (Chinese), shown as:

$$P(\boldsymbol{y}^c, \boldsymbol{y}^e | \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A}) \tag{5.1}$$

where $\mathcal{A}$ is the set of word alignment pairs.

Through the following derivation by applying Bayes rule, we can derive Eq. 5.4 from Eq. 5.1 as:

$$P(\boldsymbol{y}^c, \boldsymbol{y}^e | \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A}) = \frac{P(\boldsymbol{y}^c, \boldsymbol{y}^e, \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})}{P(\boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})} \tag{5.2}$$

$$= \frac{P(\boldsymbol{y}^c, \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})}{P(\boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})} \cdot \frac{P(\boldsymbol{y}^e, \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})}{P(\boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})} \cdot \frac{P(\boldsymbol{y}^c, \boldsymbol{y}^e, \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A}) P(\boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})}{P(\boldsymbol{y}^c, \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A}) P(\boldsymbol{y}^e, \boldsymbol{x}^c, \boldsymbol{x}^e, \mathcal{A})} \tag{5.3}$$

$$\approx P_{CRF}(\boldsymbol{y}^c | \boldsymbol{x}^c) P_{CRF}(\boldsymbol{y}^e | \boldsymbol{x}^e) \frac{P(\boldsymbol{y}^c, \boldsymbol{y}^e | \mathcal{A})}{P(\boldsymbol{y}^c | \mathcal{A}) P(\boldsymbol{y}^e | \mathcal{A})} \tag{5.4}$$

Eq. (5.3) just simplifies by canceling to give Eq. (5.2). The first two items of Eq. (5.4) come from the first two items of Eq. (5.3), by assuming that the named entity output sequence of one language is only dependent on the input sequence of the language. Alternatively, if we assume that the named entity output sequences of two languages are only dependent on the word alignments between them, then the third item of Eq. (5.3) can be converted into the last item of Eq. (5.4). While these assumptions are not always true (and sometimes conflicting), this suggests that these probability factors can capture the major sources of information for NER tagging of a a parallel corpus.

The first two items of Eq. (5.4) are simply the monolingual conditional probabilities modeled by a standard CRF model as shown in Eq. 2.1.

Assuming that the labels are independent between different word alignment pairs, then the last item of Eq. (5.4) can be decomposed into:

$$\frac{P(\boldsymbol{y}^c, \boldsymbol{y}^e | \mathcal{A})}{P(\boldsymbol{y}^c | \mathcal{A}) P(\boldsymbol{y}^e | \mathcal{A})} = \prod_{a \in \mathcal{A}} \frac{P(y_a^c y_a^e)}{P(y_a^e) P(y_a^e)} \tag{5.5}$$

$$= \prod_{a \in \mathcal{A}} \lambda_a^{y^c y^e} \tag{5.6}$$

where y_a^c and y_a^e respectively denotes Chinese and English named entity labels in a word alignment pair a.

$\lambda^{y^c y^e} = \frac{P(y^c y^e)}{P(y^c)P(y^e)}$ assigns a probability score to the likelihood that y^c and y^e co-occur over a pair of aligned words across the two languages. Another way of interpreting $\lambda^{y^c y^e}$ factors is that they enforce a degree of agreement between the label pair—label pairs that are in agreement will be assigned higher scores. We will go into more details of different variants of $\lambda^{y^c y^e}$ and the intuition they capture in the next two sub-sections.

5.2.1 Hard Agreement Constraints

We first introduce the simplest *hard* constraints, i.e. each word alignment pair should have the same named entity label. For example, in Figure 5.1, the Chinese word "外交部" was aligned with the English words "Foreign" and "Affair". Therefore, they should have the same named entity label ORGANIZATION.[3] Similarly, "Huaqiu" and "Liu" as well as "刘华秋" were all tagged with the label PERSON.

This constraint can be enforced by simply setting $\lambda_a^{y^c y^e} = 1$ for all matching y^c and y^e pairs (i.e., $y^c = y^e$).

This version of constraints is denoted as `hard`.

5.2.2 Soft Agreement Constraints

If we apply hard agreement constraints, any output sequence pairs that disagree on any tag pair will be assigned zero probability. Such a hard constraint is not always satisfied in practice, since annotation standards in different languages can differ. An example is given in Figure 5.2a, where the phrase mention of "development zone" is considered a LOCATION in the Chinese gold-standard, but not in the English gold-standard.

We can soften these constraints by replacing the 1 and 0 value assignments to $\lambda_a^{y^c y^e}$ in `hard` constraints with a soft probability measure. The value $\lambda_a^{y^c y^e}$ is set to be the *pairwise mutual information* (PMI) score of the entity label pair (y^c, y^e). To calculate the PMI score, an annotated bilingual NER corpus is consulted. We count from all word alignment pairs

[3] The prefix B- and I- are ignored.

the number of times y^c and y^e occur together ($C(y^c y^e)$) and separately ($C(y^c)$ and $C(y^e)$). Afterwards, $\lambda^{y^c y^e}$ is calculated with maximum likelihood estimation as follows:

$$\lambda^{y^c y^e} = \frac{P(y^c y^e)}{P(y^c)P(y^e)} = \frac{N \times C(y^c y^e)}{C(y^c)C(y^e)} \tag{5.7}$$

where N is the total number of word alignment pairs.

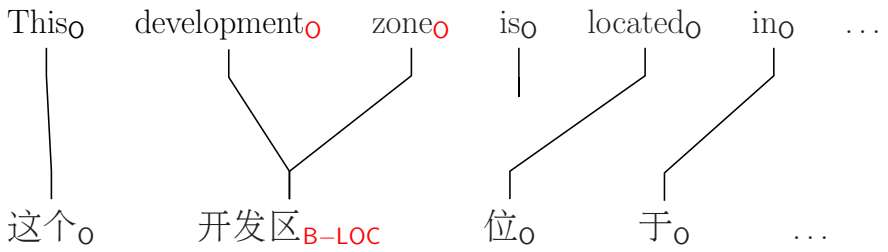
However, in this paper, we assume that no manually annotated bilingual corpus is available. Thus, the above method is only used as `oracle`. A realistic method for calculating the $\lambda^{y^c y^e}$ score requires the use of two initial monolingual NER models, such as baseline CRF, to tag named entity labels for each language on an unannotated bilingual corpus. We count from this automatically tagged corpus the statistics mentioned above. This method is henceforth referred to as `auto`.

Together we refer to these two versions of soft-agreement as `soft-label`.

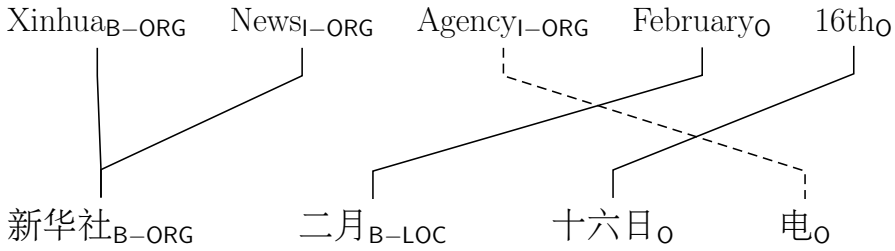
5.2.3 Alignment Uncertainty

When we consider the previous two sets of bilingual constraints, we assume the word alignments are given by some off-the-shelf alignment model which outputs a set of "hard" alignments. But inevitably, automatically induced word alignments contain errors, and a wrongly assigned word pair can cause the inclusion of wrong agreement constraints. For example, in Figure 5.2b, the English word "Agency" is wrongly aligned with the Chinese word "电 (report)", and we should not expect these two words to have the same entity.

In practice, most statistical word alignment models assign a probability to each alignment pair, and "hard" alignments are produced by cutting off alignment pairs that fall below a certain threshold θ. This condition can be regarded as a kind of *hard word alignment*. However, the following problem exists: the smaller the θ, the noisier the word alignments are; the larger the θ, the more possible word alignments are lost. To ameliorate this problem, we introduce another set of soft bilingual constraints.



(a) Inconsistent named entity standards



(b) Word alignment error

Figure 5.2: Errors of hard bilingual constraints method.

To take into account alignment uncertainties, we can re-express Eq. (5.6) as follows:

$$\prod_{a \in \mathcal{A}} \lambda_a^{y^c y^e} = \prod_{a \in \mathcal{A}} (\lambda_a^{y^c y^e})^{I_a} \tag{5.8}$$

$I_a = 1$ *iff* $P_a \geq \theta$; otherwise, $I_a = 0$; P_a is the posterior alignment probability given by the automatic aligner toolkit, and θ is some threshold value set experimentally. In this modification, we are exponentiating the PMI value to the power of the alignment probability. The intuition behind this modification is that pairs with a higher alignment probability will reflect more probability fluctuation when different label assignments are considered.

For example, consider an extreme case where a particular pair of aligned words has alignment probability 0. Then the value of the $\lambda_a^{y^c y^e}$ will always be 1 regardless of what tags are assigned to the two words, thus reducing the impact of different choices of tags for this pair in the overall tag sequence assignment.

We name this set of constraints that takes into account alignment uncertainty `soft-align`.

5.3 Approximate Inference

In a monolingual setting, exact inference in a standard linear-chain CRF can be done by applying the Viterbi algorithm to find the most likely output sequence. But when we consider the joint probability of an output sequence pair in a bilingual setting, especially when we apply the aforementioned bilingual constraints, cyclic cliques are introduced into the Markov random field which make exact inference algorithms intractable. We propose three different approximate inference algorithms, and contrast their tradeoffs through empirical experiments in Section 5.4.1. The first method we propose is to formulate the problem as solving a Integer Linear Program (ILP), and apply off-the-shelf ILP solvers for inference. This method has a direct connection with well-understood ILP literature, and performs quite well in terms of accuracy. The drawback is that speed and efficiency can become a serious issue when it is deployed to real-world large datasets. A second method we propose also aims to solve this problem through LP-relaxation. And the algorithm we adopt is called *dual decomposition*. It has the advantage that it gives strong theoretical guarantees under convergence conditions, and tends to run very fast in practice. The last method is a Gibbs sampling baed approach where we draw samples from the joint probability distribution. Like dual decomposition, it is also very fast in practice, and can be very useful when posterior probability estimates are desired for better integration with other up- or down-stream components in a Bayesian inference pipeline (Finkel et al., 2006).

5.3.1 Integer Linear Programming

We will first give a brief review of applying the Integer Linear Programming (ILP) framework for solving the standard monolingual NER problem, then we will introduce a new ILP formulation for bilingual NER tagging.

Monolingual ILP

Recall the BIO tagging scheme for NER that we introduced in Section 2.1. The label B-X (Begin) represents the first word of an named entity of type X, for example, PERSON or LOCATION. The label I-X (Inside) indicates that a word is part of an entity but not first

word; and the label O (Outside) is used for all non-entity words.

In Section 2.4, we explained that the inference problem in a standard linear-chain CRF involves finding the most likely output sequence that maximizes this objective, and is commonly done by the Viterbi algorithm. Roth and Yih (2005) proposed an ILP inference algorithm, which can capture more task-specific and global constraints than the vanilla Viterbi algorithm. But instead of directly solving the shortest-path problem in the ILP formulation, we take the marginal probabilities given by an underlying CRF model, and use ILP to solve the inference problem as a classification problem with constraints. Since the early HMM literature, it has been well known that using the marginal distributions at each position works well, as opposed to Viterbi MAP sequence labeling (Mérialdo, 1994).

We define the classification problem as:

$$\begin{aligned} \hat{\boldsymbol{y}} &= \text{argmax}_{\boldsymbol{y}}\, P(\boldsymbol{y}|\boldsymbol{x}) = \text{argmax}_{\boldsymbol{y}} \prod_i P(y_i|\boldsymbol{x}) \\ &= \text{argmax}_{\boldsymbol{y}} \sum_i \log P(y_i|\boldsymbol{x}) \end{aligned} \tag{5.9}$$

where $P(y_i|\boldsymbol{x})$ is the marginal probability taken from a underlying CRF model (*cf.* Eq. 2.1), computed using *forward-backward* inference.

This classification problem can then be expressed as an ILP:

$$\max \sum_{i=1}^{|\boldsymbol{x}|} \sum_{y \in Y} z_i^y \log P_i^y \tag{5.10}$$

where Y is the set of all possible named entity labels. $P_i^y = P(y_i = y|\boldsymbol{x})$ is the CRF marginal probability that the i^{th} word is tagged with a label y, and z_i^y is an indicator that equals to 1 *iff* the i^{th} word is labeled y; otherwise, z_i^y is equal to 0.

If no constraints are identified, then Eq. (5.10) achieves maximum when all z_i^y are assigned to 1, which violates the condition that each word should only be assigned a single entity label. We can express this with constraints:

$$\forall i : \sum_{y \in Y} z_i^y = 1 \tag{5.11}$$

After adding the constraints, the probability of the sequence is maximized when each word is assigned the tag with highest probability. However, some invalid results may still exist. For example a label O may be wrongly followed by a label I-X, although a named entity cannot start with the label I-X. Therefore, we can add the following constraints:

$$\forall i, \forall \mathsf{X} : z_{i-1}^{\mathsf{B\text{-}X}} + z_{i-1}^{\mathsf{I\text{-}X}} - z_i^{\mathsf{I\text{-}X}} \geq 0 \tag{5.12}$$

which specifies that when the i^{th} word is tagged with I-X ($z_i^{\mathsf{I\text{-}X}} = 1$), then the previous word can only be tagged with B-X or I-X ($z_{i-1}^{\mathsf{B\text{-}X}} + z_{i-1}^{\mathsf{I\text{-}X}} \geq 1$).

ILP with Bilingual Constraints

The objective function for bilingual NER can be expressed as follows:

$$\max \sum_{i=1}^{|\boldsymbol{x}^c|} \sum_{y \in Y} z_i^y \log P_i^y + \sum_{j=1}^{|\boldsymbol{x}^e|} \sum_{y \in Y} z_j^y \log P_j^y \tag{5.13}$$

where P_i^y and P_j^y are the probabilities of the i^{th} Chinese word and j^{th} English word to be tagged with a named entity label y, respectively. $\boldsymbol{x}^c$ and $\boldsymbol{x}^e$ are respectively the Chinese and English sentences.

Similar to monolingual constrained NER (Section 5.3.1), monolingual constraints are added for each language as shown in Eqs. (5.14) and (5.15):

$$\forall i : \sum_{y \in Y} z_i^y = 1; \forall j : \sum_{y \in Y} z_j^y = 1 \tag{5.14}$$

$$\forall i, \forall \mathsf{X} : z_i^{\mathsf{B\text{-}X}} + z_i^{\mathsf{I\text{-}X}} - z_{i+1}^{\mathsf{B\text{-}X}} \geq 0 \tag{5.15}$$

$$\forall j, \forall \mathsf{X} : z_j^{\mathsf{B\text{-}X}} + z_j^{\mathsf{I\text{-}X}} - z_{j+1}^{\mathsf{B\text{-}X}} \geq 0$$

`Hard` bilingual constraints can be added in Eq. (5.16) as:

$$\forall (i,j), \forall\, \mathsf{X} \in \mathcal{A} : z_i^{\mathsf{B\text{-}X}} + z_i^{\mathsf{I\text{-}X}} = z_j^{\mathsf{B\text{-}X}} + z_j^{\mathsf{I\text{-}X}} \tag{5.16}$$

where $\mathcal{A} = \{(i,j)\}$ is the word alignment pair set, i.e. the i^{th} Chinese word and the j^{th}

English word were aligned together. Chinese word i is tagged with a named entity type X ($z_i^{\text{B-X}} + z_i^{\text{I-X}} = 1$), *iff* English word j is tagged with X ($z_j^{\text{B-X}} + z_j^{\text{I-X}} = 1$). Therefore, these *hard* bilingual constraints guarantee that when two words are aligned, they are tagged with the same named entity label.

To incorporate the `soft-align` constraints, we use ILP to maximize Eq. (5.4). The new objective function is expressed as follow:

$$\max \sum_{i=1}^{|\boldsymbol{x}^c|} \sum_{y \in Y} z_i^y \log P_i^y + \sum_{j=1}^{|\boldsymbol{x}^e|} \sum_{y \in Y} z_j^y \log P_j^y + \sum_{a \in \mathcal{A}} \sum_{y^c \in Y} \sum_{y^e \in Y} z_a^{y^c y^e} \log \lambda_a^{y^c y^e} \tag{5.17}$$

where $z_a^{y^c y^e}$ is an indicator that equals 1 *iff* the Chinese named entity label is y^c and the English named entity label is y^e, given a word alignment pair a; otherwise, $z_a^{y^c y^e}$ is 0.

In addition, one and only one possible named entity label pair exists for a word alignment pair. This condition can be expressed as the following constraints:

$$\forall a \in \mathcal{A} : \sum_{y^c \in Y} \sum_{y^e \in Y} z_a^{y^c y^e} = 1 \tag{5.18}$$

When the label pair of a word alignment pair is determined, the corresponding monolingual named entity labels can also be identified. This rule can be expressed by the following constraints:

$$\forall a = (i, j) \in \mathcal{A} : z_a^{y^c y^e} \leq z_i^{y^c}, z_a^{y^c y^e} \leq z_j^{y^e} \tag{5.19}$$

Thus, if $z_a^{y^c y^e} = 1$, then $z_i^{y^c}$ and $z_j^{y^e}$ must be both equal to 1. Here, the i^{th} Chinese word and the j^{th} English word are aligned together.

Similarly, we can also express the `soft-align` constraints like above. The change in the agreement factor calculation from `soft-label` to `soft-align` does not affect the ILP formulation.

	Chinese			English		
	P	R	F_1	P	R	F_1
CRF	76.89	61.64	68.42	81.98	74.59	78.11
Monolingual ILP	76.05	61.38	67.93	81.91	72.60	76.97
Burkett	77.52	65.84	71.20	82.28	76.64	79.36
ILP-`hard`	74.23	64.10	68.79	81.69	73.28	77.25
ILP-`soft-label (auto)`	77.22	69.46	73.13	80.39	76.66	78.48
ILP-`soft-align (auto)`	77.56	71.10	74.19	80.93	76.65	78.73

Table 5.1: ILP results on bilingual parallel test set.

ILP Results

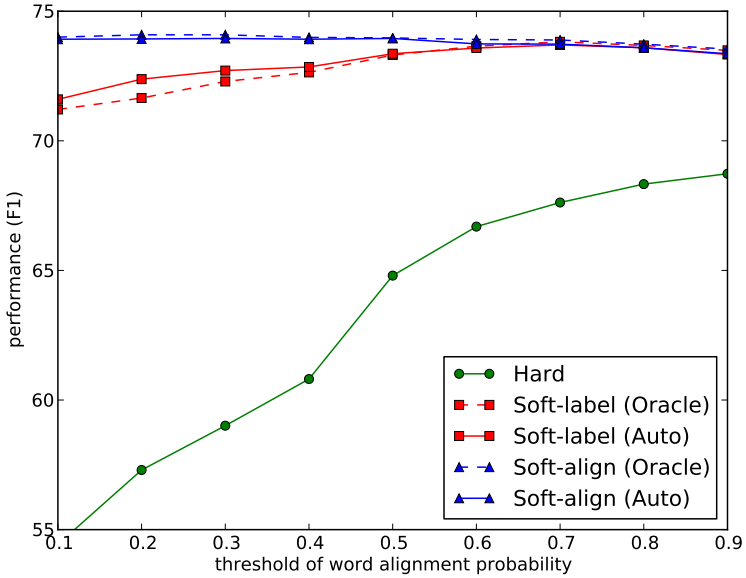
We conduct NER experiments using the experimental setup described in Section 2.5.2. The main results on Chinese and English test sets with the optimal word alignment threshold for each method are shown in Table 5.1.

The performance of ILP with only monolingual constraints is quite comparable with the CRF results, especially on the English side. The greater ILP performance on English is probably due to more accurate marginal probabilities estimated by the English CRF model.

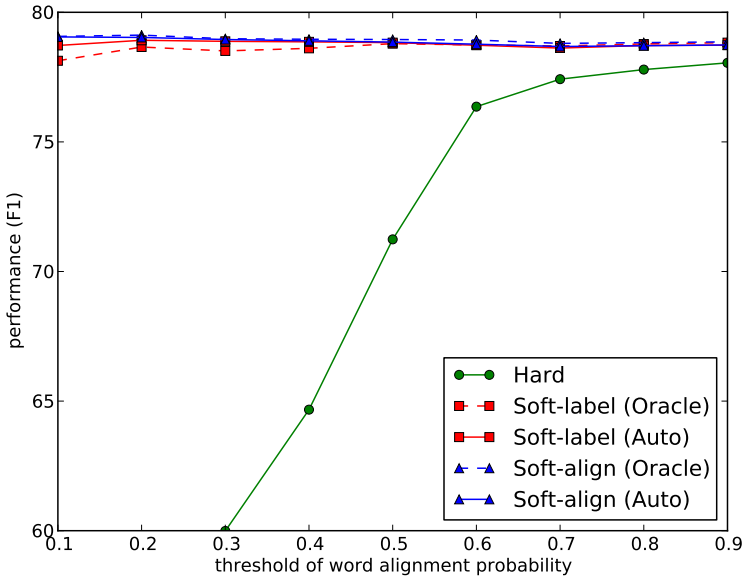
The ILP model with hard bilingual constraints gives a slight performance improvement on Chinese, but affects performance negatively on English. Once we introduced labeling uncertainties into the `soft-label` bilingual constraints, we see a very significant ($p < 0.01$) performance boost on Chinese. This method also improves the recall on the English, with a smaller decrease in precision. Overall, it improves the English F_1 score by about 0.62%, which is unfortunately not statistically significant.

Compared with `soft-label`, the final `soft-align` method can further improve performance on both Chinese and English. This is likely to be because: 1) `soft-align` includes more word alignment pairs, thereby improving recall; and 2) uses probabilities to cut wrong word alignments, thereby improving precision. In particular, the gain on the Chinese side is almost 5.8% in absolute F_1 score.

Threshold Tuning During development, we tuned the word alignment probability thresholds to find the best value. Figure 5.3 shows the performance curves.



(a) CH performance curve



(b) EN performance curve

Figure 5.3: ILP alignment probability threshold tuning results.

When the word alignment probability threshold θ is set to 0.9, the hard bilingual constraints perform quite well for both Chinese and English. But as the thresholds value gets smaller, and more noisy word alignments are introduced, we see the hard bilingual constraints method starts to perform badly.

In `soft-label` setting, where inconsistent label assignments within aligned word pairs are allowed but penalized, different languages have different optimal threshold values. For example, Chinese has an optimal threshold of 0.7, whereas English has 0.2. Thus, the optimal thresholds for different languages need to be selected with care when `soft-label` is applied in practice. `soft-align` eliminates the need for careful tuning of word alignment thresholds, and therefore can be more easily used in practice. Experimental results of `soft-align` confirms our hypothesis – the performance of both Chinese and English NER systems improves with decreasing threshold. However, we can still improve efficiency by setting a low threshold to prune away very unlikely word alignments. We set the threshold to 0.1 for `soft-align` to increase speed, and we observed very minimal performance lost when doing so.

We also found that automatically estimated bilingual label PMI scores (`Auto`) gave comparable results to `Oracle`. Therefore this technique is effective for computing the PMI scores, avoiding the need of manually annotated data.

5.3.2 Dual Decomposition

The standard procedure to solve the ILP formulation we introduced in the previous subsection is to apply an off-the-shelf ILP solver. But general-purpose ILP solvers are usually not very efficient and cannot easily be optimized for our specific application due to proprietary nature. In this sub-section, we introduce another method for solving the ILP, called *dual decomposition* (Komodakis et al., 2007; Rush et al., 2010). At a very high level, dual decomposition method takes a ILP problem that is hard to decode in its original form, and takes the Lagrangian Relaxation of the ILP, and decomposes the dual of the relaxed objective into multiple sub-problems that can be decoded independently, subject to a set of linear constraints that enforces a form of agreement between the sub-problems. The algorithm then iteratively solves the joint objective by decoding each sub-problem separately

subject to the current constraints, and update constraints based on agreement status of the decoded output from each sub-problem.

Recall the standard definition of a linear-chain CRF (*cf.* Eq. 2.1) is:

$$P_{CRF}(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta}) = \frac{\prod_{i=1}^{|\boldsymbol{x}|} \Phi(\boldsymbol{x}, y_i, y_{i-1};\boldsymbol{\theta})}{Z(\boldsymbol{x};\boldsymbol{\theta})} \tag{5.20}$$

where $\Phi(\boldsymbol{x}, y_i, y_{i-1};\boldsymbol{\theta})$ is the clique potential at position i, and $Z(\boldsymbol{x};\boldsymbol{\theta})$ is the partition function for input sequence $\boldsymbol{x}$. We let $k(\boldsymbol{y}^e)$ be the un-normalized log-probability of an English CRF model $P_{CRF^e}(\boldsymbol{y}^e|\boldsymbol{x}^e)$ for English input sentence $\boldsymbol{x}^e$, defined as:

$$k(\boldsymbol{y}^e) = \log\left(\prod_{i=1}^{|\boldsymbol{x}^e|} \Phi(\boldsymbol{x}^e, {y^e}_i, {y^e}_{i-1};\boldsymbol{\theta})\right)$$

Similarly, we define model $P_{CRF^c}(\boldsymbol{y}^c|\boldsymbol{x}^c)$ and un-normalized log-probability $l(\boldsymbol{y}^c)$ for Chinese.

We also assume that a set of word alignments ($\mathcal{A} = \{(i,j) : e_i \leftrightarrow c_j\}$) is given by a word aligner and remain fixed in our model.

For clarity, we assume $\boldsymbol{y}^e$ and $\boldsymbol{y}^c$ are binary variables in the description of our algorithms. The extension to the multi-class case is straight-forward and does not affect the core algorithms.

Hard Agreement

We first consider a bilingual NER model subject to hard agreement constraints (*cf.* Section 5.2.1) over entity labels of aligned word pairs, and call this model BI-NER. At inference time, we solve the following optimization problem:

$$\begin{aligned}
&\max_{\boldsymbol{y}^e, \boldsymbol{y}^c} \log\left(P_{CRF_e}\left(\boldsymbol{y}^e\right)\right) + \log\left(P_{CRF_c}\left(\boldsymbol{y}^c\right)\right) \\
=&\max_{\boldsymbol{y}^e, \boldsymbol{y}^c} k(\boldsymbol{y}^e) + l(\boldsymbol{y}^c) - \log Z_e(\boldsymbol{x}^e) - \log Z_c(\boldsymbol{x}^c) \\
\simeq&\max_{\boldsymbol{y}^e, \boldsymbol{y}^c} k(\boldsymbol{y}^e) + l(\boldsymbol{y}^c) \\
&\ni y_i^e = y_j^c \;\; \forall (i,j) \in \mathcal{A}
\end{aligned}$$

We dropped the $\log Z_e(\boldsymbol{x}^e)$ and $\log Z_c(\boldsymbol{x}^c)$ terms because they remain constant at inference time.

The Lagrangian relaxation of this term is:

$$L\left(\boldsymbol{y}^e, \boldsymbol{y}^c, \mathbf{U}\right) = k\left(\boldsymbol{y}^e\right) + l\left(\boldsymbol{y}^c\right) + \sum_{(i,j)\in\mathcal{A}} u(i,j)\left(y_i^e - y_j^c\right)$$

where $u(i,j)$ are the Lagrangian multipliers.

Instead of solving the Lagrangian directly, we can form the dual of this problem and solve it using dual decomposition (Rush et al., 2010):

$$\min_{\mathbf{U}}\left(\max_{y^e}\left[k\left(\boldsymbol{y}^e\right) + \sum_{(i,j)\in\mathcal{A}} u(i,j) y_i^e\right] + \max_{y^c}\left[l\left(\boldsymbol{y}^c\right) - \sum_{(i,j)\in\mathcal{A}} u(i,j) y_j^c\right]\right)$$

We solve this dual decomposition problem by iteratively updating the sub-gradient as depicted in Algorithm 1. In each iteration, we first decode the two argmax sub-problems separately with respect to the current set of constraints. Then we check the label assignments agreements from the two sub-problems' solutions; if the label assignments at all positions agree, we return the current decoded sequences and declare victory; but if the label assignments over a pair of words linked by an alignment link do not agree, we update the accumulative penalties (of opposite signs) at the two positions across the language pair.

T is the maximum number of iterations before early stopping, and α_t is the learning rate at time t. We adopt a learning rate update rule from Koo et al. (2010) where α_t is defined as $\frac{1}{N}$, where N is the number of times we observed a consecutive dual value increase from iteration 1 to t.

Rush and Collins (2012) gives a full tutorial on dual decompositions and showed more examples where dual decomposition applies to NLP problems, such as high-order dependency parsing, machine translation decoding and POS tagging with inter-sentential consistency. In Section 5.3.3, we present another technique based on MCMC sampling that also models inter-sentential consistency, specifically for NER problems.

Algorithm 1 Dual decomposition inference algorithm for hard agreement model.

$\forall (i,j) \in \mathcal{A} : \; u(i,j) = 0$
for $t \leftarrow 1$ **to** T **do**
  $\boldsymbol{y}^{e*} \leftarrow \text{argmax} \; k\left(\boldsymbol{y}^{e}\right) + \sum_{(i,j)\in\mathcal{A}} u(i,j) y_i^e$
  $\boldsymbol{y}^{c*} \leftarrow \text{argmax} \; l\left(\boldsymbol{y}^{c}\right) - \sum_{(i,j)\in\mathcal{A}} u(i,j) y_j^c$
  if $\forall (i,j) \in \mathcal{A} \colon y_i^{e*} = y_j^{c*}$ **then**
    return $(\boldsymbol{y}^{e*}, \boldsymbol{y}^{c*})$
  end if
  for all $(i,j) \in \mathcal{A}$ **do**
    $u(i,j) \leftarrow u(i,j) + \alpha_t \left(y_j^{c*} - y_i^{e*}\right)$
  end for
end for
return $\left(\mathbf{y}^{\mathbf{e*}}_{(\mathbf{T})}, \mathbf{y}^{\mathbf{c*}}_{(\mathbf{T})}\right)$

Soft Agreement

As we discussed in Section 5.2.3, the hard agreement model rests on the core assumption that aligned words must have identical entity tags, which does not always hold in reality. In order to model alignment uncertainty, we extend the two previously independent CRF models into a larger undirected graphical model, by introducing a cross-lingual edge factor $\phi(i,j)$ for every pair of word positions $(i,j) \in \mathcal{A}$. We associate a clique potential function

$h_{(i,j)}(y_i^e, y_j^c)$ for $\phi(i,j)$:

$$h_{(i,j)}\left(y_i^e, y_j^c\right) = \mathrm{PMI}\left(y_i^e, y_j^c\right)^{\hat{P}(e_i,c_j)}$$

where $\mathrm{PMI}(y_i^e, y_j^c)$ is the point-wise mutual information (PMI) of the tag pair, and we raise it to the power of a posterior alignment probability $\hat{P}(e_i, c_j)$. For a pair of named entities that are aligned with low probability, we cannot be too sure about the association of the two named entities, therefore the model should not impose too much influence from the bilingual agreement model; instead, we will let the monolingual named entity models make their decisions, and trust that those are the best estimates we can come up with when we do not have much confidence in their bilingual association. The use of the posterior alignment probability facilitates this purpose.

Initially, each of the cross-lingual edge factors will attempt to assign a pair of tags that has the highest PMI score, but if the monolingual taggers do not agree, a penalty will start accumulating over this pair, until some other pair that agrees better with the monolingual models takes the top spot. Simultaneously, the monolingual models will also be encouraged to agree with the cross-lingual edge factors. This way, the various components effectively trade penalties indirectly through the cross-lingual edges, until a tag sequence that maximizes the joint probability is achieved.

Since we assume no bilingually annotated NER corpus is available, in order to get an estimate of the PMI scores, we first tag a collection of unannotated bilingual sentence pairs using the monolingual CRF taggers, and collect counts of aligned entity pairs from this auto-generated tagged data.

Each of the $\phi(i,j)$ edge factors (e.g., the edge between node c_3 and e_4 in Figure 5.1) overlaps with each of the two CRF models over one vertex (e.g., c_3 on Chinese side and e_4 on English side), and we seek agreement with the Chinese CRF model over tag assignment of c_j, and similarly for e_i on English side. In other words, no direct agreement between the two CRF models is enforced, but they both need to agree with the bilingual edge factors.

The updated optimization problem becomes:

$$\max_{\boldsymbol{y}^{e^{(k)}}\boldsymbol{y}^{c^{(l)}}\boldsymbol{y}^{e^{(h)}}\boldsymbol{y}^{c^{(h)}}} k\left(\boldsymbol{y}^{e^{(k)}}\right) + l\left(\boldsymbol{y}^{c^{(l)}}\right) + \sum_{(i,j)\in\mathcal{A}} h_{(i,j)}\left(y_i^{e^{(h)}}, y_j^{c^{(h)}}\right)$$
$$\text{such that } \forall(i,j)\in\mathcal{A}\colon \left(y_i^{e^{(k)}} = y_i^{e^{(h)}}\right) \wedge \left(y_j^{c^{(l)}} = y_j^{c^{(h)}}\right)$$

where the notation $y_i^{e^{(k)}}$ denotes tag assignment to word e_i by the English CRF and $y_i^{e^{(h)}}$ denotes assignment to word e_i by the bilingual factor; $y_j^{c^{(l)}}$ denotes the tag assignment to word c_j by the Chinese CRF and $y_j^{c^{(h)}}$ denotes assignment to word c_j by the bilingual factor.

The updated dual decomposition algorithm is illustrated in Algorithm 2. We introduce two separate sets of dual constraints $\mathbf{w^e}$ and $\mathbf{w^c}$, which range over the set of vertices on their respective half of the graph. Decoding the edge factor model $h_{(i,j)}(y_i^e, y_j^c)$ simply involves finding the pair of tag assignments that gives the highest PMI score, subject to the dual constraints.

The way dual decomposition algorithms work in decomposing undirected graphical models is analogous to other message passing algorithms such as loopy belief propagation, but dual decomposition gives a stronger optimality guarantee upon convergence (Rush et al., 2010).

Joint Alignment and NER Decoding

In this section we develop an extended model in which NER information can in turn be used to improve alignment accuracy. Although we have seen more than a handful of recent papers that apply the dual decomposition method for joint inference problems, all of the past work deals with cases where the various model components have the same inference output space (e.g., dependency parsing (Koo et al., 2010), POS tagging (Rush et al., 2012), etc.). In our case the output space is the much more complex joint alignment and NER tagging space. We propose a novel dual decomposition variant for performing inference over this joint space.

Most commonly used alignment models, such as the IBM models and HMM-based aligner are unsupervised learners, and can only capture simple distortion features and lexical translational features due to the high complexity of the structure prediction space. On

Algorithm 2 Dual decomposition inference algorithm for soft agreement model.

for $t \leftarrow 1$ **to** T **do**
  $\boldsymbol{y}^{e^{(k)*}} \leftarrow \text{argmax } f\left(\boldsymbol{y}^{e^{(k)}}\right) + \sum_{i \in |\mathbf{e}|} w_i^e y_i^{e^{(k)}}$
  $\boldsymbol{y}^{c^{(l)*}} \leftarrow \text{argmax } g\left(\boldsymbol{y}^{c^{(l)}}\right) + \sum_{i \in |\mathbf{c}|} w_j^c y_j^{c^{(l)}}$
  for all $(i,j) \in \mathcal{A}$ **do**
    $(y_i^{e^{(h)*}} y_j^{c^{(h)*}}) \leftarrow \text{argmax } h_{(i,j)}(y_i^{e^{(q)}} y_j^{c^{(q)}}) - w_i^e y_i^{e^{(h)}} - w_j^c y_j^{c^{(h)}}$
  end for
  Converged = $(\boldsymbol{y}^{e^{(k)}} = \boldsymbol{y}^{e^{(q)}} \wedge \mathbf{y}^{c^{(l)}} = \boldsymbol{y}^{c^{(q)}})$
  if Converged = **true** , **then**
    return $\left(\boldsymbol{y}^{e^{(k)*}}, \boldsymbol{y}^{c^{(l)*}}\right)$
  else
    for all $i \in |\mathbf{e}|$ **do**
      $w_i^e \leftarrow w_i^e + \alpha_t \left(y_i^{e^{(q|h)*}} - y_i^{e^{(k)*}}\right)$
    end for
    for all $j \in |\mathbf{c}|$ **do**
      $w_j^c \leftarrow w_j^c + \alpha_t \left(y_j^{c^{(q|h)*}} - y_j^{c^{(l)*}}\right)$
    end for
  end if
end for
return $\left(\boldsymbol{y}_{(T)}^{e^{(k)*}}, \boldsymbol{y}_{(T)}^{c^{(l)*}}\right)$

the other hand, the CRF-based NER models are trained on manually annotated data, and admit richer sequence and lexical features. The entity label predictions made by the NER model can potentially be leveraged to correct alignment mistakes. For example, in Figure 5.1, if the tagger knows that the word "Agency" is tagged I-ORG, and if it also knows that the first comma in the Chinese sentence is not part of any entity, then we can infer it is very unlikely that there exists an alignment link between "Agency" and the comma.

To capture this intuition, we extend the BI-NER model to jointly perform word alignment and NER decoding, and call the resulting model BI-NER-WA. As a first step, instead of taking the output from an aligner as fixed input, we incorporate two uni-directional aligners into our model. We name the Chinese-to-English aligner model as $m(\mathbf{B}^e)$ and the reverse directional model $n(\mathbf{B}^c)$. $\mathbf{B}^e$ is a matrix that holds the output of the Chinese-to-English aligner. Each $b^e(i,j)$ binary variable in $\mathbf{B}^e$ indicates whether c_j is aligned to e_i; similarly we define output matrix $\mathbf{B}^c$ and $b^c(i,j)$ for Chinese. In our experiments, we used two HMM-based alignment models. But in principle we can adopt any alignment model as long as we can perform efficient inference over it.

We introduce a cross-lingual edge factor $\zeta(i,j)$ in the undirected graphical model for every pair of word indices (i,j), which predicts a binary variable $a(i,j)$ for an alignment link between e_i and c_j. The edge factor also predicts the entity tags for e_i and c_j.

The new edge potential q is defined as:

$$q_{(i,j)}\left(y_i^e, y_j^c, a(i,j)\right) = \log(P(a(i,j)=1)) + S(y_i^e, y_j^c | a(i,j))^{P(a(i,j)=1)}$$

$$S(y_i^e, y_j^c | a(i,j)) = \begin{cases} \text{pmi}(y_i^e, y_j^c), \text{if } a(i,j)=1 \\ 0, \text{else} \end{cases}$$

$P(a(i,j)=1)$ is the alignment probability assigned by the bilingual edge factor between node e_i and c_j. We initialize this value to $\hat{P}(e_i,c_j) = \frac{1}{2}(P_m(e_i,c_j) + P_n(e_i,c_j))$, where $P_m(e_i,c_j)$ and $P_n(e_i,c_j)$ are the posterior probabilities assigned by the HMM-aligners.

Algorithm 3 Dual decomposition inference algorithm for joint alignment and NER model.

$S \leftarrow \{(i,j) \colon \forall i \in |\mathbf{e}|, \forall j \in |\mathbf{c}|\}$
for $t \leftarrow 1$ **to** T **do**
  $\boldsymbol{y}^{e(k)*} \leftarrow \text{argmax } f\left(\boldsymbol{y}^{e(k)}\right) + \sum_{i \in |\mathbf{e}|} w_i^e y_i^{e(k)}$
  $\boldsymbol{y}^{c(l)*} \leftarrow \text{argmax } g\left(\boldsymbol{y}^{c(l)}\right) + \sum_{i \in |\mathbf{c}|} w_j^c y_j^{c(l)}$
  $\mathbf{B}^{e*} \leftarrow \text{argmax } m\left(\mathbf{B}^e\right) + \sum_{(i,j)} d^e(i,j) b^e(i,j)$
  $\mathbf{B}^{c*} \leftarrow \text{argmax } n\left(\mathbf{B}^c\right) + \sum_{(i,j)} d^c(i,j) b^c(i.j)$
  for all $(i,j) \in S$ **do**
    $(y_i^{e(q)*} y_j^{c(q)*} a(i,j)^*) \leftarrow \text{argmax } q_{(i,j)}(y_i^{e(q)} y_j^{c(q)} a(i,j))$
      $-w_i^e y_i^{e(q)} - w_j^c y_j^{c(q)} - d^e(i,j) a(i,j) - d^c(i,j) a(i,j)$
  end for
  Converged = $(\mathbf{B}^e{=}\mathbf{A}{=}\mathbf{B}^c \wedge \mathbf{y}^{e(k)}{=}\mathbf{y}^{e(q)} \wedge \mathbf{y}^{c(l)}{=}\mathbf{y}^{c(q)})$
  if Converged = **true** , **then**
    return $\left(\boldsymbol{y}^{e(k)*}, \boldsymbol{y}^{c(l)*}, \mathbf{A}\right)$
  else
    for all $(i,j) \in S$ **do**
      $d^e(i,j) \leftarrow d^e(i,j) + \alpha_t \left(a^*(i,j) - b^{e*}(i,j)\right)$
      $d^c(i,j) \leftarrow d^c(i,j) + \alpha_t \left(a^*(i,j) - b^{c*}(i,j)\right)$
    end for
    for all $(i,j) \in S \ni a^*(i,j) = 1$ **do**
      $w_i^e \leftarrow w_i^e + \alpha_t \left(y_i^{e(q)*} - y_i^{e(k)*}\right)$
      $w_j^c \leftarrow w_j^c + \alpha_t \left(y_j^{c(q)*} - y_j^{c(l)*}\right)$
    end for
  end if
end for
return $\left(\mathbf{y}_{(\mathbf{T})}^{\mathbf{e(k)}*}, \mathbf{y}_{(\mathbf{T})}^{\mathbf{c(l)}*}, \mathbf{A}_{(T)}\right)$

The joint optimization problem is defined as:

$$\max_{\boldsymbol{y}^{e^{(k)}}\boldsymbol{y}^{c^{(l)}}\boldsymbol{y}^{e^{(h)}}\boldsymbol{y}^{c^{(h)}}\mathbf{B}^e\mathbf{B}^c\mathbf{A}} k(\boldsymbol{y}^{e^{(k)}}) + l(\boldsymbol{y}^{c^{(l)}}) +$$
$$m(\mathbf{B}^e) + n(\mathbf{B}^c) + \sum_{(i\in|\mathbf{e}|,j\in|\mathbf{c}|)} q_{(i,j)}(y_i^{e^h}, y_j^{c^{(h)}}, a(i,j))$$
$$\ni \forall(i,j)\colon \big(b^e(i,j){=}a(i,j)\big) \wedge \big(b^c(i,j){=}a(i,j)\big)$$
$$\wedge \text{ if } a(i,j) = 1 \text{ then } \big(y_i^{e^{(k)}}{=}y_i^{e^{(h)}}\big) \wedge \big(y_j^{c^{(l)}}{=}y_j^{c^{(h)}}\big)$$

We include two dual constraints $d^e(i,j)$ and $d^c(i,j)$ over alignments for every bilingual edge factor $\zeta(i,j)$, which are applied to the English and Chinese sides of the alignment space, respectively.

The dual decomposition algorithm used for this model is given in Algorithm 3. One special note is that after each iteration when we consider updates to the dual constraint for entity tags, we only check tag agreements for cross-lingual edge factors that have an alignment assignment value of 1. In other words, cross-lingual edges that are not aligned do not affect bilingual NER tagging.

Similar to $\phi(i,j)$, $\zeta(i,j)$ factors do not provide that much additional information other than some selectional preferences via PMI score. But the real power of these cross-language edge cliques is that they act as a liaison between the NER and alignment models on each language side, and encourage these models to indirectly agree with each other by having them all agree with the edge cliques.

It is also worth noting that since we decode the alignment models with Viterbi inference, additional constraints such as the neighborhood constraint proposed by DeNero and Macherey (2011) can be easily integrated into our model. The neighborhood constraint enforces that if c_j is aligned to e_i, then c_j can only be aligned to e_{i+1} or e_{i-1} (with a small penalty), but not any other word position. We report results of adding neighborhood constraints to our model in the next section.

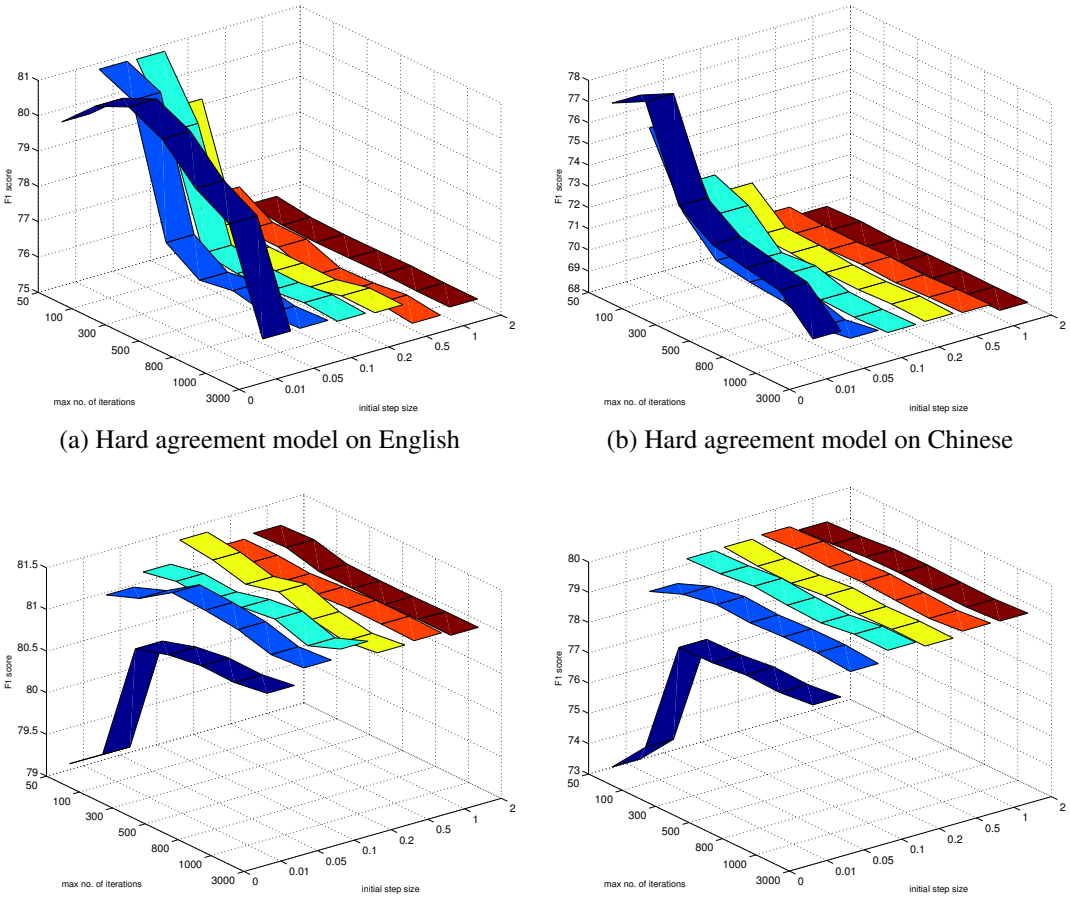
	Chinese			English		
	P	R	F_1	P	R	F_1
CRF	76.89	61.64	68.42	81.98	74.59	78.11
Burkett	77.52	65.84	71.20	82.28	76.64	79.36
ILP	77.56	71.10	74.19	80.93	76.65	78.73
Bi-NER	**79.14**	**71.55**	**75.15**	**82.58**	**77.96**	**80.20**

Table 5.2: Dual decomposition results on bilingual parallel test set.

Dual Decomposition Results

The main results on bilingual NER over the test portion of *full-set* are shown in Table 5.2. The algorithms are run for 1000 iterations and terminated if not converged. The decoded sequence of the last iteration is taken as final output. Best numbers on each measure that are statistically significantly better than the monolingual baseline and Burkett et al. (2010b) are highlighted in bold. We initially experimented with the hard agreement model, but it performs quite poorly for reasons we discussed in Section 5.2.2. The BI-NER model with soft agreement constraints, however, significantly outperforms all baselines, including the ILP method from Section 5.3.1. In particular, it achieves an absolute F_1 improvement of 6.7% in Chinese and 2.1% in English over the CRF monolingual baselines.

Hyper Parameter Tuning A well-known issue with the dual decomposition method is that when the model does not necessarily converge, then the procedure can be very sensitive to hyper-parameters such as initial step size and early termination criteria. If a model only gives good performance with well-tuned hyper-parameters, then we must have manually annotated data for tuning, which would significantly reduce the application and portability of this method to other language pairs and tasks. To evaluate the parameter sensitivity of our model, we run the model from 50 to 3000 iterations before early stopping, and with 6 different initial step sizes from 0.01 to 1. The results are shown in Figure 5.4. The numbers are obtained on the dev dataset, as a function of step size (*x-axis*) and maximum number of iterations before early stopping (*y-axis*). The soft agreement model does not seem to be sensitive to initial step size and almost always converges to a superior solution than the baseline.



(a) Hard agreement model on English

(b) Hard agreement model on Chinese

(c) Soft agreement model on English

(d) Soft agreement model on Chinese

Figure 5.4: Performance variance of the *hard* and *soft* agreement models.

Alignment Experiment Setup

For alignment experiments, we train two uni-directional HMM models as our baseline and monolingual alignment models. The parameters of the HMM were initialized by IBM Model 1 using the agreement-based EM training algorithms from Liang et al. (2006). Each model is trained for 2 iterations over a parallel corpus of 12 million English words and Chinese words, almost twice as much data as used in previous work that yields state-of-the-art unsupervised alignment results (DeNero and Klein, 2008; Haghighi et al., 2009; DeNero and Macherey, 2011).

Word alignment evaluation is done over the sections of OntoNotes that have matching

	NER-Chinese			NER-English			word alignment			
	P	R	F_1	P	R	F_1	P	R	F_1	AER
HMM-WA	-	-	-	-	-	-	**90.43**	40.95	56.38	43.62
Mono-CRF	82.50	66.58	73.69	84.24	78.70	81.38	-	-	-	-
Bi-NER	**84.87**	75.30	79.80	**84.47**	81.45	82.93	-	-	-	-
Bi-NER-WA	84.42	**76.34**	**80.18**	84.25	**82.20**	**83.21**	77.45	50.43	61.09	38.91
Bi-NER-WA+NC	84.25	75.09	79.41	84.28	82.17	**83.21**	76.67	**54.44**	**63.67**	**36.33**

Table 5.3: Joint alignment and NER test results. +NC means incorporating additional neighbor constraints from DeNero and Macherey (2011) to the model. The best number in each column is highlighted in bold.

gold-standard word alignment annotations from GALE Y1Q4 dataset.[4] This subset contains 288 documents and 3,391 sentence pairs. We will refer to this subset as *wa-subset*. This evaluation set is over 20 times larger than the 150 sentences set used in most past evaluations (DeNero and Klein, 2008; Haghighi et al., 2009; DeNero and Macherey, 2011).

Alignments input to the BI-NER model are produced by thresholding the averaged posterior probability at 0.5. In joint NER and alignment experiments, instead of posterior thresholding, we take the direct intersection of the Viterbi-best alignment of the two directional models. We report the standard P, R, F_1 and Alignment Error Rate (AER) measures for alignment experiments.

Joint NER and Alignment Results

We present results for the model that jointly performs word alignment and NER decoding (BI-NER-WA, *cf.* p.84) in Table 5.3. +NC means incorporating additional neighbor constraints from DeNero and Macherey (2011) to the model. The best number in each column is highlighted in bold.

By jointly decoding NER with word alignment, our model not only maintains significant improvements in NER performance, but also yields significant improvements to alignment performance. Joint decoding with NER alone yields a 10.8% error reduction in AER over the baseline HMM-aligners, and also gives improvement over BI-NER in NER. Adding additional neighborhood constraints gives a further 6% error reduction in AER, at the cost of a small loss in Chinese NER.

[4] LDC Catalogue No. LDC2006E86.

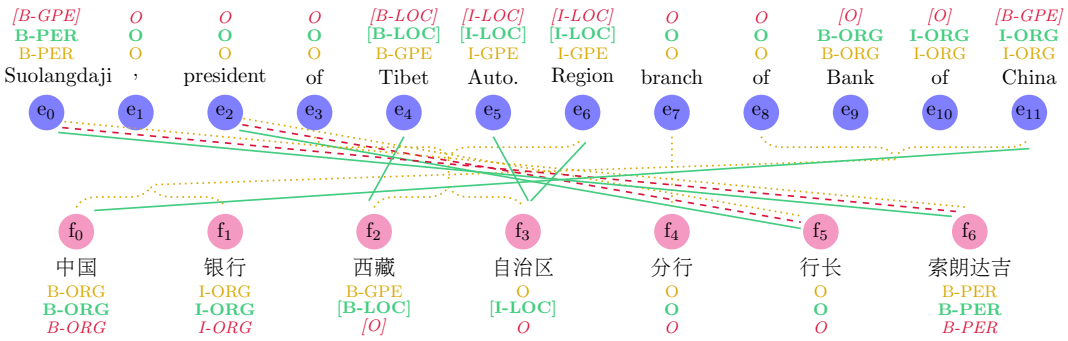


Figure 5.5: An example output of the joint word alignment and NER model.

Error Analysis and Discussion

We can examine the example in Figure 5.5 to gain an understanding of the model's performance. In this example, a snippet of a longer sentence pair is shown with NER and word alignment results. Gold dotted alignment links are the oracle, crimson dashed links are alignments from HMM baseline, and green solid links are outputs of our model. Entity tags in gold are the gold-standard tags; in crimson are baseline output, and in green are output from our model.

The monolingual Chinese tagger provides a strong cue that word f_6 is a person name because the unique 4-character word pattern is commonly associated with foreign names in Chinese, and also the word is immediately preceded by the word "president". The English monolingual tagger, however, confuses the aligned word e_0 with a GPE.

Our bilingual NER model is able to correct this error as expected. Similarly, the bilingual model corrects the error over e_{11}. However, the model also propagates labeling errors from the English side over the entity "Tibet Autonomous Region" to the Chinese side. Nevertheless, the resulting Chinese tags are arguably more useful than the original tags assigned by the baseline model.

In terms of word alignment, the HMM models failed badly on this example because of the long distance swapping phenomena. The two unidirectional HMMs also have strong disagreements over the alignments, and the resulting baseline aligner output only recovers two links. If we were to take this alignment as fixed input, most likely we would not be able to recover the error over e_{11}, but the joint decoding method successfully recovered 4

more links, and indirectly resulted in the NER tagging improvement discussed above.

5.3.3 Gibbs Sampling

Markov Chain Monte Carlo (MCMC) methods offer yet another simple and elegant solution for approximate inference by constructing a Markov chain whose stationary distribution is the target distribution. We adopt a specific MCMC sampling method called Gibbs sampling (Geman and Geman, 1984).

We define a Markov chain over output sequences by observing a simple transition rule: from a current sequence assignment at time $t-1$, we can transition into the next sequence at time t by changing the label at any position i. And the distribution over these transitions is defined as:

$$P(\boldsymbol{y}^t|\boldsymbol{y}^{t-1}) = P(y_i^t|\boldsymbol{y}_{-i}^{t-1}, \boldsymbol{x}) \tag{5.21}$$

where $\boldsymbol{y}_{-i}^{t-1}$ is the set of all labels except y_i at time $t-1$.

To apply the bilingual constraints during decoding, we formulate a new *factored* model by combining the two monolingual CRF models (one for each language) with the bilingual constraint model via a simple product.[5] The resulting model is of the following form:

$$P(\boldsymbol{y}^c, \boldsymbol{y}^e|\boldsymbol{x}^c, \boldsymbol{x}^e) = P_{CRF}(\boldsymbol{y}^c|\boldsymbol{x}^c)P_{CRF}(\boldsymbol{y}^e|\boldsymbol{x}^e)P_{bi}(\boldsymbol{y}^c, \boldsymbol{y}^e) \tag{5.22}$$

Obtaining the state transition model $P(y_i^t|\boldsymbol{y}_{-i}^{t-1}, \boldsymbol{x})$ for the monolingual CRF models is straight-forward. In the case of a first order linear-chain CRF, the Markov blanket is the neighboring two cliques. Given the Markov blanket of state i, the label at position i is independent of all other states. Thus we can compute the transition model simply by normalizing the product of the neighboring clique potentials. Finkel et al. (2005) gave a more detailed account of how to compute this quantity.

The transition probability of label ${y^c}_i$ in the bilingual constraint model is defined as (*cf.*

[5]This model double-counts the state sequence conditioned on a given observation, and therefore is likely deficient. However, we do not find this to be a problem in practice.

Eq. 5.8):

$$\prod_{a \in A} \lambda_a^{y^c y^e} = \prod_{a \in \mathcal{A}} (\lambda_a^{y^c y^e})^{I_a}$$

At decoding time, we walk the Markov chain by taking samples at each step. We start from some random assignment of the label sequence, and at each step we randomly sample a new value for y_i at a randomly chosen position i. After a fixed number of steps, we output a complete sequence as our final solution. In practice, MCMC sampling could be quite slow and inefficient, especially when the input sentence is long. To speed up the sampling process, we initialize the state sequence from the best sequences found by Viterbi decoding using only the monolingual models.

A bigger problem with vanilla Gibbs sampling is that the random samples we draw do not necessarily give us the most likely state sequence, as given by Viterbi in the exact inference case. One way to tackle this problem is to borrow the *simulated annealing* technique from optimization research (Kirkpatrick et al., 1983). We redefine the transition probability in Eq. 5.21 as:

$$P(\boldsymbol{y}^t | \boldsymbol{y}^{t-1}) = \frac{P(y_i^t | \boldsymbol{y}_{-i}^{t-1}, \boldsymbol{x})^{1/c_t}}{\sum_j P(y_j^t | \boldsymbol{y}_{-j}^{t-1}, \boldsymbol{x})^{1/c_t}} \tag{5.23}$$

where $\mathbf{c} = \{c_0 \dots c_T\}$ is the schedule of annealing "temperature," with $0 \leq c_i \leq 1$. The distribution becomes sharper as the value of c_i move towards 0. In our experiments we adopted a linear cooling schedule, where $c_0 = 1$, and $c_{t+1} = c_t - 1/T$. This technique has been shown to be effective by Finkel et al. (2005).

Speeding Up Sampling

Sampling based approaches often suffer from the problem of speed and accuracy tradeoff—taking more samples will converge to higher probabilities and thus improve model performance (as illustrated in Figure 5.6), but the more samples we draw the more time it takes. We employ a number of strategies to improve the efficiency of the sampling process.

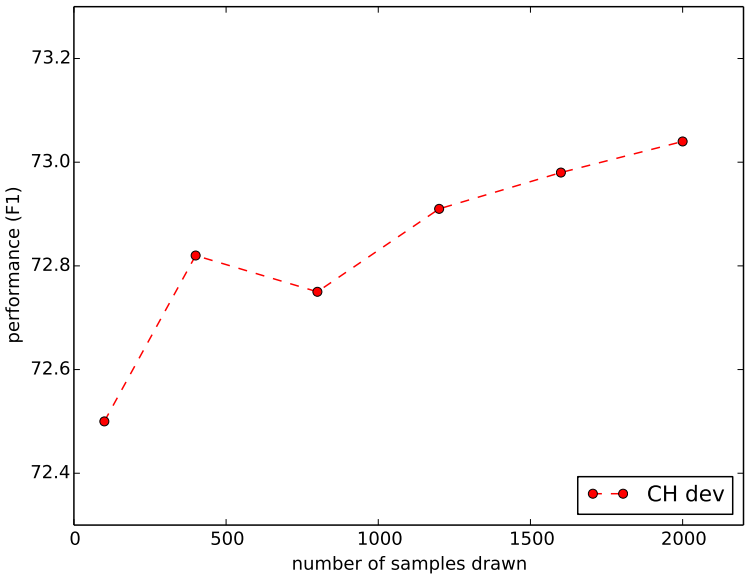


Figure 5.6: Performance on Chinese dev set by varying the number of samples

Parallel Sampling In a standard Gibbs sampling algorithm, we sequentially sample word labels in a sentence by picking one word position at a time. An avenue for improvement of efficiency is to explore parallelization strategies. Although naive parallelization strategies that sample multiple arbitrary word positions in parallel and mix the results tend to work reasonably well in practice, it does not guarantee the convergence of the Gibbs sampler to its stationary distribution (Gonzalez et al., 2011) since the conditional probabilities at each position are not necessarily independent. Gonzalez et al. (2011) proposed a more theoretically-sound parallelization strategy known as *chromatic sampling*. The basic idea is that we can first partition the graph (linear-chain in our case) into conditionally independent sub-graphs using a graph-coloring algorithm (such as the Welsh-Powell algorithm (Welsh and Powell, 1967)), then draw from nodes in each sub-graph in parallel. Since the Welsh-Powell algorithm is a greedy algorithm, a chromatic sampling strategy gives us a convergence guarantee at a negligible additional runtime cost. In our experiments, 92% of the examples are 4-colorable, and the rest are 5-colorable. Employing chromatic sampling with 16 parallel threads gives us a 14.2 times speedup.[6]

[6]The speedup is shy of 16 times due to computational overhead in multi-threading.

	Chinese			English			Speed
K	P	R	F_1	P	R	F_1	(second/doc)
100	79.60	67.08	72.80	80.75	73.70	77.06	**1.64**
200	79.76	67.18	72.93	80.78	73.74	77.10	2.85
300	79.70	67.15	72.89	80.79	73.75	77.11	3.82
400	79.84	67.33	73.05	80.75	73.70	77.06	4.98
500	79.93	67.31	**73.08**	80.78	73.80	**77.13**	5.99

Table 5.4: Speed and accuracy trade-off on the dev set using the position selection heuristic.

Position Selection The sampling process can be described in two steps: firstly we sample a position in the sentence, and secondly we sample a new label at the chosen position. One observation for the task of NER is that most words in a sentence are not entities, and those word positions tend to have very sharp local distributions and are not likely to have a label change within a small number of samples. Therefore, we devise a simple yet effective heuristic to take advantage of this observation. We monitor the word positions that are sampled in the first K iterations, and keep track of positions that have had a label change; then after K iteration, we only sample from word positions that have had a label change. This heuristic will encourage the sampler to explore higher likelihood states of word positions with more uncertainties. Table 5.4 lists the speed and accuracy tradeoff using this strategy. We can see that generally speaking, the larger the K, the higher the accuracy (with the exception of Chinese at $K = 300$ and English at $K = 400$). The amount of time it takes to process each document scales linearly with value K.

Gibbs Sampling Results

The main results on Chinese and English test sets are shown in Table 5.5. We take 1000 samples in the final experiments. The best F_1 scores in each column that are statistically significantly better than the CRF baseline is highlighted in bold. The first row (`CRF`) shows the baseline monolingual model performance. As we can see, the performance on Chinese is much lower than on English. This is partially attributed to the fact that the Chinese NER tagger was trained on less than half as much data, but it is also because NER in Chinese is a harder problem (e.g., there are no capitalization features in Chinese, which is a very strong

	Chinese			English		
	P	R	F_1	P	R	F_1
CRF	76.89	61.64	68.42	81.98	74.59	78.11
Burkett	77.52	65.84	71.20	82.28	76.64	79.36
ILP	77.56	71.10	74.19	80.93	76.65	78.73
DualDecomp	79.14	**71.55**	**75.15**	82.58	**77.96**	80.20
hard	76.19	64.47	69.84	82.13	72.85	77.21
manual	80.02	65.85	72.24	**82.87**	74.56	78.50
auto	78.53	66.90	72.25	82.11	75.40	78.62
auto+aP	79.17	68.46	73.43	82.05	75.56	78.67

Table 5.5: Gibbs sampling results on bilingual parallel test set. F_1 scores that are statistically significantly better than the CRF baseline is highlighted in bold. *+aP* means the model with alignment uncertainty, same as `soft-align` in Table 5.1.

indicator of named entities in English).

By enforcing hard agreement constraints, we can see from row `hard` that there is an increase of about 1.4% in absolute F_1 score on the Chinese side, but at the expense of a 0.9% drop on the English side. The tradeoff mainly occurs in recall.

When we loosen the bilingual constraint to allow soft-agreement by simply assigning a hand-picked value (0.02) to aligned entities of different types (row `manual`), we observe a significant increase in accuracy in both Chinese and English. This suggests that the soft alignment successfully accounted for the cases where annotation standards differ in the two languages. In particular, the Chinese results are 3.8% better than the monolingual baseline, a 12% relative error reduction.

When we replace the arbitrary hand-picked soft-agreement probabilities with empirical counts from the auto-tagged dataset (row `auto`), we see a small increase in recall on both sides, but a drop in precision for Chinese. However, accounting for alignment uncertainty (row `auto+aP`) increases both precision and recall for Chinese, resulting in another 1.2% increase in absolute F_1 score over the `auto` model.

Comparing against Burkett et al. (2010b) (second row from the top), we can see that both our method and Burkett et al. (2010b) significantly outperform the monolingual CRF baseline. This suggests that methods that exploring bilingual language cues does have great utility in the NER task. Our best model (`auto+aP`) gives a significant gain over Burkett

et al. (2010b) on Chinese (by 2.2%), but trails behind on English by 0.7%.

In comparison to the ILP and dual decomposition algorithms in the previous two subsections, Gibbs sampling base decoding does not seem to improve output accuracy. However, we will show in the next section some further improvements to our method by modeling global label consistency, which allow us to outperform Burkett et al. (2010b) on both languages, and give additional gains over the ILP method.

Global Consistency Constraints

A distinctive feature of the proposed factored model and Gibbs sampling inference is the ability to incorporate non-local constraints that are not easily captured in a traditional Markov network model. The bilingual constraint model described earlier is certainly a benefactor of this unique characteristic.

Still, there are further linguistic constraints that we can apply to improve the NER system. For example, many previous papers have made the observation that occurrences of the same word sequence within a given document are unlikely to take on different entity types (Bunescu and Mooney 2004; Sutton and McCallum 2004; Finkel et al. 2005; *inter alia*) . Similar to Finkel et al. (2005), we devise a global consistency model as follows:

$$P_{glo}(\boldsymbol{y}|\boldsymbol{x}) = \prod_{\gamma \in \Gamma} \phi_{\gamma}^{\#(\gamma, \boldsymbol{y}, \boldsymbol{x})} \tag{5.24}$$

Γ is the set of all possible entity type violations, ϕ_γ is the penalty parameter for violation type γ, and $\#(\gamma, \boldsymbol{y}, \boldsymbol{x})$ is the count of violations γ in sequence $\boldsymbol{y}$. For example, if the word sequence "China Daily" has occurred both as GPE and ORGANIZATION exactly once, then the penalty ϕ_γ for *GPE-to-Organization* violation will apply once. The parameter values of ϕ_γ are estimated empirically by counting the occurrences of entity pairs of the same word sequence in the training data.

We can now factor in one global consistency model for each language by taking the product of Eq. 5.22 with Eq. 5.24. The same Gibbs sampling procedure applies unchanged to this new factored model. At test time, instead of tagging one sentence at a time, we group together sentences that belong to the same document, and tag one document at a time.

	Chinese			English		
	P	R	F_1	P	R	F_1
mono	76.89	61.64	68.42	81.98	74.59	78.11
+*global*	77.30	58.96	66.90	83.89	74.88	79.13
+*global'*	75.23	68.12	71.50	82.31	77.63	79.90
auto	78.53	66.90	72.25	82.11	75.40	78.62
+*global*	79.02	64.57	71.07	**84.02**	75.73	79.66
+*global'*	76.17	71.04	73.52	82.87	**78.84**	**80.81**
auto+aP	79.17	68.46	73.43	82.05	75.56	78.67
+*global*	**79.31**	65.93	72.01	84.01	75.81	79.70
+*global'*	76.43	**72.32**	**74.32**	82.30	78.35	80.28

Table 5.6: Results of enforcing global consistency. *global* is the global consistency without "reward" parameter, and *global'* is the one with "reward" parameter. "mono" is the monolingual CRF baseline. The best number in each column is highlighted in bold. *+aP* means the model with alignment uncertainty, same as `soft-align` in Table 5.1.

Enhancing Recall A flaw of the Finkel et al. (2005) model described above is that consistency is enforced by applying penalties to entity type violations. But if a word is not tagged with an entity type, it will not receive any penalty since no entity type violations would occur. Therefore, this model has the tendency of favoring null annotations, which can result in losses in model recall.

We fix this deficiency in Finkel et al. (2005) by introducing a new "reward" parameter δ, which has value > 0. δ is activated each time we see a matching pair of entities for the same word occurrence. The new P_{glo} is modified as:

$$P_{glo}(\boldsymbol{y}|\boldsymbol{x}) = \delta^{\#(\delta,\boldsymbol{y},\boldsymbol{x})} \prod_{\gamma \in \Gamma} \phi_{\gamma}^{\#(\gamma,\boldsymbol{y},\boldsymbol{x})} \tag{5.25}$$

where $\#(\delta, \boldsymbol{y}, \boldsymbol{x})$ is the activation count of δ in sequence $\boldsymbol{y}$.

This model is in fact a naive Bayes model, where the parameters δ and ϕ are empirically estimated (a value of 2 is used for δ in our experiments, based on tuning on a development set). A similar global consistency model was shown to be effective in Rush et al. (2012), where parameters were also tuned on a development set.

Global Consistency Results Table 5.6 shows results on the test set after factoring in a global consistency model. *Global* is the global consistency without "reward" parameter, and *global'* is the one with "reward" parameter. "mono" is the monolingual CRF baseline. The best number in each column is highlighted in bold. Adding global consistency to the monolingual baseline (`mono`) increases performance on English (consistent with results from previous work (Finkel et al., 2005)), but hurts Chinese results, especially in recall.

A possible explanation is that CRF models for English are more certain about which words are entities (by having strong indicative features such as word capitalization), and thus a penalty does not persuade the model to label a word as a non-entity. However, in the Chinese case, the CRF model is weaker, and thus less certain about words being an entity or not. It is also much more likely that the same word (string) will be both an entity and a common word in Chinese than English. In some cases, the model will be better off marking a word as a non-entity, than risking taking a penalty for labeling it inconsistently. By applying the "reward" function, we see a drastic increase in recall on both Chinese and English, with a relatively small sacrifice in precision on Chinese. The overall F_1 scores increase by about 3.1% in Chinese and 0.8% in English.

Similar results can be found when we apply global consistency to the bilingual model (`auto`). Again we see a recall-precision tradeoff between models with or without a "reward" function. But overall, we observe a significant increase in performance when global consistency with a reward function is factored in.

Modeling alignment uncertainty continues to improve the Chinese results when the global consistency model is added, but shows a small performance decrease on the English side. But the gain on the Chinese side is more significant than the loss on English side.

The best overall F_1 scores are achieved when bilingual constraints, global consistency with reward, and alignment uncertainty are conjoined. The combined model outperforms the CRF monolingual baseline, with an error reduction of 18.6% for Chinese and 9.9% for English. This model also significantly improves over the method of Burkett et al. (2010b) with an error reduction of 10.8% for Chinese and 4.5% for English.

	Chinese			English		
	P	R	F_1	P	R	F_1
CRF	76.89	61.64	68.42	81.98	74.59	78.11
Burkett	77.52	65.84	71.20	82.28	76.64	79.36
ILP	77.56	71.10	74.19	80.93	76.65	78.73
Gibbs	76.43	**72.32**	74.32	82.30	**78.35**	**80.28**
DualDecomp	**79.14**	71.55	**75.15**	**82.58**	77.96	80.20

Table 5.7: Results summary of bilingual tagging experiment.

5.4 Results

5.4.1 Summary of Bilingual Results

We presented detailed experimental results for each of the three approximate inference algorithms earlier as we described them. In summary, the bilingual NER model with all three approximate inference methods gives significant improvements over both Chinese and English CRF baselines. The new model also yields significant gains over previous work by Burkett et al. (2010b).

The dual decomposition algorithm gives the highest overall F_1 score among the three inference methods. However, there is actually a precision-recall tradeoff when we compare dual decomposition with Gibbs sampling and ILP. In our error analysis, we also found that the dual decomposition algorithm assigns more non-entity background tokens than the other two inference methods. This could be a result of the different mixing strategies of dual decomposition and Gibbs sampling. For both dual decomposition and Gibbs sampling, we run the algorithms for 1000 iterations; in the dual decomposition case, in each iteration all alignment-based sub-components exchange messages by checking agreement and trading penalties; whereas in the case of Gibbs sampling, only one position is sampled in each iterations. Therefore it is more likely the case that the dual decomposition algorithm output is more sufficiently mixed than Gibbs sampling algorithm. So far we rely on testing using development data for tuning these hyper-parameters. Automatically learning hyper-parameters remain a quest for future work.

	Chinese			German		
	P	R	F_1	P	R	F_1
CRF	79.09	63.59	70.50	**86.69**	71.30	78.25
Burkett	79.25	65.67	71.83	84.00	72.17	77.64
ILP	**81.31**	65.50	**72.55**	85.99	**72.98**	**78.95**
Gibbs	80.31	65.78	72.33	85.98	72.37	78.59
DualDecomp	78.55	**66.54**	72.05	85.19	**72.98**	78.62

Table 5.8: Semi-supervised results using uptraining.

5.4.2 Semi-supervised NER Results

In the previous section we demonstrated the utility of our proposed method in a bilingual setting, where parallel sentence pairs are tagged together and directly evaluated. In reality, this is not the common use case. Most downstream NLP applications operate in a monolingual environment. Therefore, in order to benefit general monolingual NLP systems, we propose a semi-supervised learning setting where we use the bilingual tagger to annotate a large amount of unannotated bilingual text, then we take the tagged sentences on the Chinese side to retrain a monolingual Chinese tagger.

To evaluate the effectiveness of this approach, we randomly sample 80,000 sentences from the Chinese-English part of the Foreign Broadcast Information Service corpus (FBIS, LDC2003E14), and tagged it with the bilingual NER model using each of the three approximate inference algorithms. Unlike the OntoNotes dataset, this corpus does not contain document boundaries. In order to apply the document-level label consistency model, we divide the test set into blocks of ten sentences, and use the blocks as pseudo-documents.

Results from uptraining using model outputs from Burkett et al. (2010b) (`Burkett`) as well as bilingual NER models with all three approximate inference methods are shown in Table 5.8. As we noted earlier when we described the experiment setup in Section 2.5.3, for the semi-supervised experiments, we use individual sentences as input instead of whole documents. Because sentences are shorter than whole documents, they are easier to handle for sequence models, and as a result model performances across the board are higher than ones in Table 5.7.

We can see that by using only 80,000 additional sentences, our best method (`ILP`)

gives a significant boost ($\sim$2.05%, an error reduction of $\sim$7%) over the CRF baseline. Our method also improves over Burkett et al. (2010b) by a significant margin.

The gains are more pronounced in recall than precision, which suggests that the semi-supervised approach using bilingual data is very effective in increasing the coverage of the monolingual tagger.

Overall the performance difference among the three approximate inference algorithms is not significantly different. However, we do notice that the method that performs the best on the bilingual tagging experiment (`DualDecomp`) does not produce the same advantage in the semi-supervised learning experiments. In fact, in the bilingual tagging experimental results, dual decomposition algorithm gives the highest precision, and lowest recall among all three decoding methods, but when translated into uptraining results, it actually gives the reverse—lowest precision and highest recall. This discrepancy is likely to be due to the inconsistency innate to the "uptraining" procedure. Similar to self-training (McClosky et al., 2006), uptraining also suffers from instability problems. Later in Chapter 6 we will present a more principled method for performing semi-supervised learning based on expectation projection, which gives consistent improvements over uptraining methods.

5.4.3 Efficiency

A key design decision of which approximate inference method to employ is their speed-accuracy tradeoff. We saw earlier from the Table 5.8 that among the three approximate inference methods, `ILP` gives the highest accuracy in the semi-supervised learning setting. Table 5.9 shows the amount of time it took to decode the 80,000 sentence corpus and retrain the CRF model using uptraining. We can see that both the Gibbs sampling and dual decomposition methods give a significant speedup over the ILP method. Although dual decomposition sacrifices 0.5% accuracy in comparison to Gibbs sampling, it enjoys a three-times speed advantage over Gibbs sampling. These two methods are also significantly faster and more accurate than (Burkett et al., 2010b).

	Chinese	German
CRF	00h19m30s	00h07m15s
Burkett	06h16m21s	02h42m17s
ILP	16h42m07s	04h49m02s
Gibbs	03h17m25s	01h01m45s
DualDecomp	00h54m34s	00h22m43s

Table 5.9: Timing stats of the sum of decoding and model uptraining time.

5.5 Related Work

The idea of employing bilingual resources to improve over monolingual systems has been explored by much previous work. Huang et al. (2009) proposed methods to improve parsing performance using bilingual parallel corpus. Li et al. (2012a) jointly labeled bilingual named entities with a cyclic CRF model, where approximate inference was done using loopy belief propagation. These methods require manually annotated bilingual corpora, which are expensive to construct, and hard to obtain. Kim et al. (2012) proposed a method of labeling bilingual corpora with named entity labels automatically based on Wikipedia. However, this method is restricted to Wikipedia domain only.

One of the most interesting papers in this regard is Burkett et al. (2010b), which explored an "up-training" mechanism by using the outputs from a strong monolingual model as ground-truth, and simulated a learning environment where a bilingual model is trained to help a "weakened" monolingual model to recover the results of the strong model. Noisy output of a "strong" tagger is used as training data to learn parameters of a log-linear re-ranking model with additional bilingual features, simulated by the "weakened" tagger. The learned parameters are then reused with the "strong" tagger to re-rank its own outputs for unseen inputs. This framework can be seen as a "multi-view" learning scheme where k-best outputs of two monolingual taggers are reranked. Designing good "weak" taggers so that they complement the "view" of bilingual features in the log-linear re-ranker is crucial to the success of this algorithm. But the reranking method involves rather complicated bootstrapping and self-training, and is difficult to employ in practice. It is worth mentioning that since the methods we are going to propose here do not require additional training and can take pretty much any existing model as "black-box" during decoding, the richer and

more accurate bilingual model learned from Burkett et al. (2010b) can be directly plugged into our method.

In addition to bilingual corpora, bilingual dictionaries are also useful resources. Huang and Vogel (2002) and Chen et al. (2010) proposed approaches for extracting bilingual named entity pairs from unlabeled bilingual corpus, in which verification is based on bilingual named entity dictionaries. However, large-scale bilingual named entity dictionaries are difficult to obtain for most language pairs.

Yarowsky and Ngai (2001) proposed a projection method that transforms high-quality analysis results of one language, such as English, into other languages on the basis of word alignment. Das and Petrov (2011) applied the above idea to part-of-speech tagging with a more complex model. Fu et al. (2011) projected English named entities into Chinese entities by carefully designed heuristic rules. Although this type of method does not require manually annotated bilingual corpora or dictionaries, errors in the source language results, wrong word alignments and inconsistency between two languages limit the application of this method.

Constraint-based monolingual methods by using ILP have been successfully applied to many natural language processing tasks, such as Semantic Role Labeling (Punyakanok et al., 2004), Dependency Parsing (Martins et al., 2009) and Textual Entailment (Berant et al., 2011). Zhuang and Zong (2010) proposed a joint inference method for bilingual semantic role labeling with ILP. However, their approach requires training an alignment model with a manually annotated corpus.

A similar dual decomposition algorithm to ours was proposed by Riedel and McCallum (2011) for biomedical event detection. In their Model 3, the trigger and argument extraction models are reminiscent of the two monolingual CRFs in our model; additional binding agreements are enforced over every protein pair, similar to how we enforce agreement between every aligned word pair. Martins et al. (2010) presented a new dual decomposition method that combines the power of dual decomposition with the augmented Lagrangian method. They showed that their method can achieve faster convergence than traditional sub-gradient methods in models with many overlapping components (Martins et al., 2011). This method is directly applicable to our work.

Another promising direction for improving NER performance is in enforcing global

label consistency across documents, which is an idea that has been greatly explored in the past (Sutton and McCallum, 2004; Bunescu and Mooney, 2004; Finkel et al., 2005). More recently, Rush et al. (2012) and Chieu and Teow (2012) have shown that combining local prediction models with global consistency models, and enforcing agreement via dual decomposition is very effective. It is straightforward to incorporate an additional global consistency model into our model for further improvements.

Our joint alignment and NER decoding approach is inspired by prior work on improving alignment quality through encouraging agreement between bi-directional models (Liang et al., 2006; DeNero and Macherey, 2011). Instead of enforcing agreement in the alignment space based on best sequences found by Viterbi, we could opt to encourage agreement between posterior probability distributions, which is related to the posterior regularization work by Graça et al. (2008). Cromières and Kurohashi (2009) proposed an approach that takes phrasal bracketing constraints from parsing outputs, and uses them to enforce phrasal alignments. This idea is similar to our joint alignment and NER approach, but in our case the phrasal constraints are indirectly imposed by entity spans. We also differ in the implementation details, where in their case belief propagation is used in both training and Viterbi inference.

Burkett et al. (2010a) presented a supervised learning method for performing joint parsing and word alignment using log-linear models over parse trees and an ITG model over alignment. The model demonstrates performance improvements in both parsing and alignment, but shares the common limitations of other supervised work in that it requires manually annotated bilingual joint parsing and word alignment data.

Chen et al. (2010) also tackled the problem of joint alignment and NER. Their method employs a set of heuristic rules to expand a candidate named entity set generated by monolingual taggers, and then rank those candidates using a bilingual named entity dictionary. Our approach differs in that we provide a probabilistic formulation of the problem and do not require pre-existing NE dictionaries.

5.6 Summary

We introduced a factored bilingual model with three approximate inference algorithms, that can be used to produce more accurate tagging results for a parallel corpus. Our model makes use of cross-language bilingual constraints and intra-document consistency constraints. We further demonstrated that unlabeled parallel corpora tagged with our bilingual model can then be used to improve monolingual tagging results, using an uptraining scheme. Among the three inference algorithms, dual decomposition method achieves the fastest speed while Gibbs sampling method yields the highest accuracy improvement under the semi-supervised setting. At a higher level, the methods and results presented in this chapter demonstrated bitext as a useful alternative source of unlabeled data for improving supervised system performance. Our approach also provides computationally feasible approaches in dealing with complex factored graphical model and constrained inference problem. This approach can be generalized to more multilingual learning scenarios to handle many more languages, a lot of which have limited training resources.

Chapter 6

Projected Expectation Regularization

6.1 Introduction

In the previous chapter (Chapter 5), we proposed a semi-supervised learning scheme that takes advantage of information contained in translational bitext for improving monolingual NER systems. The main idea there was to jointly decode two sentences that are translations of each other, and enforce bilingual agreement constraints across alignment links between the language pair. Tagging results using this approach are significantly better than independently decoding each language alone. And then we showed that taking the decoded bitext as additional training data in an uptraining (Petrov et al., 2010) framework to retrain monolingual taggers can lead to significant improvements in monolingual tagging performance as well.

A major drawback of that approach, however, is that it requires access to a readily-trained tagging model in in both languages. This shortcoming severely prevents the application of the approach, since for the majority of the world's languages that we are interested in, there exists no annotated training resources for NER.

It is our goal to devise a method that would allow us to recognize entities in a minority language without having access to any annotation data. The method we will describe in this chapter (called **CLiPER**, stands for **C**ross-**Li**ngual **P**rojection-based **E**xpectation **R**egularization) is designed specifically for this goal.

The idea behind CLiPER is to project linguistic knowledge from a resource-rich language such as English, to a resource poor language. The projection is based on word-aligned bilingual parallel text, which is publicly available for many languages thanks to the recent advance in Machine Translation research.

Suppose we have a readily available NER tagger in English, one approach is to first perform NER on the English side of the bitext, then for every aligned pair of words, we assign the English label to the word in the "foreign" language. After the projection step, we can now harvest "tagged" documents on the foreign side, and use the projected labels as ground-truth to train a NER tagger for the foreign language, and use it in the future to tag unseen monolingual text. This is indeed the same procedure as some of the earlier seminal work on projection-based knowledge transfer (Yarowsky and Ngai, 2001).

The overall workflow of CLiPER is similar, but differ in two important details. Firstly, instead of projecting the actual label assignment from the English side to the foreign side, CLiPER packs up uncertainties in the English model into model expectations, and projects expectations instead. In other words, when the English model is not too certain about a decision, e.g., both PERSON and LOCATION tags gets probability 0.5, we do not have to force the foreign side model to commit to a best outcome; instead, we ask the foreign model to also assign a similar probability distribution to the two most-likely outcomes. It can be thought of as the "soft" version of Yarowsky-style projection.

The second aspect in which CLiPER differs from the traditional approach is that when learning the foreign model, it maximizes a different objective from the standard maximum-likelihood objective in supervised CRF. The new objective is based on the Generalized Expectation criteria (GE) framework by Mann and McCallum (2010).

We evaluate our approach on NER tasks for English-Chinese and English-German language pairs on standard public datasets. We report results in two settings: a weakly supervised setting where no labeled data or a small amount of labeled data is available, and semi-supervised settings where labeled data is available, but we can gain predictive power by learning from unlabeled bitext.

6.2 Related Work

Supervised statistical learning methods have enjoyed great popularity in Natural Language Processing (NLP) over the past decade. The success of supervised methods depends heavily upon the availability of large amounts of annotated training data. Manual curation of annotated corpora is a costly and time consuming process. To date, most annotated resources resides within the English language, which hinders the adoption of supervised learning methods in many multilingual environments.

To minimize the need for annotation, significant progress has been made in developing unsupervised and semi-supervised approaches to NLP (Collins and Singer 1999; Klein 2005; Liang 2005; Smith 2006; Goldberg 2010; *inter alia*).

Most semi-supervised learning approaches embody the principle of learning from constraints. There are two broad categories of constraints: multi-view constraints, and external knowledge constraints.

Examples of methods that explore multi-view constraints include self-training (Yarowsky, 1995; McClosky et al., 2006),[1] co-training (Blum and Mitchell, 1998; Sindhwani et al., 2005), multi-view learning (Ando and Zhang, 2005b; Carlson et al., 2010), and discriminative and generative model combination (Suzuki and Isozaki, 2008; Druck and McCallum, 2010).

An early example of using knowledge as constraints in weakly-supervised learning is the work by Collins and Singer (1999). They showed that the addition of a small set of "seed" rules greatly improves a co-training style unsupervised tagger. Chang et al. (2007) proposed a constraint-driven learning (CODL) framework where constraints are used to guide the selection of best self-labeled examples to be included as additional training data in an iterative EM-style procedure. The kind of constraints used in applications such as NER are the ones like "the words CA, Australia, NY are LOCATION" (Chang et al., 2007). Notice the similarity of this particular constraint to the kinds of features one would expect to see in a discriminative model such as MaxEnt. The difference is that instead of learning the validity (or weight) of this feature from labeled examples—since we do not have them—we can constrain the model using our knowledge of the domain. Druck et al. (2009) also

[1] A multi-view interpretation of self-training is that the self-tagged additional data offers new views to learners trained on existing labeled data.

demonstrated that in an active learning setting where annotation budget is limited, it is more efficient to label features than examples. Other sources of knowledge include lexicons and gazetteers (Druck et al., 2007; Chang et al., 2007).

While it is straight-forward to see how resources such as a list of city names can give a lot of mileage in recognizing locations, we are also exposed to the danger of over-committing to hard constraints. For example, it becomes problematic with city names that are ambiguous, such as Augusta, Georgia.[2] To soften these constraints, Mann and McCallum (2010) proposed the Generalized Expectation (GE) Criteria framework, which encodes constraints as a regularization term over some score function that measures the divergence between the model's expectation and the target expectation. The connection between GE and CODL is analogous to the relationship between hard (Viterbi) EM and soft EM, as illustrated by Samdani et al. (2012).

Another closely related work is the Posterior Regularization (PR) framework by Ganchev et al. (2010). In fact, as Bellare et al. (2009) have shown, in a discriminative model these two methods optimize exactly the same objective.[3] The two differ in optimization details: PR uses a EM algorithm to approximate the gradients which avoids the expensive computation of a covariance matrix between features and constraints, whereas GE directly calculates the gradient. However, later results (Druck, 2011) have shown that using the Expectation Semiring techniques of Li and Eisner (2009), one can compute the exact gradients of GE in a Conditional Random Field (CRF) (Lafferty et al., 2001) at cost no greater than computing the gradients of ordinary CRF. And empirically, GE tends to perform more accurately than PR (Bellare et al., 2009; Druck, 2011).

Obtaining appropriate knowledge resources for constructing constraints remain as a bottleneck in applying GE and PR to new languages. However, a number of past work recognizes parallel bitext as a rich source of linguistic constraints, naturally captured in the translations. As a result, bitext has been effectively utilized for unsupervised multilingual grammar induction (Alshawi et al., 2000; Snyder et al., 2009), parsing (Burkett and Klein, 2008), and sequence labeling (Naseem et al., 2009).

[2]This is a city in the state of Georgia in USA, famous for its golf courses. It is ambiguous since both Augusta and Georgia can also be used as person names.

[3]The different terminology employed by GE and PR may be confusing to discerning readers, but the "expectation" in the context of GE means the same thing as "marginal posterior" as in PR.

Projection-based methods can be very effective in weakly-supervised scenarios, as demonstrated by Yarowsky and Ngai (2001), and Xi and Hwa (2005). One problem with projected labels is that they are often too noisy to be directly used as training signals. To mitigate this problem, Das and Petrov (2011) designed a label propagation method to automatically induce a tag lexicon for the foreign language to smooth the projected labels. Fossum and Abney (2005) filter out projection noise by combining projections from from multiple source languages. However, this approach is not always viable since it relies on having parallel bitext from multiple source languages. Li et al. (2012b) proposed the use of crowd-sourced Wiktionary as additional resources for inducing tag lexicons. More recently, Täckström et al. (2013) combined token-level and type-level constraints to constrain legitimate label sequences and and recalibrate the probability distribution in a CRF. The tag dictionary used for POS tagging is analogous to the gazetteers and name lexicons used for NER by Chang et al. (2007).

Our work is also closely related to Ganchev et al. (2009). They used a two-step projection method similar to Das and Petrov (2011) for dependency parsing. Instead of using the projected linguistic structures as ground truth (Yarowsky and Ngai, 2001), or as features in a generative model (Das and Petrov, 2011), they used them as constraints in a PR framework. Our work differs by projecting expectations rather than Viterbi one-best labels. We also choose the GE framework over PR. Experiments in Bellare et al. (2009) and Druck (2011) suggest that in a discriminative model (like ours), GE is more accurate than PR. More recently, Ganchev and Das (2013) further extended this line of work to directly train discriminative sequence models using cross lingual projection with PR. The types of constraints applied in this new work are similar to the ones in the monolingual PR setting proposed by Ganchev et al. (2010), where the total counts of labels of a particular kind are expected to match some fraction of the projected total counts. Our work differs in that we enforce expectation constraints at the token level, which gives tighter guidance in learning the model.

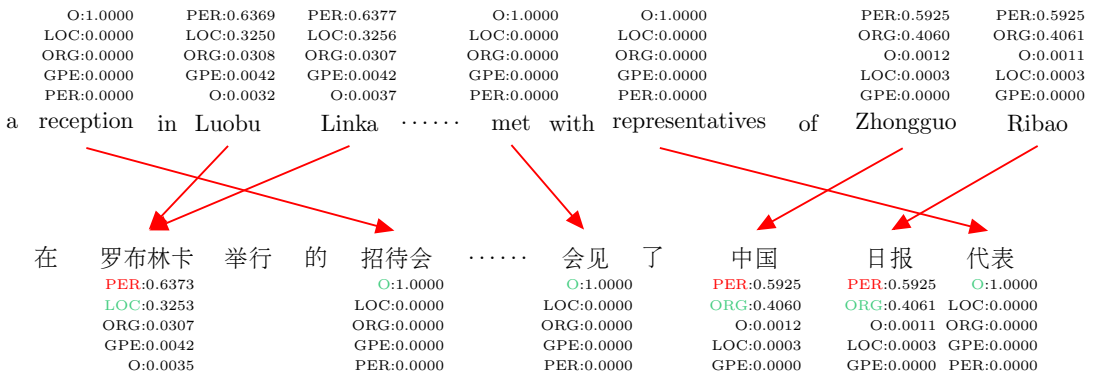


Figure 6.1: Diagram illustrating the projection of model expectation from English to Chinese.

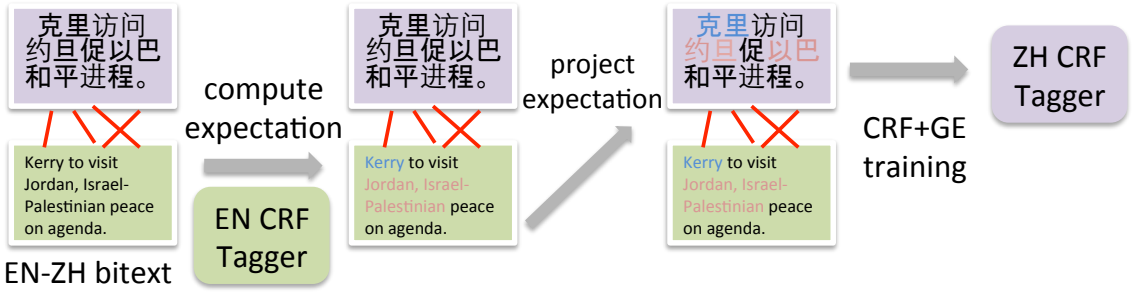


Figure 6.2: Diagram illustrating the workflow of CLiPER method.

6.3 Approach

Given bitext between English and a foreign language, our goal is to learn a CRF model in the foreign language from little or no labeled data. Our method performs cross-lingual projection-based expectation regularization.

Figure 6.2 illustrates the high-level workflow. Colors over the bitext in this diagram are intended to denote model expectations, not the actual label assignments. Figure 6.1 shows a snippet of a sentence from a real corpus, the posterior probabilities assigned by the English CRF model are shown above each English word; automatically induced word alignments are shown in red; the correct projected labels for Chinese words are shown in green, and incorrect labels are shown in red.

For every aligned sentence pair in the bitext, we first compute the posterior marginal at each word position on the English side using a pre-trained English CRF tagger; then for

each aligned English word, we project its posterior marginal as expectations to the aligned word position on the foreign side.

We would like to learn a CRF model in the foreign language that has similar expectations as the projected expectations from English. To this end, we adopt the Generalized Expectation (GE) Criteria framework introduced by Mann and McCallum (2010). In the remainder of this section, we follow the notation used in (Druck, 2011) to explain our approach.

6.3.1 CLiPER

The general idea of GE is that we can express our preferences over models through constraint functions. A desired model should satisfy the imposed constraints by matching the expectations on these constraint functions with some target expectations (attained by external knowledge like lexicons or in our case transferred knowledge from English). We define a constraint function $\boldsymbol{\phi}_{i,l_j}$ for each word position i and output label assignment l_j as a label identity indicator:

$$\phi_{i,l_j}(\mathbf{y}) = \begin{cases} 1 & \text{if } l_j = y_i \text{ and } A_i \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

The set $\{l_1, \cdots, l_m\}$ denotes all possible label assignment for each y_i, and m is number of label values. A_i is the set of English words aligned to Chinese word i. The condition $A_i \neq \emptyset$ specifies that the constraint function applies only to Chinese word positions that have at least one aligned English word. Each $\phi_{i,l_j}(\mathbf{y})$ can be treated as a Bernoulli random variable, and we concatenate the set of all ϕ_{i,l_j} into a *random vector* $\boldsymbol{\phi}(\mathbf{y})$, where $\boldsymbol{\phi}_k = \phi_{i,l_j}$ if $k = i \times m + j$. We drop the $(\mathbf{y})$ in $\boldsymbol{\phi}$ for simplicity.

The target expectation over ϕ_{i,l_j}, denoted as $\tilde{\phi}_{i,l_j}$, is the expectation of assigning label l_j to English word A_i[4] under the English conditional probability model.

The expectation over $\boldsymbol{\phi}$ under a conditional probability model $P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta})$ is denoted as $E_{P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta})}[\boldsymbol{\phi}]$, and simplified as $E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]$ whenever it is unambiguous.

The conditional probability model $P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta})$ in our case is a standard linear-chain CRF

[4] An English word aligned to foreign word at position i. When multiple English words are aligned to the same foreign word, we average the expectations.

defined in Eq. 2.1:

$$P_{CRF}(\mathbf{y}|\mathbf{x}) = \frac{\prod_{i=1}^{|\mathbf{x}|} \exp\{\boldsymbol{f}(y_i, y_{i-1}, \mathbf{x}) \cdot \boldsymbol{\theta}\}}{Z(\mathbf{x})} \tag{6.1}$$

where $\boldsymbol{f}$ is a set of feature functions; $\boldsymbol{\theta}$ are the matching parameters to learn.

The objective function to maximize in a standard CRF is the log probability over a collection of labeled documents:

$$L_{CRF}(\boldsymbol{\theta}) = \sum_{a=1}^{a'} \log P_{CRF}(\mathbf{y}_a^*|\mathbf{x}_a; \boldsymbol{\theta}) \tag{6.2}$$

a' is the number of labeled sentences. $\mathbf{y}^*$ is an observed label sequence.

The objective function to maximize in GE is defined as the sum over all unlabeled examples (foreign side of bitext), over some cost function S between between the model expectation ($E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]$) and the target expectation ($\tilde{\boldsymbol{\phi}}$) over $\boldsymbol{\phi}$.

We choose S to be the negative L_2^2 squared error,[5] defined as:

$$\begin{aligned} L_{GE}(\boldsymbol{\theta}) &= \sum_{b=1}^{n'} S\left(E_{P(\mathbf{y}|\mathbf{x}_b;\boldsymbol{\theta})}[\boldsymbol{\phi}(\mathbf{y}_b)], \tilde{\boldsymbol{\phi}}(\mathbf{y}_b)\right) \\ &= \sum_{b=1}^{b'} -\|\tilde{\boldsymbol{\phi}}(\mathbf{y}_b) - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}(\mathbf{y}_b)]\|_2^2 \end{aligned} \tag{6.3}$$

b' is the total number of unlabeled bitext sentence pairs.

When both labeled and bitext training data are available, the joint objective is the sum of Eq. 6.2 and 6.3. Each is computed over the labeled training data and foreign half in the bitext, respectively.

We can optimize this joint objective by computing the gradients and use a gradient-based optimization method such as L-BFGS. Gradients of L_{CRF} decomposes down to the

[5] In general, other loss functions such as KL-divergence can also be used for S. We found L_2^2 to work well in practice.

gradients over each labeled training example $(\mathbf{x}, \mathbf{y}^*)$, computed as:

$$\frac{\partial}{\partial \boldsymbol{\theta}}(\log P(\mathbf{y}_a^*|\mathbf{x}_a; \boldsymbol{\theta}) = \tilde{E}[\boldsymbol{\theta}] - E[\boldsymbol{\theta}]$$

where $\tilde{E}[\boldsymbol{\theta}]$ and $E[\boldsymbol{\theta}]$ are the empirical and expected feature counts, respectively.

Computing the gradient of L_{GE} decomposes down to the gradients of $S(E_{P(\mathbf{y}|\mathbf{x}_b;\boldsymbol{\theta}}[\boldsymbol{\phi}])$ for each unlabeled foreign sentence $\mathbf{x}$ and the constraints over this example $\boldsymbol{\phi}$. The gradients can be calculated as:

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\theta}} S(E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]) &= -\frac{\partial}{\partial \boldsymbol{\theta}} \left(\tilde{\boldsymbol{\phi}} - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]\right)^T \left(\tilde{\boldsymbol{\phi}} - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]\right) \\ &= 2\left(\tilde{\boldsymbol{\phi}} - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]\right)^T \left(\frac{\partial}{\partial \boldsymbol{\theta}} E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]\right)\end{aligned}$$

We redefine the penalty vector $\mathbf{u} = 2\left(\tilde{\boldsymbol{\phi}} - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]\right)$ to be u. $\frac{\partial}{\partial \boldsymbol{\theta}} E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]$ is a matrix where each column contains the gradients for a particular model feature θ with respect to all constraint functions $\boldsymbol{\phi}$. It can be computed as:

$$\begin{aligned}&\frac{\partial}{\partial \boldsymbol{\theta}} E_{\boldsymbol{\theta}}[\boldsymbol{\phi}] = \sum_{\mathbf{y}} \boldsymbol{\phi}(\mathbf{y}) \frac{\partial}{\partial \boldsymbol{\theta}} P(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta}) \\ &= \sum_{\mathbf{y}} \boldsymbol{\phi}(\mathbf{y}) \frac{\partial}{\partial \boldsymbol{\theta}} \left(\frac{1}{Z(\mathbf{x})} \exp(\boldsymbol{\theta}^T \boldsymbol{f}(\mathbf{x}, \mathbf{y}))\right) \\ &= \sum_{\mathbf{y}} \boldsymbol{\phi}(\mathbf{y}) \left(\frac{1}{Z(\mathbf{x})} \left(\frac{\partial}{\partial \boldsymbol{\theta}} \exp(\boldsymbol{\theta}^T \boldsymbol{f}(\mathbf{x}, \mathbf{y}))\right)\right. \\ &\quad \left. + \exp(\boldsymbol{\theta}^T \boldsymbol{f}(\mathbf{x}, \mathbf{y})) \left(\frac{\partial}{\partial \boldsymbol{\theta}} \frac{1}{Z(\mathbf{x})}\right)\right)\end{aligned}$$

$$
\begin{aligned}
&= \sum_{\mathbf{y}} \boldsymbol{\phi}(\mathbf{y}) \Big(P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}) \boldsymbol{f}(\mathbf{x},\mathbf{y})^T - P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}) \sum_{\mathbf{y}'} P(\mathbf{y}'|\mathbf{x};\boldsymbol{\theta}) \boldsymbol{f}(\mathbf{x},\mathbf{y}')^T \Big) \\
&= \sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}) \sum_{\mathbf{y}} \boldsymbol{\phi}(\mathbf{y}) \boldsymbol{f}(\mathbf{x},\mathbf{y})^T - \big(\sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}) \boldsymbol{\phi}(\mathbf{y}) \big) \big(\sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}) \boldsymbol{f}(\mathbf{x},\mathbf{y})^T \big) \\
&= \mathrm{COV}_{P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta})} \left(\boldsymbol{\phi}(\mathbf{y}), \boldsymbol{f}(\mathbf{x},\mathbf{y}) \right) \qquad (6.4) \\
&= E_{\boldsymbol{\theta}}[\boldsymbol{\phi}\boldsymbol{f}^T] - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}] E_{\boldsymbol{\theta}}[\boldsymbol{f}^T] \qquad (6.5)
\end{aligned}
$$

Eq. 6.4 gives the intuition of how optimization works in GE. In each iteration of L-BFGS, the model parameters are updated according to their covariance with the constraint features, scaled by the difference between current expectation and target expectation. The term $E_{\boldsymbol{\theta}}[\boldsymbol{\phi}\boldsymbol{f}^T]$ in Eq. 6.5 can be computed using a dynamic programming (DP) algorithm, but solving it directly requires us to store a matrix of the same dimension as $\boldsymbol{f}^T$ in each step of the DP. We can reduce the complexity by using the following trick:

$$
\begin{aligned}
&\frac{\partial}{\partial \boldsymbol{\theta}} S(E_{\boldsymbol{\theta}}[\boldsymbol{\phi}]) = u^T \left(\frac{\partial}{\partial \boldsymbol{\theta}} E_{\boldsymbol{\theta}}[\boldsymbol{\phi}] \right) \\
=&\mathbf{u}^T \left(E_{\boldsymbol{\theta}}[\boldsymbol{\phi}\boldsymbol{f}^T] - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}] E_{\boldsymbol{\theta}}[\boldsymbol{f}^T] \right) \\
=&E_{\boldsymbol{\theta}}[\mathbf{u}^T\boldsymbol{\phi}\boldsymbol{f}^T] - E_{\boldsymbol{\theta}}[\mathbf{u}^T\boldsymbol{\phi}] E_{\boldsymbol{\theta}}[\boldsymbol{f}^T] \\
=&E_{\boldsymbol{\theta}}[\boldsymbol{\phi}'\boldsymbol{f}^T] - E_{\boldsymbol{\theta}}[\boldsymbol{\phi}'] E_{\boldsymbol{\theta}}[\boldsymbol{f}^T] \qquad (6.6) \\
&\boldsymbol{\phi}' = \mathbf{u}^T\boldsymbol{\phi}
\end{aligned}
$$

Now in Eq. 6.6, $E_{\boldsymbol{\theta}}[\boldsymbol{\phi}']$ becomes a scalar value; and to compute the term $E_{\boldsymbol{\theta}}[\boldsymbol{\phi}'\boldsymbol{f}^T]$, we only need to store a vector in each step of the following DP algorithm (Druck, 2011, 93):

$$
E_{\boldsymbol{\theta}}[\boldsymbol{\phi}'\boldsymbol{f}^T] = \sum_{i=1}^{n} \sum_{y_i} \sum_{y_{i+1}} \left\{ \Big[\sum_{j=1}^{n} \sum_{y_j} P(y_i, y_{i+1}, y_j|\mathbf{x}) \cdot \boldsymbol{\phi}(y_j) \Big] \cdot \boldsymbol{f}(y_i, y_{i+1}, \mathbf{x})^T \right\}
$$

The bracketed term can be broken down to two parts:

$$\begin{aligned}
&\sum_{j=1}^{n}\sum_{y_j} P(y_i, y_{i+1}, y_j|\mathbf{x})\boldsymbol{\phi}(y_j) \\
=&\sum_{j=1}^{i}\sum_{y_j} P(y_i, y_{i+1}, y_j|\mathbf{x})\boldsymbol{\phi}(y_j) + \sum_{j=i+1}^{n}\sum_{y_j} P(y_i, y_{i+1}, y_j|\mathbf{x})\boldsymbol{\phi}(y_j) \\
=&\ \alpha(y_i, y_{i+1}, i) + \beta(y_i, y_{i+1}, i)
\end{aligned}$$

$$\begin{aligned}
&\alpha(y_0, y_1, 0) \equiv P(y_0, y_1|\mathbf{x})\boldsymbol{\phi}(y_0) \\
&\alpha(y_i, y_{i+1}, i) \equiv P(y_i, y_{i+1}|\mathbf{x})\boldsymbol{\phi}(y_i) + P(y_{i+1}|y_i, \mathbf{x})\sum_{y_{i-1}} \alpha(y_{i-1}, y_i, i-1) \\
&\beta(y_{n-1}, y_n, n-1) \equiv P(y_{n-1}, y_n|\mathbf{x})\boldsymbol{\phi}(y_n) \\
&\beta(y_i, y_{i+1}, i) \equiv P(y_i, y_{i+1}|\mathbf{x})\boldsymbol{\phi}(y_{i+1}) + P(y_i|y_{i+1}, \mathbf{x})\sum_{y_{i+2}} \beta(y_{i+1}, y_{i+2}, i+1)
\end{aligned}$$

The resulting algorithm has complexity $O(nm^2)$, which is the same as the standard forward-backward inference algorithm for CRF.

6.3.2 Hard vs. Soft Projection

Projecting expectations instead of one-best label assignments from English to foreign language can be thought of as a soft version of the method described in (Das and Petrov, 2011) and (Ganchev et al., 2009). Soft projection has its advantage: when the English model is not certain about its predictions, we do not have to commit to the current best prediction. The foreign model has more freedom to form its own belief since any marginal distribution it produces would deviates from a flat distribution by just about the same amount. In general, preserving uncertainties till later is a strategy that has benefited many NLP tasks (Finkel et al., 2006). Hard projection can also be treated as a special case in our framework. We can simply recalibrate posterior marginal of English by assigning probability mass 1 to the most likely outcome, and zero everything else out, effectively taking the argmax of the marginal at each word position. We refer to this version of expectation as the "hard"

expectation. In the hard projection setting, GE training resembles a "project-then-train" style semi-supervised CRF training scheme (Yarowsky and Ngai, 2001; Täckström et al., 2013). In such a training scheme, we project the one-best predictions of English CRF to the foreign side through word alignments, then include the newly "tagged" foreign data as additional training data to a standard CRF in the foreign language. Rather than projecting labels on a per-word basis, Yarowsky and Ngai (2001) also explored an alternative method for noun-phrase (NP) bracketing task that amounts to projecting the spans of NPs based on the observation that individual NPs tend to retain their sequential spans across translations. We experimented with the same method for NER, but found that this method of projecting the NE spans does not help in reducing noise and actually lowers model performance.

Besides the difference in projecting expectations rather than hard labels, our method and the "project-then-train" scheme also differ by optimizing different objectives: CRF optimizes maximum conditional likelihood of the observed label sequence, whereas GE minimizes squared error between model's expectation and "hard" expectation based on the observed label sequence. In the case where squared error loss is replaced with a KL-divergence loss, GE has the same effect as marginalizing out all positions with unknown projected labels, allowing more robust learning of uncertainties in the model. As we will show in the experimental results in Section 6.4.2, soft projection in combination of the GE objective significantly outperforms the project-then-train style CRF training scheme.

6.3.3 Source-side noise

An additional source of noise comes from errors generated by the source-side English CRF models. We know that the English CRF models gives F_1 score of 81.68% on the OntoNotes dataset for English-Chinese experiment, and 90.45% on the CoNLL-03 dataset for English-German experiment. We present a simple way of modeling English-side noise by picturing the following process: the labels assigned by the English CRF model (denoted as $\mathbf{y}$) are some noised version of the true labels (denoted as $\mathbf{y}^*$). We can recover the probability of the true labels by marginalizing over the observed labels: $P(\mathbf{y}^*|\mathbf{x}) = \sum_{\mathbf{y}} P(\mathbf{y}^*|\mathbf{y}) * P(\mathbf{y}|\mathbf{x})$. $P(\mathbf{y}|\mathbf{x})$ is the posterior probabilities given by the CRF model, and we can approximate $P(\mathbf{y}^*|\mathbf{y})$ by the column-normalized error confusion matrix shown in Table 6.1. The top

	O	PER	LOC	ORG	GPE
O	291339	391	141	1281	221
PER	1263	6721	5	56	73
LOC	409	23	546	123	133
ORG	2423	143	52	8387	196
GPE	566	239	69	668	6604

	O	PER	LOC	ORG	MISC
O	81209	24	38	155	103
PER	77	5725	41	69	10
LOC	49	40	3743	127	60
ORG	178	102	142	4075	91
MISC	175	41	30	114	1826

Table 6.1: Raw counts in the error confusion matrix of English CRF models.

table contains the counts on OntoNotes test data, and the bottom table contains CoNLL-03 test data counts. Rows are the true labels and columns are the observed labels. For example, item at row 2, column 3 of the top table reads: we observed 5 times where the true label should be PERSON, but English CRF model output label LOCATION. This source-side noise model is likely to be overly simplistic. Generally speaking, we could build much more sophisticated noising model for the source-side, possibly conditioning on context, or capturing higher-order label sequences.

6.4 Experiments

We conduct experiments on Chinese and German NER. We evaluate CLiPER in two learning settings: weakly supervised and semi-supervised. In the weakly supervised setting, we simulate the condition of having no labeled training data, and evaluate the model learned from bitext alone. We then vary the amount of labeled data available to the model, and examine the model's learning curve. In the semi-supervised setting, we assume our model has access to the full labeled data; our goal is to improve performance of the supervised method by learning from additional bitext.

6.4.1 Dataset and Setup

We used the latest version of the Stanford NER Toolkit (Finkel et al., 2005)[6] as our base CRF model in all experiments. Features for English, Chinese and German CRFs are documented extensively in (Che et al., 2013) and (Faruqui and Padó, 2010) and omitted here for brevity. It it worth noting that the current Stanford NER models include recent improvements from semi-supervised learning approaches that induce distributional similarity features from large word clusters. These models represent the current state-of-the-art in supervised methods, and serve as a very strong baseline.

For Chinese NER experiments, we follow the same setup as described in Section 2.5.3.

For German NER experiments, we evaluate using the standard CoNLL-03 NER corpus (Sang and Meulder, 2003). The labeled training set has 12k and 15k sentences. We used the de-en portion of the *News Commentary*[7] data from WMT13 as bitext. The English CRF tagger trained on CoNLL-03 English training corpus gives F_1 score of 90.4% on the CoNLL-03 test set.

We compare against three approaches that were introduced in Chapter 5: semi-supervised learning method using factored bilingual models with Gibbs sampling, Integer Linear Programming (ILP) and dual decomposition. We also compare against constraint-driven bilingual-reranking approach from Burkett et al. (2010b).

Since the objective function in Eq. 6.3 is non-convex, we adopted the early stopping training scheme from (Turian et al., 2010) as the following: after each iteration in L-BFGS training, the model is evaluated against the development set; the training procedure is terminated if no improvements have been made in 20 iterations.

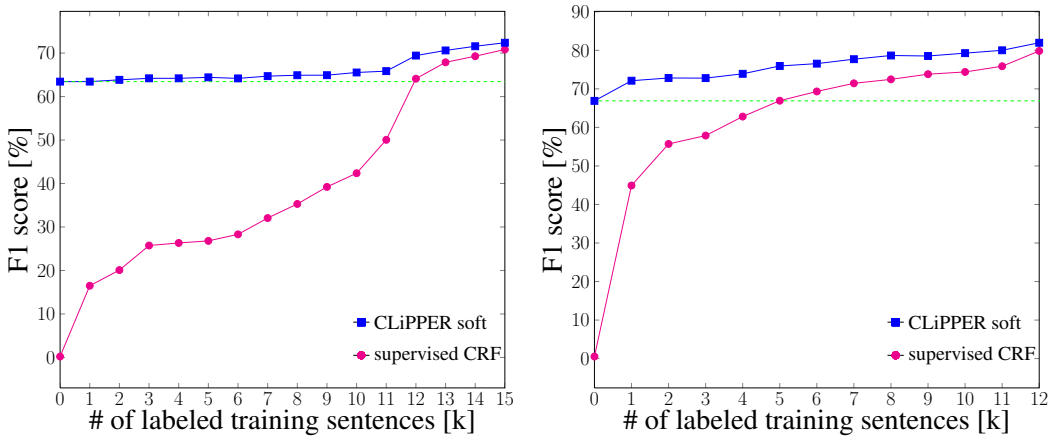
6.4.2 Weakly Supervised Results

The plots in Figure 6.3 show results of weakly supervised learning experiments. Vertical axes show the F_1 score on the development and test set, respectively. Performance curves of the supervised CRF and the "project-then-train" CRF are plotted for comparison.

Remarkably, on Chinese test set, our proposed method (CLiPER) achieves a F_1 score

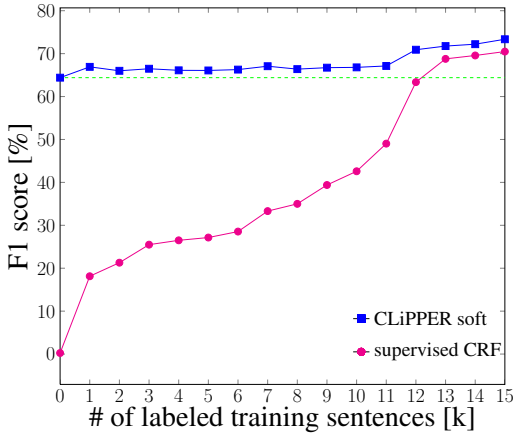
[6] `http://www-nlp.stanford.edu/ner`

[7] `http://www.statmt.org/wmt13/training-parallel-nc-v8.tgz`

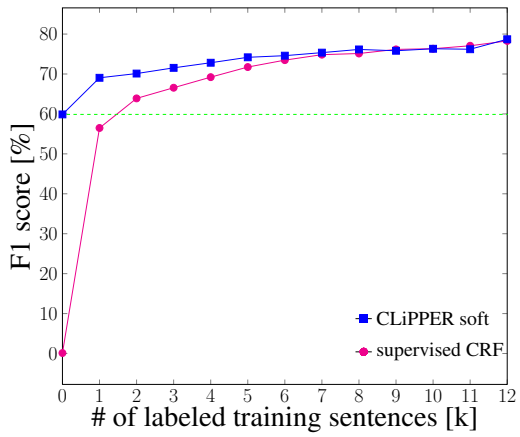


(a) Chinese Dev

(b) German Dev

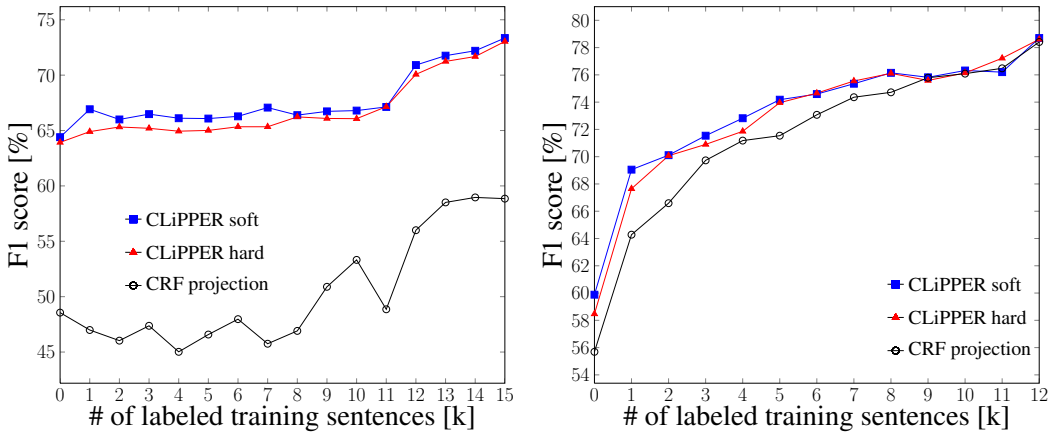


(c) Chinese Test



(d) German Test

Figure 6.3: Performance curves of CLiPER with varying amounts of available labeled training data in a weakly supervised setting.



(a) Soft vs. Hard on Chinese Test

(b) Soft vs. Hard on German Test

Figure 6.4: Performance curves of CLiPER with varying amounts of available labeled training data in a weakly supervised setting.

of 64.4% with 80k bitext, when no labeled training data is used. In contrast, the supervised CRF baseline would require as much as 12k labeled sentences to attain the same accuracy. Results on the German test set are less striking. With no labeled data and 40k of bitext, CLiPER performs at F_1 of 60.0%, the equivalent of using 1.5k labeled examples in the supervised setting. When combined with 1k labeled examples, performance of CLiPER reaches 69%, a gain of over 5% absolute over supervised CRF. We also notice that the supervised CRF model learns much faster in German than Chinese. This result is not too surprising, since it is well recognized that Chinese NER is more challenging than German or English due to the lack of orthographical features, such as word capitalization. Chinese NER relies more on lexicalized features, and therefore needs more labeled data to achieve good coverage. The results also suggest that CLiPER seems to be very effective at transferring lexical knowledge from English to Chinese.

The two figures in Figure 6.4 compares soft GE projection with hard GE projection and the "project-then-train" style CRF training scheme (*cf.* Section 6.3.2). Vertical axes show the F_1 score on the development and test set, respectively. Performance curves of supervised CRF and "project-then-train" CRF are plotted for comparison. We observe that both soft and hard GE projection significantly outperform the "project-then-train" style training scheme. The difference is especially pronounced on the Chinese results when fewer labeled

		Chinese			German		
		P	R	F_1	P	R	F_1
CRF		79.87	63.62	70.83	88.05	73.03	79.84
CLiPER$_s$	10k	81.36	65.16	72.36	85.23	77.79	81.34
	20k	**81.79**	64.80	72.31	88.11	75.93	81.57
	40k	79.24	**66.08**	72.06	**88.25**	76.52	**81.97**
	80k	80.26	65.92	**72.38**	87.80	**76.82**	81.94

Table 6.2: Chinese and German NER results on the development set using CLiPER with varying amounts of unlabeled bitext (10k, 20k, etc.).

examples are available. Soft projection gives better accuracy than hard projection when no labeled data is available, and also has a faster learning rate.

6.4.3 Semi-supervised Results

In the semi-supervised experiments, we let the CRF model use the full set of labeled examples in addition to the unlabeled bitext. Table 6.2 shows results on the development dataset for Chinese and German using 10-80k bitext. The best number of each column is highlighted in bold. The F_1 score improvements over CRF baseline in all cases are statistically significant at 99.9% confidence level.

We see that with merely 10k additional bitext, CLiPER is able to improve significantly over state-of-the-art CRF baselines by as much as 1.5% F_1 on both Chinese and German. With more unlabeled data, we notice a tradeoff between precision and recall on Chinese. The final F_1 score on Chinese at 80k level is only marginally better than 10k. On the other hand, we observe a modest but steady improvement on German as we add more unlabeled bitext, up until 40k sentences. We select the best configurations on development set (80k for Chinese and 40k for German) to evaluate on test set.

Results on the test set are shown in Table 6.3. Best number of each column is highlighted in bold. All except the last two rows and the were CRF is the supervised baseline. CRF$_{ptt}$ is the "project-then-train" semi-supervised scheme for CRF. Results of BURKETT, GIBBS, ILP, DUALDECOMP have been presented earlier in Table 5.8 in the previous chapter. CLIPER$_s$ and CLIPER$_h$ are the soft and hard projections. § indicates F_1 scores that are

	Chinese			German		
	P	R	F_1	P	R	F_1
CRF	79.09	63.59	70.50	86.69	71.30	78.25
CRF_{ptt}	**84.01**	45.29	58.85	81.50	**75.56**	78.41
Burkett	79.25	65.67	71.83	84.00	72.17	77.64
ILP	81.31	65.50	72.55	85.99	72.98	78.95
Gibbs	80.31	65.78	72.33	85.98	72.37	78.59
DualDecomp	78.55	66.54	72.05	85.19	72.98	78.62
$CLiPER_h$	83.67	64.80	$73.04^{\S\ddagger}$	86.52	72.02	78.61^{*}
$CLiPER_s$	82.57	**65.99**	$\mathbf{73.35}^{\S\dagger\star}_{\diamond *}$	**87.11**	72.56	$\mathbf{79.17}^{\ddagger\star}_{*\S}$

Table 6.3: Chinese and German NER results on the test set.

statistically significantly better than CRF baseline at 99.5% confidence level; $\star$ marks significance over CRF_{ptt} with 99.5% confidence; † and ‡ marks significance over GIBBS with 99.9% and 94% confidence; and $\diamond$ marks significance over ILP with 99.7% confidence; $*$ marks significance over BURKETT with 99.9% confidence.

All semi-supervised baselines are tested with the same number of unlabeled bitext as CLiPER in each language. The "project-then-train" semi-supervised training scheme severely hurts performance on Chinese, but gives a small improvement on German. Moreover, on Chinese it learns to achieve high precision but at a significant loss in recall. On German its behavior is the opposite. Such a drastic and erratic imbalance suggests that this method is not robust or reliable. The other three semi-supervised baselines (row 3-5) all show improvements over the CRF baseline, consistent with their reported results. $CLiPER_s$ gives the best results on both Chinese and German, yielding statistically significant improvements over all baselines except for ILP on German. The hard projection version of CLiPER also gives sizable gain over CRF. However, in comparison, $CLiPER_s$ is superior.

The improvements of $CLiPER_s$ over CRF on Chinese test set is over 2.8% in absolute F_1. The improvement over CRF on German is almost a percent. To our knowledge, these are the best reported numbers on the OntoNotes Chinese and CoNLL-03 German datasets.

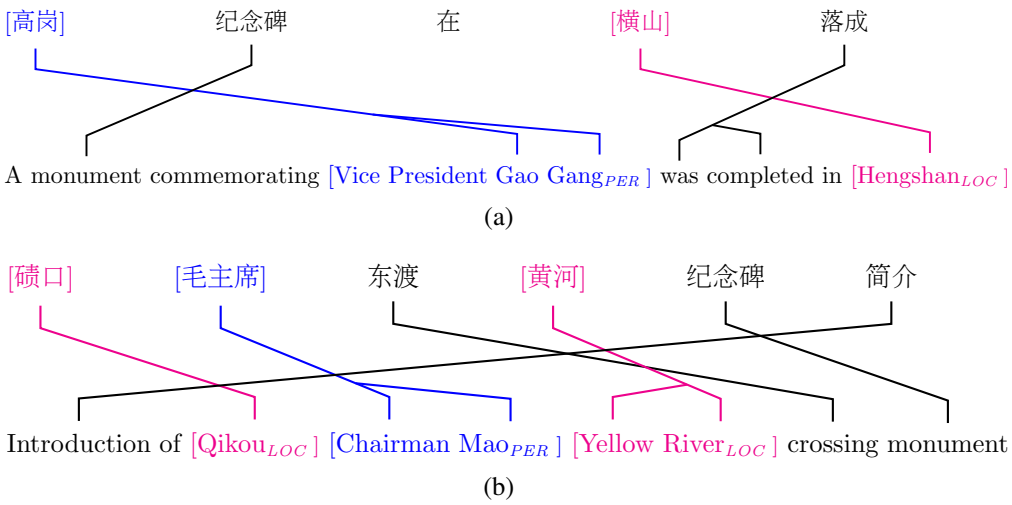


Figure 6.5: Examples of aligned sentence pairs in Chinese and English.

6.4.4 Efficiency

Another advantage of our proposed approach is efficiency. Because we eliminated the previous multi-stage "project-then-train" paradigm, but instead integrating the semi-supervised and supervised objective into one joint objective, we are able to attain significant speed improvements. Table 6.4 shows the training time required to produce models that give results in Table 6.3.

6.5 Error Analysis and Discussion

Figure 6.5 gives two examples of CLiPER in action. Both examples have a named entity that immediately proceeds the word "纪念碑" (monument) in the Chinese sentence. In Figure 6.5a, the word "高岗" has literal meaning of *a hillock located at a high position*, which also happens to be the name of a former vice president of China. Without having previously observed this word as a person name in the labeled training data, the CRF model does not have enough evidence to believe that this is a PERSON, instead of LOCATION. But the aligned words in English ("Gao Gang") are clearly part of a person name as they were preceded by a title ("Vice President"). The English model has high expectation that the aligned Chinese word of "Gao Gang" is also a PERSON. Therefore, projecting the

	Chinese	German
CRF	00h19m30s	00h07m15s
CRF_{ptt}	00h34m02s	00h12m45s
Burkett	06h16m21s	02h42m17s
ILP	16h42m07s	04h49m02s
Gibbs	03h17m25s	01h01m45s
DualDecomp	00h54m34s	00h22m43s
CLiPER_h	01h28m57s	00h16m30s
CLiPER_s	01h40m43s	00h18m51s

Table 6.4: CLiPER Timing stats during projection and model training.

English expectations to Chinese provides a strong clue to help disambiguating this word. Figure 6.5b gives another example: the word "黄河"(Huang He, the Yellow River of China) can be confused with a person name since "黄"(Huang or Hwang) is also a common Chinese last name.[8]. Again, knowing the translation in English, which has the indicative word "River" in it, helps disambiguation.

The CRF_{ptt} and CLiPER_h methods successfully labeled these two examples correctly, but failed to produce the correct label for the example in Figure 6.1. On the other hand, a model trained with the CLiPER_s method does correctly label both entities in Figure 6.1, demonstrating the merits of the soft projection method.

6.6 Summary

In this chapter, we introduced a domain and language independent method for training discriminative models by projecting expectations across bitext. This method is designed for scenarios where we want to learn a NER tagger for a new language where we have very little or no annotated training data. Experiments on Chinese and German NER show that our method, learned over bitext alone, can rival performance of supervised models trained with thousands of labeled examples. Furthermore, applying our method in a setting where

[8]In fact, a people search of the name 黄河 on the Chinese equivalent of Facebook (www.renren.com) returns over 13,000 matches.

all labeled examples are available also shows improvements over state-of-the-art supervised methods. Our experiments also showed that soft expectation projection is more favorable to hard projection.

Chapter 7

Conclusions

In this thesis, we presented a set of techniques and new models for improving the performance of sequence labeling models, specifically for Named Entity Recognition (NER) task.

In Chapter 3, we first looked at ways to improve the learning of sequence models. In particular, we addressed the problem of feature weight undertraining, which is a prevalent issue in training feature-rich log-linear models. We propose two solutions to counter this problem. The first solution uses a product-of-experts model to break-up feature coadaptation by creating diverse experts each capturing some parts of the information contained in the training data. Traditional approaches to the product-of-experts model rely heavily on domain expertise and manual effort to craft a diverse set of experts, which is more of an art, and does not generalize well to new domains and datasets. We proposed an automatic expert-induction method by applying a elitist-lasso style regularization term to a logarithmic opinion pooling model, where the regularization term encourages the experts (opinion pools) to be as diverse from each other as possible, while being accurate at their predictions. A second solution we proposed is based on the recent line of work on dropout learning and feature noising. We extend the second-order approximation based feature noising method to structured prediction, and derived an efficient dynamic-programming algorithm for computing the gradients necessary for optimizing the noised version of the CRF learning objective.

We then turn to examine the internal model structure and feature representation of common log-linear models used for sequence labeling. In Chapter 4, we question the conventional wisdom that linear models suffice for most NLP tasks, and little or no advantage can be gained by moving to a non-linear model architecture. We argue that the suitability of a linear architecture cannot be judged independent of the form of feature representations used by the model. We draw the connection between conventional CRF models and the newly proposed deep architecture sequence models, and examine the link between linearity and feature representation empirically. We found that non-linear architectures are more effective in low dimensional continuous input spaces, but that they are not better suited for conventional high-dimensional discrete input spaces.

The focus of the first two parts of our investigation was on model building and learning methods from an inward-looking perspective. In Chapter 5, we turn our lens outwards and examine the sources of signals we use for training sequence models. By and large the main source of training signal comes from manually annotated labels in supervised training data. Unfortunately manual annotation is a resource that is hard to obtain at scale without incurring high cost. Recent advance in semi-supervised learning literature has demonstrated that unlabeled data, which exists in huge volumes, can significantly improve system performance when used in an intelligent way. We draw the observation that there is a rich source of under-utilized information hidden in the vast amount of bilingual translated texts, which has become easily available over the past decade, thanks to advances in statistical machine translation research. Hence, we devise a bilingual tagging framework where we explore links between two pieces of translated text as a form of semantic constraints, and enforce agreements between models in the two languages while decoding a piece of bitext. We show that such agreement constraints can be captured nicely in a factored probabilistic model framework, as well as in an Integer Linear Programming framework. Our results show that bilingual systems that explore these constraints can significantly outperform monolingual systems. Furthermore, we carried out experiments demonstrating that if we take the bilingually decoded output as additional training data in an uptraining approach, we can also improve monolingual system performance.

One drawback in the agreement-seeking bilingual approach is that it requires two

readily-trained models in each side of the language pair as prerequisites. For many languages we do not have any annotated data to train a baseline NER system, and therefore this approach would not apply. To address this concern, in Chapter 6 we further propose a projection-based method that only requires a trained model on one side of the bilingual pair, and effectively projects over bitext signals from the trained model to the other language where we do not have any training signal. Our method makes a key improvement over past projection-based methods, by taking into account label uncertainty and projects model expectations instead of hard label assignments.

There are several high-level conclusions we can draw from this work. Chapter 3 and Chapter 5 both explore ways to get more out of existing training resources without incurring more annotation costs. Chapter 3 improves training data quality by explicitly modeling feature noise in the generation of the training data; Chapter 5 finds more training data by taking advantage of translated text. The empirical success of both attempts showcase that getting more annotated data is not necessarily the most efficient way to improve supervised system performance, and is certainly not the only way. We can actually get quite a lot of mileage by taking a deeper look at resources we already have.

Another common theme we observe from the results obtained in this thesis is that approximate methods that model uncertainties can often lead to significant gains. We have seen this from modeling uncertainty in training data generation (Chapter 3), uncertainty in word alignment (Chapter 5), as well as uncertainty in model output (Chapter 6). Joint modeling of dependent sub-problems can also be beneficial. We saw how joint modeling word alignment and NER in Chapter 5 yields improvements in both, and joint modeling NER in two languages can sometimes improve NER results in both languages.

But of course, nothing comes for free in the world of engineering. As we explore richer data sources such as bitext, we are also facing more challenging engineering problems. For example, modeling the joint probability of a bilingual sequence can quickly render exact inference algorithms intractable. The three approximate inference algorithms we examined in Chapter 5 are quite effective for this problem, and can be broadly applied to many other complex inference problems. However, instead of randomly choosing any approximate inference method and blindly applying it to a problem, it is worth the effort to explore a few options and make problem-specific adjustments, as testified by the various speed and

accuracy tradeoffs of the inference methods we experimented with.

There are many promising directions for future work, building off the work presented in this thesis. For example, the soft-projection framework in Chapter 6 can be greatly extended in two orthogonal directions: we can apply it to learn models for many more language pairs, and we can apply it to more tasks such as dependency parsing and semantic role labeling. In both the soft-projection model and the bilingual tagging model, alignment error is still a major cause of failure. It will be very interesting to explore alignment uncertainties in the soft-projection framework. The model is likely to become very complex and too hard for exact inference, but sampling-based approaches like the Gibbs sampler we presented can offer viable solutions. One of the most challenging problems in NLP is domain adaptation, and although in Chapter 3 we presented some attempts to improve model robustness when applied to out-of-domain data via better regularization, fully robust models and learning methods is still an aloof goal. One interesting idea that needs yet to be explored is to extend the bilingual ideas in this thesis to bi-domain—imagine if we can obtain some form of comparable data that can relate two domains, then we can encode the connection as constraints, and transfer knowledge from one domain to the other. But exactly what sort of data and what sort of constraints can facilitate this purpose still remains unclear.

The vision that one day we will be able to automatically extract semantic relational information from natural language text accurately is a goal that hopefully is not that far away. The work presented in this thesis represents a small fraction of the total research efforts that are aiming at achieving that goal. I hope that the ideas, methods and learnings from this work will inspire more ideas, better methods, and greater learnings in the future, and help moving the needle towards to the Holy Grail of natural language understanding.

Bibliography

Abu-Mostafa, Yaser S. 1990. Learning from hints in neural networks. *Journal of Complexity* 6(2):192–198.

Alshawi, Hiyan, Srinivas Bangalore, and Shona Douglas. 2000. Head-transducer models for speech translation and their automatic acquisition from bilingual data. *Machine Translation (MT)* 15.

Ando, Rie K., and Tong Zhang. 2005a. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research (JMLR)* 6:1817–1853.

Ando, Rie Kubota, and Tong Zhang. 2005b. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Attali, Jean-Gabriel, and Gilles Pagés. 1997. Approximations of functions by a multilayer perceptron: a new approach. *Neural Networks* 10:1069–1081.

Babych, Bogdan, and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT workshop on MT and other Language Technology Tools, Improving MT through other Language Technology Tools: Resources and Tools for Building MT*.

Bellare, Kedar, Gregory Druck, and Andrew McCallum. 2009. Alternating projections for learning with expectation constraints. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

Bengio, Yoshua. 2009. Learning deep architectures for AI. *Found. Trends Mach. Learn.* 2 (1):1–127.

Bengio, Yoshua, and Samy Bengio. 2000. Modeling high-dimensional discrete data with multi-layer neural networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Bengio, Yoshua, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2006. Greedy layer-wise training of deep networks. In *Proceedings of Advances in Neural Information Processing Systems (NIPS) 19*.

Berant, Jonathan, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*.

Bishop, Chris M. 1995. Training with noise is equivalent to Tikhonov regularization. *Neural Computation* 7(1):108–116.

Blum, Avrim, and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*.

Borthwick, Andrew Eliot. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. PhD thesis, New York, NY, USA. AAI9945252.

Brown, Peter F., Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.* 18(4): 467–479.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1991. Word-sense disambiguation using statistical methods. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Bunescu, Razvan, and Raymond J. Mooney. 2004. Collective information extraction with relational Markov networks. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Burges, Chris J.C., and Bernhard Schölkopf. 1997. Improving the accuracy and speed of support vector machines. In *Advances in Neural Information Processing Systems (NIPS)*.

Burkett, David, John Blitzer, and Dan Klein. 2010a. Joint parsing and alignment with weakly synchronized grammars. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*.

Burkett, David, and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Burkett, David, Slav Petrov, John Blitzer, and Dan Klein. 2010b. Learning better monolingual models with unannotated bilingual text. In *Proceedings of the 14th Conference on Computational Natural Language Learning (CoNLL)*.

Camastra, Francesco. 2003. Data dimensionality estimation methods: A survey. *Pattern Recognition* 36:2945–2954.

Carlson, Andrew, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Proceedings of the Third International Conference on Web Search and Data Mining (WSDM)*.

Chang, Ming-Wei, Lev Ratinov, and Dan Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Che, Wanxiang, Mengqiu Wang, and Christopher D. Manning. 2013. Named entity recognition with bilingual constraints. In *Proceedings of the 14th Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Chen, Yufeng, Chengqing Zong, and Keh-Yih Su. 2010. On jointly recognizing and aligning bilingual named entities. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Chieu, Hai Leong, and Loo-Nin Teow. 2012. Combining local and non-local information with dual decomposition for named entity recognition from text. In *Proceedings of 15th International Conference on Information Fusion (FUSION)*.

Chinchor, Nancy. 1998. Overview of MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Clark, Alexander. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics (EACL)*.

Collins, Michael. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Collins, Michael, and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Collobert, Ronan. 2011. Deep learning for efficient discriminative parsing. In *Proceedings of the 14th International Conference on Articial Intelligence and Statistics (AISTATS)*.

Collobert, Ronan, and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*.

Collobert, Ronan, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research (JMLR)* 12:2461–2505.

Cromières, Fabien, and Sadao Kurohashi. 2009. An alignment algorithm using belief propagation and a structure-based distortion model. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*.

Dahl, George E., Dong Yu, Li Deng, and Alex Acero. 2012. Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing* 20:33–42.

Das, Dipanjan, and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*.

Das, Dipanjan, and Noah A. Smith. 2012. Graph-based lexicon expansion with sparsity-inducing penalties. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

DeNero, John, and Dan Klein. 2008. The complexity of phrase alignment problems. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*.

DeNero, John, and Klaus Macherey. 2011. Model-based aligner combination using dual decomposition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*.

Do, Trinh-Minh-Tri, and Thierry Artieres. 2010. Neural conditional random fields. In *Proceedings of the 13rd International Conference on Articial Intelligence and Statistics (AISTATS)*.

Doddington, George R, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ACE) program-tasks, data, and evaluation. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)*.

Druck, Gregory. 2011. *Generalized Expectation Criteria for Lightly Supervised Learning*. PhD thesis, University of Massachusetts Amherst.

Druck, Gregory, Gideon Mann, and Andrew McCallum. 2007. Leveraging existing resources using generalized expectation criteria. In *Proceedings of NIPS Workshop on Learning Problem Design*.

Druck, Gregory, and Andrew McCallum. 2010. High-performance semi-supervised learning using discriminatively constrained generative models. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*.

Druck, Gregory, Burr Settles, and Andrew McCallum. 2009. Active learning by labeling features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Duchi, John, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)* 12:2121–2159.

Duchi, John, and Yoram Singer. 2009. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research (JMLR)* 10:2899–2934.

Efron, Bradley, and Robert J. Tibshirani. 1993. *An Introduction to the Bootstrap*. Chapman & Hall.

Erhan, Dumitru, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research (JMLR)* 11:625–660.

Faruqui, Manaal, and Sebastian Padó. 2010. Training and evaluating a German named entity recognizer with semantic generalization. In *Proceedings of The 9th Conference on Natural Language Processing (KONVENS)*.

Finkel, Jenny Rose, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Finkel, Jenny Rose, Christopher D. Manning, and Andrew Y. Ng. 2006. Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Fossum, Victoria, and Steven Abney. 2005. Automatically inducing a part-of-speech tagger by projecting from multiple source languages across aligned corpora. In *Proceedings of the Second International Joint Conference on Natural Language Processing (IJCNLP)*.

Fu, Ruiji, Bing Qin, and Ting Liu. 2011. Generating Chinese named entity data from a parallel corpus. In *Proceedings of 5th International Joint Conference on Natural Language Processing (IJCNLP)*.

Ganchev, Kuzman, Jo ao Graça, Jennifer Gillenwater, and Ben Taskar. 2010. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research (JMLR)* 10:2001–2049.

Ganchev, Kuzman, and Dipanjan Das. 2013. Cross-lingual discriminative learning of sequence models with posterior regularization. In *Proceedings of EMNLP*.

Ganchev, Kuzman, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*.

Ganchev, Kuzman, Joao Graca, John Blitzer, and Ben Taskar. 2008. Multi-view learning over structured and non-identical outputs. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

Geman, Stuart, and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transitions on Pattern Analysis and Machine Intelligence* 6:721–741.

Goldberg, Andrew B. 2010. *New Directions in Semi-supervised Learning*. PhD thesis, University of Wisconsin-Madison.

Gonzalez, Joseph, Yucheng Low, Arthur Gretton, and Carlos Guestrin. 2011. Parallel Gibbs sampling: From colored fields to thin junction trees. In *In Artificial Intelligence and Statistics (AISTATS)*.

Graça, Joao, Kuzman Ganchev, and Ben Taskar. 2008. Expectation maximization and posterior constraints. In *Advances in Neural Information Processing Systems (NIPS)*.

Grishman, Ralph, and Beth Sundheim. 1996. Message understanding conference-6: A brief history. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING)*.

Haghighi, Aria, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised ITG models. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*.

Heskes, Tom. 1998. Selecting weighting factors in logarithmic opinion pools. In *Advances in Neural Information Processing Systems (NIPS)*.

Hinton, Geoffery. 1999. Products of experts. In *Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN)*.

Hinton, Geoffrey E., Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation* 18:1527–1554.

Hinton, Geoff E., and Ruslan R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Hinton, Geoffrey E., Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Hovy, Eduard, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: the 90% solution. In *Proceedings of the Human Language Technology conference - North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL)*.

Huang, Fei, and Stephan Vogel. 2002. Improved named entity translation and bilingual named entity extraction. In *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces (ICMI)*.

Huang, Liang, Wenbin Jiang, and Qun Liu. 2009. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Joachims, Thorsten. 2004. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*. Kluwer Academic Publishers.

Kim, Sungchul, Kristina Toutanova, and Hwanjo Yu. 2012. Multilingual named entity recognition using parallel data and metadata from Wikipedia. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi. 1983. Optimization by simulated annealing. *Science* 220:671–680.

Klein, Dan. 2005. *The Unsupervised Learning of Natural Language Structure*. PhD thesis, Stanford University.

Kohli, Pushmeet, L'ubor Ladicky, and Philip HS Torr. 2009. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision* 82(3):302–324.

Komodakis, Nikos, Nikos Paragios, and Georgios Tziritas. 2007. MRF optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8.

Koo, Terry, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Koo, Terry, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag.

2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Kowalski, Matthieu, and Bruno Torrésani. 2009. Sparsity and persistence: mixed norms provide simple signal models with dependent coefficients. *Signal, Image and Video Processing* 3(3):251–264.

Lafferty, John D., Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*.

Lavergne, Thomas, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Le, Hai-Son, Alexandre Allauzen, and Franois Yvon. 2012a. Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL)*.

Le, Quoc, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg Corrado, Jeff Dean, and Andrew Ng. 2012b. Building high-level features using large scale unsupervised learning. In *Proceedings of the 29th International Conference in Machine Learning (ICML)*.

LeCun, Yann, and Yoshua Bengio. 2007. Scaling learning algorithms towards AI. In L. Bottou et al. (ed.), *Large-Scale Kernel Machines*. MIT Press.

Lee, Honglak, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*.

Li, Qi, Haibo Li, Heng Ji, Wen Wang, Jing Zheng, and Fei Huang. 2012a. Joint bilingual name tagging for parallel corpora. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM)*.

Li, Shen, Jo ao Graça, and Ben Taskar. 2012b. Wiki-ly supervised part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Li, Zhifei, and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Liang, Percy. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

Liang, Percy, Hal Daume, and Dan Klein. 2008. Structure compilation: Trading structure for features. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*.

Liang, Percy, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the Human Language Technology conference - North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL)*.

Liu, Dong C., and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Math. Programming* 45:503–528.

Mann, Gideon, and Andrew McCallum. 2010. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *Journal of Machine Learning Research (JMLR)* 11:955–984.

Martins, Andre, Noah Smith, and Eric Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*.

Martins, André F. T., Noah A. Smith, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2011. Dual decomposition with many overlapping components. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Martins, Andre F. T., Noah A. Smith, Eric P. Xing, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2010. Augmenting dual decomposition for map inference. In *Proceedings of the International Workshop on Optimization for Machine Learning (OPT)*.

Matsuoka, Kiyotoshi. 1992. Noise injection into inputs in back-propagation learning. *Systems, Man and Cybernetics, IEEE Transactions on* 22(3):436–440.

McCallum, Andrew, Dayne Freitag, and Fernando C. N. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*.

McClosky, David, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology conference - North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL)*.

McCullagh, Peter. 1984. Generalized linear models. *European Journal of Operational Research* 16(3):285–292.

Mérialdo, Bernard. 1994. Tagging English text with a probabilistic model. *Computational Linguistics (CL)* 20(2):155–171.

Mikolov, Tomas, Anoop Deoras, Stefan Kombrink, Lukas Burget, and Jan Cernocky. 2011. Empirical evaluation and combination of advanced language modeling techniques. In *Proceedings 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Miller, Scott, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of the Human Language Technology conference - North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL)*.

Naseem, Tahira, Benjamin Snyder, Jacob Eisenstein, and Regina Barzilay. 2009. Multilingual part-of-speech tagging: Two unsupervised approaches. *Journal of Artificial Intelligence Research (JAIR)* 36:1076–9757.

Palmer, Martha, Dan Gildea, and Paul Kingsbury. 2005. The proposition bank: A corpus annotated with semantic roles. *Computational Linguistics* 31(1).

Peng, Jian, Liefeng Bo, and Jinbo Xu. 2009. Conditional neural fields. In *Advances in Neural Information Processing Systems (NIPS)*.

Petrov, Slav, Pi-Chuan Chang, Michael Ringgaard, and Hiyan Alshawi. 2010. Uptraining for accurate deterministic question parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Pitkow, Xaq. 2012. Compressive neural representation of sparse, high-dimensional probabilities. In *Advances in Neural Information Processing Systems (NIPS)*.

Prabhavalkar, Rohit, and Eric Fosler-Lussier. 2010. Backpropagation training for multilayer conditional random field based phone recognition. In *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*.

Punyakanok, Vasin, Dan Roth, Wen tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of International Conference on Computational Linguistics (COLING)*.

Rabiner, Lawrence, and Biing-Hwang Juang. 1986. An introduction to hidden Markov models. *ASSP Magazine, IEEE* 3(1):4–16.

Ratinov, Lev, and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL)*.

Riedel, Sebastian, and Andrew McCallum. 2011. Fast and robust joint models for biomedical event extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Rifai, Salah, Yoshua Bengio, Yann N. Dauphin, and Pascal Vincent. 2012. A generative process for sampling contractive auto-encoders. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*.

Rifai, Salah, Yann Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. 2011a. The manifold tangent classifier. In *Advances in Neural Information Processing Systems (NIPS)*.

Rifai, Salah, Xavier Glorot, Yoshua Bengio, and Pascal Vincent. 2011b. Adding noise to the input of a model trained with a regularized objective. *arXiv preprint arXiv:1104.3250*.

Riloff, Ellen, and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the 13rd AAAI Conference on Artificial Intelligence (AAAI)*.

Roth, Dan, and Wen-tau Yih. 2005. Integer linear programming inference for conditional random fields. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*.

Rumelhart, David E., and James L. McClelland. 1986. On learning the past tenses of English verbs. In James L. McClelland and David E. Rumelhart (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume II*. MIT Press.

Rush, Alexander M., and Michael Collins. 2012. A tutorial on dual decomposition and Lagrangian relaxation for inference in natural language processing. *Journal of Artificial Intelligence Research (JAIR)* 45:305–362.

Rush, Alexander M., Roi Reichert, Michael Collins, and Amir Globerson. 2012. Improved parsing and POS tagging using inter-sentence consistency constraints. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Rush, Alexander M., David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Samdani, Rajhans, Ming-Wei Chang, and Dan Roth. 2012. Unified expectation maximization. In *Proceedings of the 13th meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Sang, Erik F, and Jorn Veenstra. 1999. Representing text chunks. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics (EACL)*.

Sang, Erik F. Tjong Kim, and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In *Proceedings of the 7th Conference on Computational Natural Language Learning (CoNLL)*.

Schölkopf, Bernhard, , Kay Sung, Christopher J. C. Burges, Federico Girosi, Partha Niyogi, Tomaso Poggio, and Vladimir Vapnik. 1997. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing* 45:2758–2765.

Settles, Burr. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications*.

Simard, Patrice Y., Yann A. Le Cun, John S. Denker, and Bernard Victorri. 2000. Transformation invariance in pattern recognition: Tangent distance and propagation. *International Journal of Imaging Systems and Technology* 11(3):181–197.

Sindhwani, Vikas, Partha Niyogi, and Mikhail Belkin. 2005. A co-regularization approach to semi-supervised learning with multiple views. In *Proceedings of ICML Workshop on Learning with Multiple Views, International Conference on Machine Learning*.

Smith, Andrew, Trevor Cohn, and Miles Osborne. 2005. Logarithmic opinion pools for conditional random fields. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL)*.

Smith, Andrew, and Miles Osborne. 2007. Diversity in logarithmic opinion pools. *Named Entities: Recognition, classification and use: Special issue of Lingvistic Investigationes* 30(1):27–47.

Smith, Noah A. 2006. *Novel Estimation Methods for Unsupervised Discovery of Latent Structure in Natural Language Text*. PhD thesis, Johns Hopkins University.

Snyder, Benjamin, Tahira Naseem, and Regina Barzilay. 2009. Unsupervised multilingual grammar induction. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*.

Socher, Richard, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems (NIPS)*.

Socher, Richard, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. 2011b. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*.

Socher, Richard, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011c. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Surdeanu, Mihai, Ramesh Nallapati, and Christopher D. Manning. 2010. Legal claim identification: Information extraction with hierarchically labeled data. In *Proceedings of the LREC 2010 Workshop on the Semantic Processing of Legal Texts (SPLeT-2010)*.

Sutton, Charles, and Andrew McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *Proceedings of ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*.

Sutton, Charles, Michael Sindelar, and Andrew McCallum. 2007. Reducing weight undertraining in structured discriminative learning. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*.

Suzuki, Jun, and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Täckström, Oscar, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Turian, Joseph, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

van der Maaten, Laurens, Minmin Chen, Stephen Tyree, and Kilian Q. Weinberger. 2013. Learning with marginalized corrupted features. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Verleysen, Michel, Damien Francois, Geoffroy Simon, and Vincent Wertz. 2003. On the effects of dimensionality on data analysis with neural networks. In *Proceedings of the 7th International Work-Conference on Artificial and Natural Neural Networks: Part II*.

Viterbi, Andrew J. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on* 13(2):260–269.

Wager, Stefan, Sida Wang, and Percy Liang. 2013. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems (NIPS)*.

Wang, Mengqiu, Wanxiang Che, and Christopher D. Manning. 2013a. Effective bilingual constraints for semi-supervised learning of named entity recognizers. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI)*.

Wang, Mengqiu, Wanxiang Che, and Christopher D. Manning. 2013b. Joint word alignment and bilingual named entity recognition using dual decomposition. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Wang, Mengqiu, and Christopher D. Manning. 2013a. Effect of non-linear deep architecture in sequence labeling. In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP)*.

Wang, Mengqiu, and Christopher D. Manning. 2013b. Learning a product of experts with elitist lasso. In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP)*.

Wang, Mengqiu, and Christopher D. Manning. 2014. Cross-lingual projected expectation regularization for weakly supervised learning. *Transactions of the Association for Computational Linguistics (TACL)* 2(5):55–66.

Wang, Sida, and Christopher D. Manning. 2013c. Fast dropout training. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Wang, Sida, Mengqiu Wang, Christopher D. Manning, Percy Liang, and Stefan Wager. 2013c. Feature noising for log-linear structured prediction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Welsh, D. J. A., and M. B. Powell. 1967. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal* 10(1):85–86.

Xi, Chenhai, and Rebecca Hwa. 2005. A backoff model for bootstrapping resources for non-English languages. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP)*.

Yarowsky, David. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yarowsky, David, and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Zhuang, Tao, and Chengqing Zong. 2010. Joint inference for bilingual semantic role labeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zou, Hui, and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B* 67:301–320.