Syntactic Agreement in Bilingual Corpora

by

David Burkett

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

Doctor of Philosophy
in

Computer Science

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Fall 2012

The dissertation of David Burkett is approved.

Chair	Date
	Date
	Date

University of California, Berkeley

Fall 2012

Syntactic Agreement in Bilingual Corpora

Abstract

Syntactic Agreement in Bilingual Corpora

by

David Burkett

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Dan Klein, Chair

The task of automatic machine translation (MT) is the focus of a huge variety of active research efforts, both because of the intrinsic utility of this difficult task, and the theoretical and linguistic insights that arise from modeling relationships between natural languages. However, MT systems that leverage syntactic information are only recently becoming practical, and in a typical system of this sort, syntactic information is generated by monolingual parsers; the task of explicitly modeling syntactic relationships between target and source languages is yet to be fully explored.

This thesis investigates the problem of finding syntactic parse trees of target and/or source sentences that are more appropriate for use in a syntactic MT system. Two basic methodologies are explored.

First, we present a sequence of two statistical models that leverage bilingual information to improve the linguistic quality of syntactic parses, as measured by their ability to replicate human-generated gold-standard annotations. The first model uses word to word alignments as an external source of information, while the second models the alignments jointly. These models are both quite effective at improving the intrinsic quality of the parse trees, and the second model additionally improves word alignment performance. However, while the two models achieve similar parsing improvements, we find that improving parses in conjunction with word alignments is much more helpful for the downstream machine translation task.

In the next part of the thesis, we explore this finding further by investigating the effects on MT performance of *agreement* between parse trees and word alignments. We present a simple method for transforming input trees in a way that ignores gold-standard annotations, concentrating instead on improving syntactic agreement directly. In experiments, we find that though we obviously lose fidelity to more linguistically informed treebank annotation guidelines, this transformation-based approach yields the strongest improvements in syntactic machine translation.

Professor Dan Klein
Dissertation Committee Chair

To Deven and Priya, who remind me daily that some NPs are more important than others.

Contents

Contents ii

Acknowledgements v

1 Introduction 1

1.1 Statistical Machine Translation 1
1.2 Translation Rules 2
1.3 Intermediate Annotations 4
1.4 The Trouble with Multiple Annotations 6
1.5 Contributions of this Thesis 8

2 Experimental Setup 9

2.1 Corpora 9
2.1.1 Monolingual Treebanks 10
2.1.2 Parallel Corpora 10
2.1.3 Parallel Treebanks 11
2.2 Parsing and Word Alignment Experiments 12
2.3 Machine Translation Experiments 14
2.3.1 Training Parsing Models for MT 14

3 Bilingual Parsing 16

3.1 Model 19
3.2 Features 19
3.2.1 Monolingual Features 19
3.2.2 Word Alignment Features 21

3.2.3 Tree Structure Features 23
3.2.4 Typed vs untyped features 23
3.3 Training 24
3.3.1 Viterbi Alignments 24
3.3.2 Pseudo-gold Trees 26
3.3.3 Training Set Pruning 26
3.4 Joint Selection 27
3.4.1 Test Set Pruning 27
3.5 Experiments 28
3.5.1 Feature Ablation 28
3.5.2 Training Set Pruning 29
3.5.3 Test Set Pruning 31
3.5.4 Sensitivity to k 31
3.5.5 Parsing Results 31
3.5.6 Machine Translation Results 32
3.5.7 Analysis 32

4 Joint Parsing and Alignment **34**
4.1 Task Definition 36
4.2 Weakly Synchronized Grammars 38
4.3 Features 40
4.3.1 Parsing 40
4.3.2 Alignment 40
4.3.3 Synchronization 41
4.4 Learning 41
4.5 Mean Field Inference 43
4.5.1 Pruning 45
4.6 Testing 46
4.7 Experiments 47
4.7.1 Dataset-specific ITG Terminals 48
4.7.2 Parsing and Word Alignment 49
4.7.3 Machine Translation 50

4.7.4 Analysis 50

5 Transforming Trees **52**

5.1 Agreement 55

5.1.1 Relation to MT Performance 57

5.2 Transformation-Based Learning 57

5.3 Tree Transformations 58

5.4 Results and Analysis 63

5.5 Machine Translation 68

5.5.1 Binarization 70

5.5.2 Comparison with Supervised Parsing Models 71

6 Conclusions **73**

6.1 Future Directions 74

6.2 Impact 76

Bibliography **77**

Acknowledgements

There are many people to thank, but obviously the largest contributor to the work I've completed over my last five years at Berkeley is my advisor, Dan Klein, so I believe I'll start with him. Looking back, I really am kind of amazed at the huge stroke of luck that resulted in me working with Dan. When I came to Berkeley, I knew I would be an AI researcher, but beyond that I had nothing more than a vague interest in language, and approximately zero knowledge of contemporary NLP research. Through Dan, I learned everything that's made me the researcher I am today, from technical and historical background, to how to formulate an NLP problem, to the right way of quickly evaluating (and often discarding) research ideas, to his beautifully polished technique for writing up and presenting results. Along the way, Dan even expanded his *own* research agenda to indulge the half-baked impulse some of us had to advance the state-of-the-art in video game AI. Beyond being a better research mentor than I had any right to hope for, Dan has also been instrumental in developing my love of teaching and my success as a teacher, both through direct instruction and by demonstrating what excellent teaching really looks like. I've been quite priveliged to have Dan as an advisor, and the last five years wouldn't have been the same without him.

Of course, one of the other benefits of having an excellent advisor, is that he tends to attract other excellent students, many of whom I've been fortunate enough to learn from directly. John DeNero taught me most of what I know about how machine translation works and everything I know about how to explain it to other people. John Blitzer is more or less responsible for my ability to use the mean field approximation that shows up in this thesis. Without the guidance and direct assistance of Adam Pauls, it is likely that I would never have been able to actually run a practical machine translation experiment. David Hall is my one-stop reference for the inner workings of anything I use but don't understand, from Java HotSpot optimizations, to more esoteric variational inference schemes. He also was the only one who stuck with me when everybody else (understandably) abandoned our

StarCraft project for less-frustrating pursuits. Beyond the people already mentioned, I've also had chances to work directly with Slav Petrov and Taylor Berg-Kirkpatrick, either of whom I highly recommend to anyone who wants an object lesson in being hugely productive in a ridiculously short amount of time. And even just sitting around and talking with Percy Liang, Alexandre Bouchard-Côté, Aria Haghighi, Mohit Bansal, Dave Golland, Greg Durrett, and Jonathan Kummerfeld has been invaluable in helping me to work out thornier issues in my work and in reinvigorating my interest in NLP questions outside my own narrow corner of the field. I've often told people that one of the simultaneous advantages and disadvantages of being at Berkeley is that you're constantly surrounded by people who love the same things you do, but they're smarter and better at it than you are. It's a very educational and humbling experience, and I am much the better for it.

While at Berkeley, I've been lucky enough to have some truly excellent teachers beyond just my advisor. Martin Wainwright and Line Mikkelsen are both simply amazing classroom instructors, who taught me just about everything I've thus far needed to know about statistics and linguistics, respectively. Tom Griffiths' seminar on computational models of cognition was probably the most fun class I took here, and he later became a collaborator as we wrote up the work that started out as my class project. I've also received great mentorship from academics outside of Berkeley. Conversations with David Chiang, Bob Moore, and Jason Eisner have helped shape the way I think about my specific research questions and about NLP in general. External collaborators, first at SRI, then later at BBN have kept me constantly in mind of how to turn interesting results into practically significant ones.

Finally, I would be incredibly remiss if I did not thank my family for the support they've provided through this process. Deven Burkett was born about two weeks before I submitted the final version of the paper that became the first major contribution of this thesis. He has been there through most of this time, as a source of joy and of constructive commentary on figure design ("I like that rainbow, Daddy!"). Even more supportive and patient has been my wife, Priya Shimpi. Despite having already completed her own PhD and being well aware of the absurdity of the process, she never stopped providing love and encouragement. Thanks also to the rest of my family, who were kind enough to cheer me on without inquiring too loudly about why I'm 30 years old and still not out of school yet.

It's been an amazing five years, and it's impossible to name everybody who's influenced me in that time. So finally, one big thank you to all those I've known while at Berkeley and whose presence in my life has enriched it.

Chapter 1

Introduction

Statistical machine translation (MT) is one of the most active areas of modern natural language processing research, and an increasingly large number of statistical MT systems make use of linguistic tree structure annotations. The focus of this thesis is on improving these annotations as a means of affecting downstream translation quality. In this chapter, I'll give a high-level description of how syntactic MT systems typically work and survey alternative solutions to problems introduced by reliance on tree structure annotations. Chapter 2 will describe the specific datasets and experimental setup for the experiments in this thesis. Novel contributions begin in Chapter 3.

1.1 Statistical Machine Translation

The task of a machine translation system is to take a document in some source language, and translate it into coherent target language text with the same meaning. At a very high level, statistical MT systems work by searching through a database of known human translations to find the correct translation of each piece of the source document. These translation fragments are then stitched together to form the final target language output.

Systems of this type are considered to be *statistical* in nature because the system's decisions about which translations to use and how to put them together are based primarily on statistics gathered from large *parallel corpora*: collections of documents that have been translated by humans between the source and target languages.

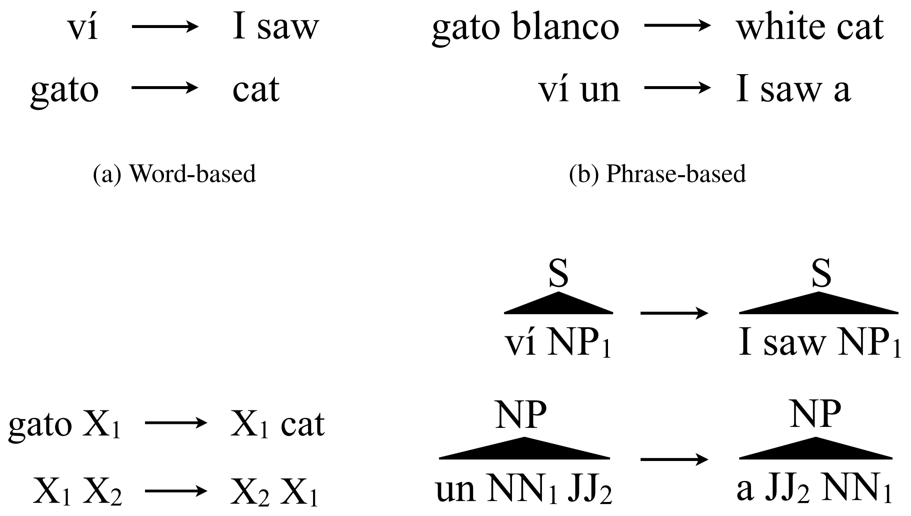
A standard statistical MT system is run as a pipeline with two stages: first, the system learns a database of *translation rules* from a parallel corpus. Second, the system uses these rules along with some decoding procedure to translate novel source language documents.[1] In this thesis, we will concentrate primarily on the initial rule extraction stage.

1.2 Translation Rules

Of course, before any translation rules can be discovered at all, it's necessary to decide what form they will take. Clearly, it would not be particularly effective to simply memorize translations of entire sentences, as it's vanishingly unlikely that all novel source sentences presented to the system will have been seen before. Thus, we need to operate at the level of smaller linguistic units. The earliest, simplest statistical MT systems learned translation rules for individual source language words, as in Figure 1.1a (Brown et al., 1994). However, due to syntactic divergence, it's rare for word-to-word translations to be accurate in even closely related languages, so word-level systems have since been supplanted by phrase-based ones, which operate at the level of multi-word sequences, as in Figure 1.1b (Koehn et al., 2003).

Phrase-based systems often achieve state-of-the-art performance for pairs of closely related languages, but don't do as well when the source and target languages have more divergent structure and long-range reorderings are required. One simple way to add more representational power to MT systems and help combat this problem is to augment phrase-

[1]If the decoder is parameterized, there is often also an intermediate stage where these parameters are tuned.



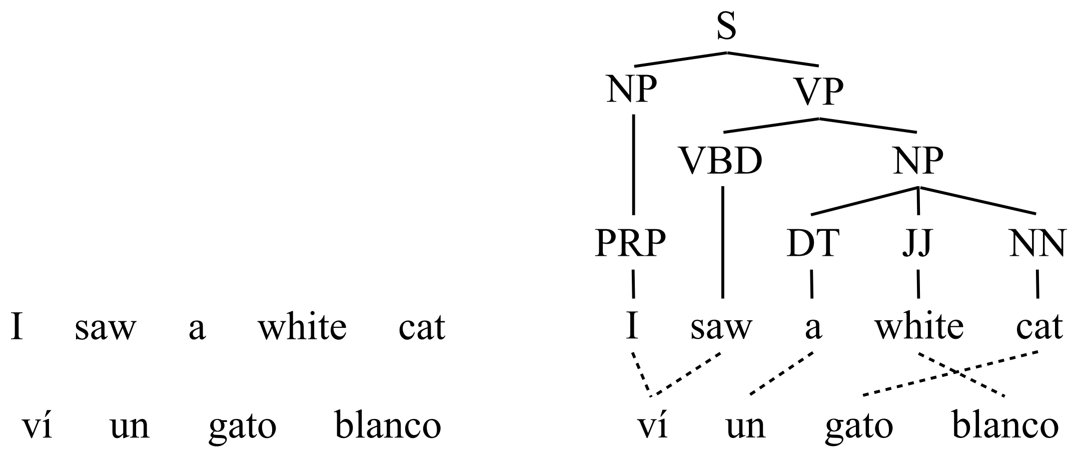
(a) Word-based (b) Phrase-based

(c) Hierarchical phrase-based (d) Syntax-based

Figure 1.1: Example translation rules.

based systems with hierarchical rules, such as those in Figure 1.1c (Chiang, 2007). Hierarchical rules contain coindexed gaps (marked as ‘X’) that can recursively be filled in by any other translation rule, including standard phrasal translations. For example, the second rule in Figure 1.1c is particularly powerful, allowing arbitrarily long phrases to swap places as they’re translated from the source to target languages.

Finally, hierarchical rules can additionally be augmented by replacing the generic ‘X’ nonterminals with syntactic categories. Syntactic rules are also often assigned top-level categories, resulting in rules such as those shown in Figure 1.1d (Galley et al., 2004). The first of these example rules translates simple Spanish sentences with the main verb ví (I saw) and an object NP that must be recursively filled in by subsequently plugging in another translation rule whose top-level category is ‘NP’. The second example explicitly encodes the adjective/noun reordering that’s usually required in Spanish to English translation; this is not really possible to the same degree in either fully lexicalized phrase-based rules or generically typed hierarchical ones.



(a) Raw sentence pair

(b) Annotated sentence pair

Figure 1.2: An example sentence pair, first unannoted, then annotated with word alignments and target-side syntax.

1.3 Intermediate Annotations

As mentioned earlier, MT systems discover translation rules by mining parallel corpora. Fortunately, there are several naturally occurring sources of human-translated text, such as news articles, transcriptions of government proceedings, technical manuals, or commercial websites, all of which are regularly published in multiple languages. Therefore, for many language pairs, parallel corpora are widely available. Unfortunately, though, translations are generally not marked with how they decompose into smaller units, such as the translation rules we wish to learn. For example, all the rules from Figure 1.1 can be derived by carving out fragments of the sentence pair in Figure 1.2a, but from the sentences alone, there's no way to know which fragments make up valid partial translations.

Typically, we simplify the problem slightly by assuming that our parallel corpora are aligned at the sentence level;[2] that is, each sentence in our data is part of a pair of sentences,

[2]Sometimes corpora of this sort are called *sentence-aligned* parallel corpora.

each of which is a direct translation of the other.[3] This still leaves us with two questions: first, which individual phrases within the sentence pair form valid pieces of the overall translation? Second, for systems that use syntactic information, what are the appropriate categories to assign to our translation fragments?

For almost all modern MT systems, these questions are answered by first annotating all the sentence pairs in the training corpus with additional information. Correspondences between fragments of sentence pairs are established via *word alignments*: mappings between individual words in each sentence. Word-based translation rules can be read directly from the word alignments, but more complicated rules are still typically found by applying simple surface heuristics to build longer aligned phrase pairs out of individual word-to-word mappings (Koehn et al., 2003).[4] For MT systems that use syntactic rules, the necessary linguistic information comes from *syntactic parse trees*. In Figure 1.2b we see the fully annotated sentence pair whose word alignments and English parse tree can be used to find all the translation rules in Figure 1.1.

In this thesis, we will be concerned primarily with MT systems that use syntactic translation rules. There are a wide variety of syntactic MT systems: some that extract translation rules using only target-language trees (Yamada and Knight, 2001; Galley et al., 2004; Zollmann et al., 2006; Shen et al., 2008), others that use only source-language trees (Quirk et al., 2005; Huang et al., 2006; Marton and Resnik, 2008), and still others that use both (Och et al., 2003; Aue et al., 2004; Ding and Palmer, 2005; Chiang, 2010). Because the MT experiments in this thesis were all carried out using Chinese to English syntactic translation rules in the style of Galley et al. (2004), our examples will tend to focus on target-side (English) syntax. However, it should be pointed out that the techniques presented here are equally applicable to all MT systems that rely on parse tree annotations.

[3]No distinction is made as to which sentence is the “original” and which was generated by a human translator.

[4]The notable exception to this general rule is the considerable body of work on models of direct phrase-to-phrase alignment (e.g. Marcu and Wong, 2002; Birch et al., 2006; Cherry and Lin, 2007; DeNero et al., 2008; Zhang et al., 2008; Blunsom et al., 2009; DeNero and Klein, 2010; Burkett and Klein, 2012a).

1.4 The Trouble with Multiple Annotations

When the set of translation rules that can be extracted from a sentence pair depends on both a word alignment and a syntactic parse tree, it is important that the two annotations agree on what consitutes a valid phrase. For example, in Figure 1.3, the English phrase "The first step" both has a coherent phrasal Chinese translation ("第一 步") and forms a constituent in the English parse tree. Thus, it is possible to extract a corresponding syntactic translation rule. However, lack of agreement between the parse tree and the word alignment is a frequently recurring problem. For example, in the same sentence pair, the VP in red cannot appear as a translation rule because the corresponding English phrase ("select team members") does not map nicely onto any Chinese phrase according to the word alignments. Conversely, there is a correspondence between the English phrase "to select" and the Chinese word "挑选", and a phrase-based system would extract that phrase pair, but since "to select" does not make up a constituent in the English parse tree, no simple syntactic rule can be extracted here either.[5]

There has been a large amount of previous work that attempts to address this problem of incompatibility between word alignments and syntactic annotations. Some systems attempt to directly model the divergence between source and target language syntax (Yamada and Knight, 2001; Eisner, 2003), but most recent work focuses on reducing the sensitivity of the rule-extraction procedure to the constituency decisions made by 1-best syntactic parsers, either by using forest-based methods for learning translation rules (Mi and Huang, 2008; Zhang et al., 2009), or by learning rules that encode syntactic information but do not strictly adhere to constituency boundaries (Zollmann et al., 2006; Marton and Resnik, 2008; Chiang, 2010).

Instead of making improvements to specific syntactic MT systems, the approach we

[5]It is possible to extract a more complicated rule for a VP that needs the object NP to be filled, but the simpler rule is more widely applicable and thus would also be independently useful.

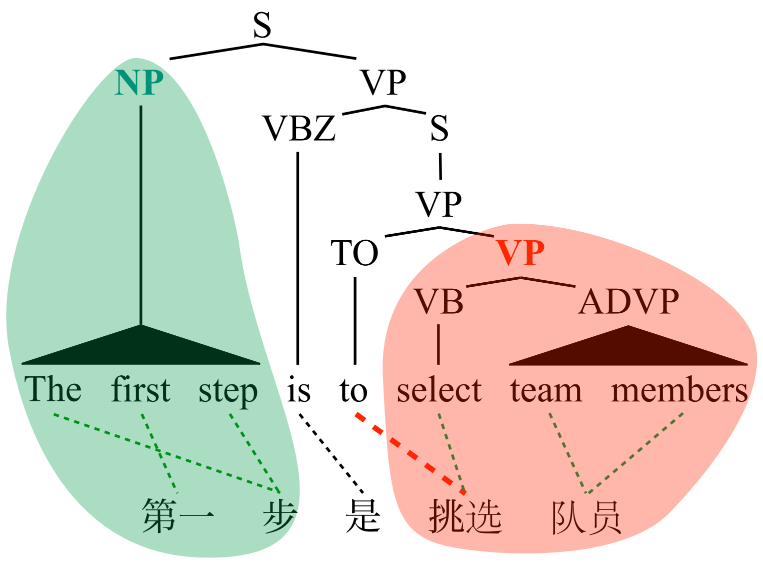


Figure 1.3: A simple annotated sentence pair in which the NP in green is extractable, but the VP in red and the aligned phrase pair whose English side is “to select” are not due to disagreement between the parse tree and the word alignment.

take in the present work is to directly modify the input annotations in order to improve downstream MT performance in a way that is independent of the specific syntactic MT system being used. There has also been previous work of this sort, particularly related to word alignments. DeNero and Klein (2007) built a word alignment model that conditioned on target-side syntax, while May and Knight (2007) actually used the output of a syntactic MT system to inform retraining of the word alignment model. Fossum et al. (2008) took a more direct approach, modifying the output of existing word aligners to agree more closely with the syntactic parse trees. In this thesis, rather than focusing on making word alignments more suitable for syntactic MT, we will instead concentrate primarily on modifying the syntactic annotations.

1.5 Contributions of this Thesis

The novel contributions of this thesis are in two parts. First, we take the stance that a large number of the disagreements between the parse trees and the word alignments may be due to parser errors. By leveraging bilingual information, we can improve the linguistic quality of syntactic parses and thereby hope to improve MT performance. Chapter 3 presents a simple bilingual parsing model that exploits this insight. In Chapter 4 we extend this approach to simultaneously improve the quality of our parses and word alignments.

However, improving the quality of the annotations may not be enough. For example, the sentence pair in Figure 1.3 was taken directly from a gold-standard dataset,[6] but still had areas where the parse and word alignment disagreed. Therefore, in Chapter 5, we describe a method for reanalyzing linguistic syntax to directly optimize an MT-related metric.

[6]The parse tree and word alignment were both human annotated.

Chapter 2

Experimental Setup

This thesis primarily contains two different types of experiments. The models in Chapters 3 and 4 are designed primarily to improve the intrinsic quality of parses and, in the case of the latter model, word alignments, as measured by their fidelity to gold-standard human annotations. Thus, both of those models are evaluated as parsers (and, in the second case, as a word aligner) by measuring parsing (and word alignment) F_1 on a human-annotated bilingual treebank. The technique described in Chapter 5 is designed to optimize a different metric, but the effects on this metric will also be evaluated on the same treebank. Of course, the motivating application for all the work presented here is machine translation, and thus all three techniques described are also evaluated in terms of their effects on downstream MT performance.

2.1 Corpora

The various techniques presented here rely on three types of data: monolingual treebanks, for training baseline monolingual parsers; parallel corpora, for training baseline un-

Corpus	Source	Documents	Sentences
English Training	Penn WSJ	Sections 2-21	40k
Chinese Training	Penn CTB	Articles 400-1151	15k

Table 2.1: Monolingual treebank corpus descriptions.

supervised word aligners and also for training and evaluating machine translation systems; and bilingual treebanks, for training and evaluating the bilingual parsing models.

2.1.1 Monolingual Treebanks

Since all of our experiments are centered around the task of Chinese to English translation, we require monolingual parsers for both these languages, both as evaluation baselines and to provide features for the bilingual parsing models. Although these parsers could, in principle, be trained on the monolingual projections of our bilingual treebank, this dataset is relatively small, and the performance of the parsers is improved dramatically by including additional monolingual training data. The English treebank used is the standard training set from the Penn Wall Street Journal Treebank (Marcus et al., 1993). For Chinese, we used the Penn Chinese Treebank v5.0 (Xue et al., 2005), excluding the first 325 articles (these appear in our bilingual treebank). Details are in Table 2.1.

2.1.2 Parallel Corpora

Machine translation systems generally use two separate parallel corpora: at training time, a large corpus is used for extracting translation rules and collecting statistics (usually rule counts), and then at test time, the system is evaluated on a smaller held-out corpus. Systems that need to set parameters (including the one used in our experiments) also require an additional held-out tuning corpus, typically of about the same size as the test set.

Our MT training corpus was a 22 million word mixed genre (though primarily

Corpus	Source	Sentences	English Words	Chinese Words
Training	Mixed	506k	11.2M	10.8M
Tuning	MT04+MT05	1000	24k	22k
Test	MT04+MT05	642	15k	14k

Table 2.2: Unannotated parallel corpora descriptions.

Corpus	Articles	Sentences	English Words	Chinese Words
Training	1-270	2261	69k	50k
Development	301-325	223	5.0k	3.9k
Test	271-300	265	7.6k	5.6k

Table 2.3: Bilingual treebank corpus descriptions.

newswire) corpus that had been previously assembled and processed to enforce a consistent tokenization scheme. The tuning and test sets were taken from the NIST MT04 and MT05 development sets, with some light processing to match the tokenization of the training data. All these datasets were received from BBN as part of the DARPA GALE (now BOLT) program. Details are in Table 2.2.

2.1.3 Parallel Treebanks

The novel models presented here all use a parallel bilingual treebank for both training and evaluation.[1] Our annotated parallel data all comes from the English Chinese Translation Treebank (Bies et al., 2007). This dataset consists of the documents from the Chinese Treebank v1.0 (Xia et al., 2000), which have all since been translated into English. In addition, the translated English sentences have been manually annotated with gold-standard parse trees using similar (though not quite identical) conventions as the Penn WSJ Treebank. Finally, this dataset was also later annotated with gold-standard word alignment information.

[1]The tree transformation system described in Chapter 5 doesn't actually require a bilingual treebank, but it is still useful to use one as a clean setting for qualitative assessment of the system.

Because the various components of this data set (English translations, English trees, gold word alignments) were released piecemeal, there is no publicly available cleanly sentence-aligned version of this corpus. Thus, we did have to perform some additional processing to make this data usable, throwing away all sentences that did not neatly align one-to-one[2] or for which one or more annotations were missing. This filtering step left a total of 2749 sentence pairs from the 4180 sentences in the original data. The data was separated into training, development, and test sets according to the standard Chinese treebank division. Details are in Table 2.3.

2.2 Parsing and Word Alignment Experiments

The models presented in Chapters 3 and 4 both use features derived from unsupervised word aligners and monolingual parsers. The unsupervised word aligner we used was the paired HMM included in the Berkeley Aligner (Liang et al., 2006). The aligner was trained on all the available parallel data except for the MT tuning and test sets (i.e. the training set from Table 2.2 plus all the corpora in Table 2.3). To train the HMMs, we initialized with 5 EM iterations of joint Model 1 training, followed by 5 EM iterations of independent HMM training. The two directional models were symmetrized using the soft union heuristic with competitive thresholding (DeNero and Klein, 2007). When needed, a "one-best" alignment was generated by thresholding the symmetrized posteriors at 0.33.

We used the Berkeley Parser (Petrov and Klein, 2007) for both our English and Chinese monolingual parsers. However, selecting the data to train the grammars for these parsers was a bit subtle. To maximize parser performance, we naturally wished to make use of all available annotated data. Thus, at test time, we used grammars trained on each language's respective full monolingual training set *plus* the appropriate monolingual projection of the

[2]Many of the translations were of multiple English sentences translated from a single Chinese sentence, or more rarely, the other way around.

bilingual training set.[3] However, at training time, it was important that the monolingual grammars not be trained on any of the bilingual training data – otherwise, the monolingual parsing features would be too powerful, and there would be no way to learn appropriate weights for the remaining features in the bilingual model. Thus, when training the bilingual models, we used monolingual grammars trained on less data.

For Chinese, we simply omitted the entire bilingual corpus, using only the monolingual training data. For English, we had to be more careful, as the monolingual corpus (WSJ) was out-of-domain relative to the test data, and thus it was fairly important that *some* in-domain data be included when training the monolingual grammars. We solved this issue with a very simple jackknifing approach. We trained two English grammars: one that appended the first half of the English projection of the bilingual training corpus to the monolingual WSJ corpus, and one that appended the second half. Then, when computing monolingual English parser model scores for sentences in the training set, we used the grammar that did *not* include the current sentence.

The final parsing evaluation numbers we report are all parsing F_1 values on the test set from Table 2.3.[4] We report individual scores on both English and Chinese, as well as an aggregate "Total" score obtained by just measuring overall F_1 on a single collection of trees that includes all results from both languages. Note that since most of the training data for the monolingual English grammar comes from the out-of-domain WSJ corpus, Chinese scores tend to be higher than English ones in all the experiments reported in this thesis.

In Chapter 4, we describe a model that improves both parsing and word alignment performance. Since we thus wish to measure the intrinsic quality of the improved word alignments, we also report word alignment AER and F_1 on the same test set.

[3]In Chinese, this corresponded to the full standard training set from the Chinese Treebank v5.0. In English, this augments the standard WSJ training set with an additional 2200 sentences.

[4]Chapter 3 also contains development set results that were obtained with a slightly different experimental setup. The setup was similar, but the baseline parsers used during development were weaker, so the dev set results aren't directly comparable to the final test results.

2.3 Machine Translation Experiments

All our machine translation experiments were carried out from Chinese to English, using the syntactic tree-to-string pipeline included in the latest release of Moses (Koehn et al., 2007). This pipeline requires target-side parse trees, so we automatically annotated the training data with English parses and with word alignments. The baseline setup uses parse trees from a monolingual English parser and word alignments from the same HMM word aligner described in Section 2.2. The only differences between our various MT conditions were in the annotations. Typically, only the English parse trees changed, although we did also swap in different word alignments for experiments that involved the joint parsing and word alignment model.

We trained, tuned, and tested the MT system using the data in Table 2.2. The evaluation metric we used for both tuning and testing was multi-reference BLEU (Papineni et al., 2001).[5] We tuned parameters with MERT (Och, 2003). However, since MERT is known to be unstable (Clark et al., 2011), we ran MERT 10 times for each data condition; the scores reported here are averages taken across these 10 runs.

2.3.1 Training Parsing Models for MT

One seldom-discussed yet fairly important issue in syntactic MT is tokenization. Both parsers and MT systems are very sensitive to tokenization conditions, but the datasets used for these tasks typically have very different tokenization schemes. In order to deal with this concern in our experiments, we modified the treebanks in Tables 2.1 and 2.3 as follows:

- The surface word yields of the individual sentences in each treebank were retokenized according to the same guidelines as the MT corpora using a tokenizer obtained from BBN.

[5]The tuning and test corpora each had four English references per sentence pair.

- The gold parse trees were coerced to match the new tokenization using Procrustes, a best-fit tree retokenization tool.[6]

- For the bilingual treebanks, the gold word alignments were also transformed to match the new tokenization by duplicating or collapsing alignment links as tokens were split or merged, respectively.

All parsing models (both the monolingual baselines and the bilingual models) were retrained on the retokenized treebanks before they were used to annotate the MT training data.

[6]Thanks to David Chiang for sharing this software.

Chapter 3

Bilingual Parsing

As discussed in Chapter 1, methods for machine translation (MT) have increasingly leveraged not only the formal machinery of syntax (Wu, 1997; Chiang, 2007; Zhang et al., 2008), but also linguistic tree structures that rely on automatic parsing of one or both sides of input bitexts and are therefore impacted by parser quality. Unfortunately, parsing general bitexts well can be a challenge for newswire-trained treebank parsers for many reasons, including out-of-domain input and tokenization issues.

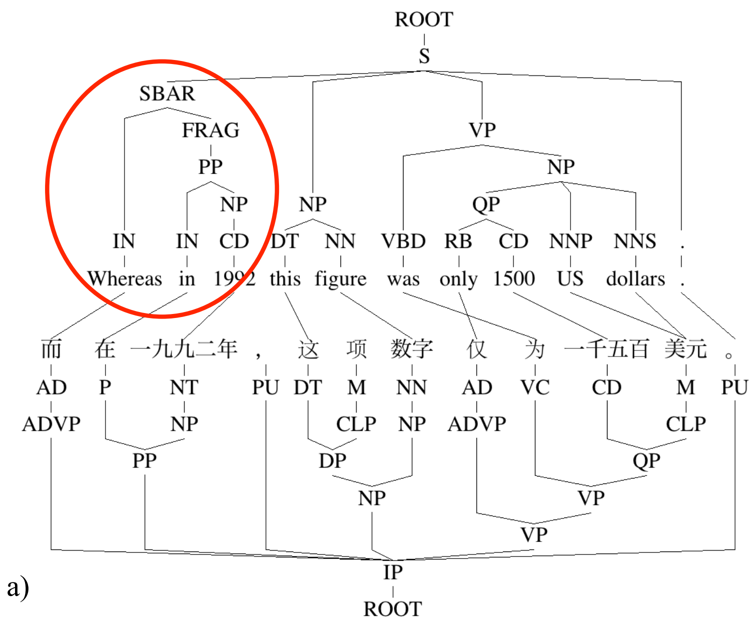
On the other hand, the presence of translation pairs offers a new source of information: bilingual constraints. For example, Figure 3.1 shows a case where a state-of-the-art English parser (Petrov and Klein, 2007) has chosen an incorrect structure which is incompatible with the (correctly chosen) output of a comparable Chinese parser. Previous work has shown that such bilingual constraints can be leveraged to transfer parse quality from a resource-rich language to a resource-impoverished one, either via joint modeling (Smith and Smith, 2004), or direct projection (Hwa et al., 2005). In the (nowadays relatively rare) case where no annotated data is available at all, multilingual constraints can also improve the performance of unsupervised grammar induction (Snyder et al., 2009), sometimes even in the absence of bitexts (Berg-Kirkpatrick and Klein, 2010). In this chapter, we will show that bilingual constraints and reinforcement can also be leveraged to enhance state-of-art parsing performance, substantially improving parses on both sides of a bitext even for two resource-rich languages.[1]

Formally, we present a log-linear model over triples of source trees, target trees, and node-to-node tree alignments between them. We consider a set of core features which capture the scores of monolingual parsers as well as measures of syntactic alignment. Our model conditions on the input sentence pair and so features can and do reference input characteristics such as posterior distributions from a word-level aligner (Liang et al., 2006).

At training time, correct trees are observed on both the source and target side. However, gold tree alignments are not present and so are induced as latent variables using an iterative training procedure. To make the process efficient and modular to existing monolingual parsers, we introduce several approximations: use of k-best lists in candidate generation, an adaptive bound to avoid considering all k^2 combinations, and Viterbi approximations to alignment posteriors.

When evaluating this model, we find that joint parse selection improves the English trees by 2.6 F_1 and the Chinese trees by 2.4 F_1. We also find that when using the improved English trees in downstream syntactic MT, there is a modest but consistent improvement of around 0.3 BLEU points over rules extracted from a monolingual English parser. In sum, leveraging bilingual information is very effective at improving parse quality, and does so in a way that starts us moving forwards towards our original goal of improving syntactic MT performance.

[1]The model described in this chapter was originally presented at EMNLP in 2008 (Burkett and Klein, 2008).


a)
ROOT
S
SBAR
FRAG
PP
NP
VP
NP
NP
QP
IN IN CD DT NN VBD RB CD NNP NNS .
Whereas in 1992 this figure was only 1500 US dollars .
而 在 一九九二年 ， 这 项 数字 仅 为 一千五百 美元 。
AD P NT PU DT M NN AD VC CD M PU
ADVP NP CLP NP ADVP CLP
PP DP QP
NP VP
VP
IP
ROOT

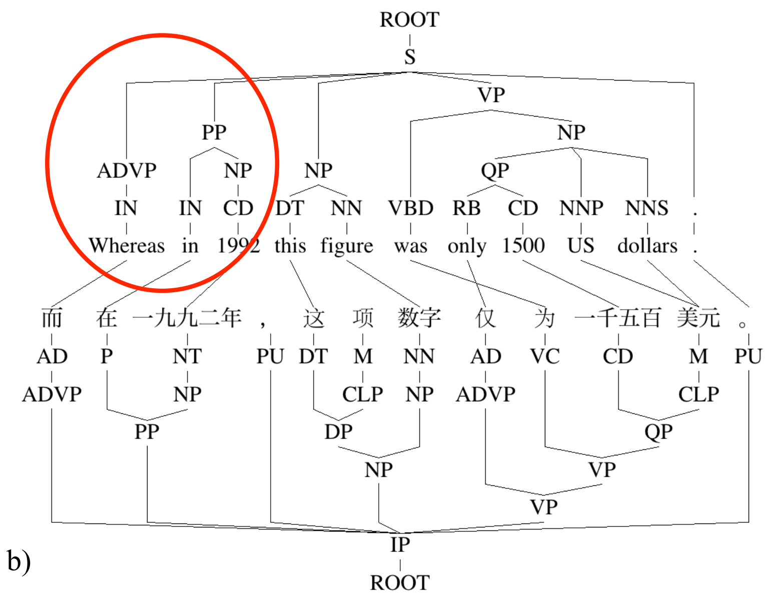

b)
ROOT
S
VP
PP
NP
ADVP NP NP QP
IN IN CD DT NN VBD RB CD NNP NNS .
Whereas in 1992 this figure was only 1500 US dollars .
而 在 一九九二年 ， 这 项 数字 仅 为 一千五百 美元 。
AD P NT PU DT M NN AD VC CD M PU
ADVP NP CLP NP ADVP CLP
PP DP QP
NP VP
VP
IP
ROOT

Figure 3.1: Two possible parse pairs for a Chinese-English sentence pair. The parses in a) are chosen by independent monolingual statistical parsers, but only the Chinese side is correct. The gold English parse shown in b) is further down in the 100-best list, despite being more consistent with the gold Chinese parse. The circles show where the two parses differ. Note that in b), the ADVP and PP nodes correspond nicely to Chinese tree nodes, whereas the correspondence for nodes in a), particularly the SBAR node, is less clear.

3.1 Model

In this model, we consider pairs of sentences (s, s'), where we use the convention that unprimed variables are source domain and primed variables are target domain. These sentences have parse trees t (respectively t') taken from candidate sets T (T'). Non-terminal nodes in trees will be denoted by n (n') and we abuse notation by equating trees with their node sets. Alignments a are simply at-most-one-to-one matchings between a pair of trees t and t' (see Figure 3.2a for an example). Note that we will also mention *word* alignments in feature definitions; a and the unqualified term *alignment* will always refer to node alignments. Words in a sentence are denoted by v (v').

Our model is a general log-linear (maximum entropy) distribution over triples (t, a, t') for sentence pairs (s, s'):

$$\mathrm{P}(t, a, t|s, s') \propto \exp(\theta^{\top}\phi(t, a, t'))$$

Features are thus defined over (t, a, t') triples; we discuss specific features below.

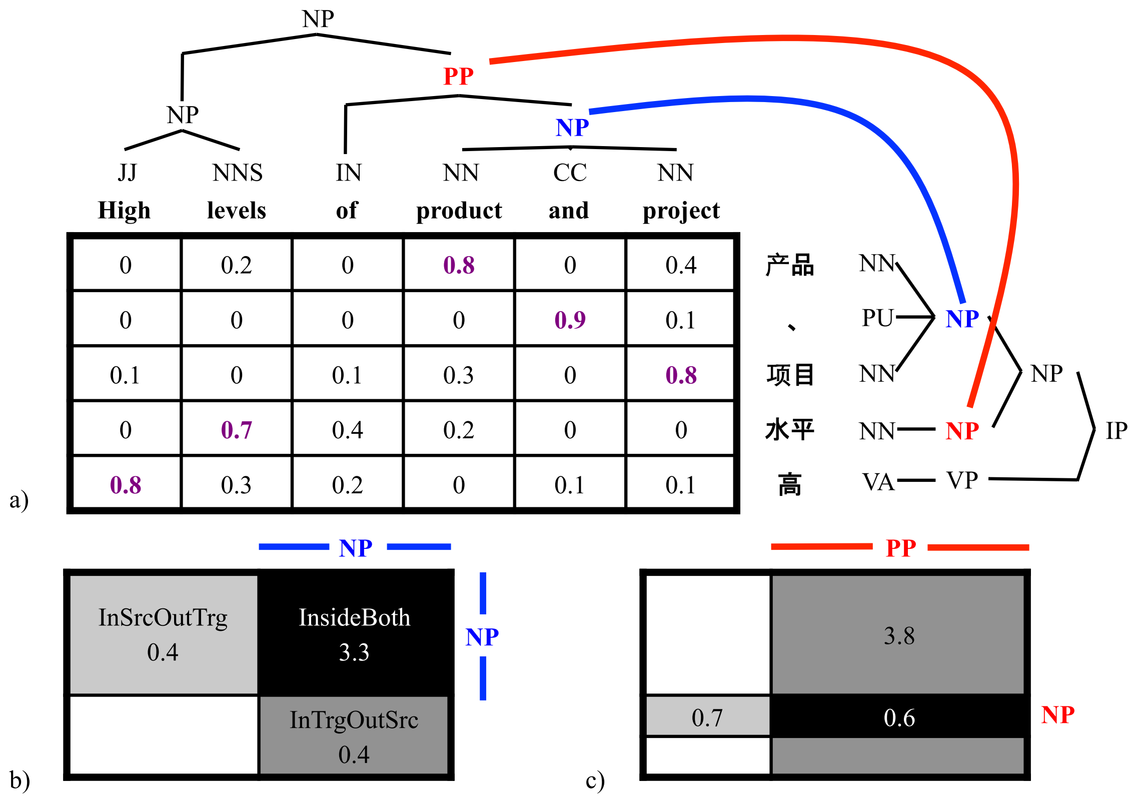
3.2 Features

To use this model, we need features of a triple (t, a, t') which encode both the monolingual quality of the trees as well as the quality of the alignment between them. We introduce a variety of features in the next sections.

3.2.1 Monolingual Features

To capture basic monolingual parse quality, we begin with a single source and a single target feature whose values are the log likelihood of the source tree t and the target tree t', respectively, as given by our baseline monolingual parsers. These two features are called

SOURCELL and TARGETLL respectively. It is certainly possible to augment these simple features with what would amount to monolingual reranking features, but we do not explore that option here. Note that with only these two features, little can be learned: all positive weights θ cause the jointly optimal parse pair (t, t') to comprise the two top-1 monolingual outputs (the baseline).



	High	levels	of	product	and	project
产品	0	0.2	0	**0.8**	0	0.4
、	0	0	0	0	**0.9**	0.1
项目	0.1	0	0.1	0.3	0	**0.8**
水平	0	**0.7**	0.4	0.2	0	0
高	**0.8**	0.3	0.2	0	0.1	0.1

Figure 3.2: a) An example of a legal alignment on a Chinese-English sentence fragment with one good and one bad node pair, along with sample word alignment posteriors. Hard word alignments are bolded. b) The word alignment regions for the good NP-NP alignment. InsideBoth regions are shaded in black, InSrcOutTrg in light grey, and InTrgOutSrc in grey. c) The word alignment regions for the bad PP-NP alignment.

3.2.2 Word Alignment Features

All other features in this model reference the entire triple (t, a, t'). We will define such features over aligned node pairs for efficiency, but generalizations are certainly possible.

Bias: The first feature is simply a bias feature which has value 1 on each aligned node pair (n, n'). This bias allows the model to learn a general preference for denser alignments.

Alignment features: Of course, some alignments are better than others. One indicator of a good node-to-node alignment between n and n' is that a good word alignment model thinks that there are many word-to-word alignments in their bispan. Similarly, there should be few alignments that violate that bispan. To compute such features, we define $a(v, v')$ to be the posterior probability assigned to the word alignment between v and v' by an independent word aligner.[2]

Before defining alignment features, we need to define some additional variables. For any node $n \in t$ ($n' \in t'$), the inside span $i(n)$ ($i(n')$) comprises the input tokens of s (s') dominated by that node. Similarly, the complement, the outside span, will be denoted $o(n)$ ($o(n')$), and comprises the tokens not dominated by that node. See Figure 3.2b,c for examples of the resulting regions.

$$\begin{aligned}
\textsc{InsideBoth} &= \sum_{v \in i(n)} \sum_{v' \in i(n')} a(v, v') \\
\textsc{InSrcOutTrg} &= \sum_{v \in i(n)} \sum_{v' \in o(n')} a(v, v') \\
\textsc{InTrgOutSrc} &= \sum_{v \in o(n)} \sum_{v' \in i(n')} a(v, v')
\end{aligned}$$

Hard alignment features: We also define the hard versions of these features, which

[2]It is of course possible to learn good alignments using lexical indicator functions or other direct techniques, but given our very limited training data, it is advantageous to leverage counts from an unsupervised alignment system.

take counts from the word aligner's hard top-1 alignment output δ:

$$\begin{aligned}
\textsc{HardInsideBoth} &= \sum_{v \in i(n)} \sum_{v' \in i(n')} \delta(v, v') \\
\textsc{HardInSrcOutTrg} &= \sum_{v \in i(n)} \sum_{v' \in o(n')} \delta(v, v') \\
\textsc{HardInTrgOutSrc} &= \sum_{v \in o(n)} \sum_{v' \in i(n')} \delta(v, v')
\end{aligned}$$

Scaled alignment features: Finally, undesirable larger bispans can be relatively sparse at the word alignment level, yet still contain many good word alignments simply by virtue of being large. We therefore define a scaled count which measures density rather than totals. The geometric mean of span lengths was a superior measure of bispan "area" than the true area because word-level alignments tend to be broadly one-to-one in our word alignment model.

$$\begin{aligned}
\textsc{ScaledInsideBoth} &= \frac{\textsc{InsideBoth}}{\sqrt{|i(n)| \cdot |i(n')|}} \\
\textsc{ScaledInSrcOutTrg} &= \frac{\textsc{InSrcOutTrg}}{\sqrt{|i(n)| \cdot |o(n')|}} \\
\textsc{ScaledInTrgOutSrc} &= \frac{\textsc{InTrgOutSrc}}{\sqrt{|o(n)| \cdot |i(n')|}}
\end{aligned}$$

Head word alignment features: When considering a node pair (n, n'), especially one which dominates a large area, the above measures treat all spanned words as equally important. However, lexical heads are generally more representative than other spanned words. Let h select the headword of a node according to standard head percolation rules (Collins, 2003; Bikel and Chiang, 2000).

$$\begin{aligned}
\textsc{AlignHeadWord} &= a(h(n), h(n')) \\
\textsc{HardAlignHeadWord} &= \delta(h(n), h(n'))
\end{aligned}$$

3.2.3 Tree Structure Features

We also consider features that measure correspondences between the tree structures themselves.

Span difference: We expect that, in general, aligned nodes should dominate spans of roughly the same length, and so we allow the model to learn to penalize node pairs whose inside span lengths differ greatly.

$$\textsc{SpanDiff} \;=\; ||i(n)| - |i(n')||$$

Number of children: We also expect that there will be correspondences between the rules of the CFGs that generate the trees in each language. To encode some of this information, we compute indicators of the number of children c that the nodes have in t and t'.

$$\textsc{NumChildren}\langle |c(n)|, |c(n')| \rangle \;=\; 1$$

Child labels: In addition, we also encode whether certain label pairs occur as children of matched nodes. Let $c(n, \ell)$ select the children of n with label ℓ.

$$\textsc{ChildLabel}\langle \ell, \ell' \rangle \;=\; |c(n, \ell)| \cdot |c(n', \ell')|$$

Note that the corresponding "self labels" feature is not listed because it arises in the next section as a typed variant of the bias feature.

3.2.4 Typed vs untyped features

For each feature above (except monolingual features), we create label-specific versions by conjoining the label pair $(\ell(n), \ell(n'))$. We use both the typed and untyped variants of all features.

3.3 Training

Recall that our data condition supplies sentence pairs (s, s') along with gold parse pairs (g, g'). We do not observe the alignments a which link these parses. In principle, we want to find weights which maximize the marginal log likelihood of what we do observe given our sentence pairs:[3]

$$\begin{aligned} \theta^* &= \operatorname*{argmax}_{\theta} \sum_{a} \mathrm{P}(g, a, g' | s, s', \theta) && (3.1) \\ &= \operatorname*{argmax}_{\theta} \frac{\sum_a exp(\theta^\top \phi(g, a, g'))}{\sum_{(t,t')} \sum_a exp(\theta^\top \phi(t, a, t'))} && (3.2) \end{aligned}$$

There are several challenges. First, the space of symmetric at-most-one-to-one matchings is #P-hard to sum over exactly (Valiant, 1979). Second, even without matchings to worry about, standard methods for maximizing the above formulation would require summation over pairs of trees, and we want to assume a fairly generic interface to independent monolingual parsers. As we have therefore chosen to operate in a reranking mode over monolingual k-best lists, we now have another issue: our k-best outputs on the data which trains this model may not include the gold tree pair. We therefore make several approximations and modifications, which we discuss in turn.

3.3.1 Viterbi Alignments

Because summing over alignments a is intractable, we cannot evaluate (3.2) or its derivatives. However, if we restrict the space of possible alignments, then we can make this optimization more feasible. One way to do this is to stipulate in advance that for each tree pair, there is a canonical alignment $a_0(t, t')$. Of course, we want a_0 to reflect actual correspondences between t and t', so we want a reasonable definition that ensures the alignments are of reasonable quality. Fortunately, it turns out that we can efficiently optimize a

[3]In this presentation, we only consider a single sentence pair for the sake of clarity, but our true objective was multiplied over all sentence pairs in the training data.

given a fixed tree pair and weight vector:

$$\begin{aligned} a^* &= \operatorname*{argmax}_a \mathrm{P}(a|t, t', s, s', \theta) \\ &= \operatorname*{argmax}_a \mathrm{P}(t, a, t'|s, s', \theta) \\ &= \operatorname*{argmax}_a \exp(\theta^\top \phi(t, a, t')) \end{aligned}$$

This optimization requires only that we search for an optimal alignment. Because all our features can be factored to individual node pairs, this can be done with the Hungarian algorithm in cubic time.[4] Note that we do not enforce any kind of domination consistency in the matching: for example, the optimal alignment might in principle have the source root aligning to a target non-root and vice versa.

We then define $a_0(t, t')$ as the alignment that maximizes $w_0^\top \phi(t, a, t')$, where w_0 is a fixed initial weight vector with a weight of 1 for INSIDEBOTH, -1 for INSRCOUTTRG and INTRGOUTSRC, and 0 for all other features. Then, we simplify (3.2) by fixing the alignments a_0:

$$\theta^* = \operatorname*{argmax}_\theta \frac{exp(\theta^\top \phi(g, a_0(g, g'), g'))}{\sum_{(t,t')} exp(\theta^\top \phi(t, a_0(t, t'), t'))} \tag{3.3}$$

This optimization has no latent variables and is therefore convex and straightforward. However, while we did use this as a rapid training procedure during development, fixing the alignments a priori is both unsatisfying and also less effective than a procedure which allows the alignments a to adapt during training.

Again, for fixed alignments a, optimizing θ is easy. Similarly, with a fixed θ, finding the optimal a for any particular tree pair is also easy. Another option is therefore to use an iterative procedure that alternates between choosing optimal alignments for a fixed θ, and then reoptimizing θ for those fixed alignments according to (3.3). By iterating, we perform

[4]There is a minor modification to allow nodes not to match. Any alignment link which has negative score is replaced by a zero-score link, and any zero-score link in the solution is considered a pair of unmatched nodes.

the following optimization:

$$\theta^* = \operatorname*{argmax}_{\theta} \frac{\max_a exp(\theta^\top \phi(g, a, g'))}{\sum_{(t,t')} \max_a exp(\theta^\top \phi(t, a, t'))} \tag{3.4}$$

Note that (3.4) is just (3.2) with summation replaced by maximization. Though we do not know of any guarantees for this EM-like algorithm, in practice it converges after a few iterations given sufficient training data. We initialize the procedure by setting w_0 as defined above.

3.3.2 Pseudo-gold Trees

When training this model, we approximate the sets of all trees with k-best lists, T and T', produced by monolingual parsers. Since these sets are not guaranteed to contain the gold trees g and g', our next approximation is to define a set of *pseudo-gold trees*, following previous work in monolingual parse reranking (Charniak and Johnson, 2005). We define $\hat{T}$ ($\hat{T}'$) as the F_1-optimal subset of T (T'). We then modify (3.4) to reflect the fact that we are seeking to maximize the likelihood of trees in this subset:

$$\theta^* = \operatorname*{argmax}_{\theta} \sum_{(t,t') \in (\hat{T}, \hat{T}')} \mathrm{P}(t, t' | s, s', \theta) \tag{3.5}$$

where

$$\mathrm{P}(t, t' | s, s', \theta) = \frac{\max_a \exp(\theta^\top \phi(t, a, t'))}{\sum_{(\bar{t}, \bar{t}') \in (T, T')} \max_a \exp(\theta^\top \phi(\bar{t}, a, \bar{t}'))} \tag{3.6}$$

3.3.3 Training Set Pruning

To reduce the time and space requirements for training, we do not always use the full k-best lists. To prune the set T, we rank all the trees in T from 1 to k, according to their log likelihood under the baseline parsing model, and find the rank of the least likely pseudo-gold tree:

$$r^* = \min_{t \in \hat{T}} rank(t)$$

Finally, we restrict T based on rank:

$$T_{pruned} = \{t \in T | rank(t) \leq r^* + \epsilon\}$$

where ϵ is a free parameter of the pruning procedure. The restricted set T'_{pruned} is constructed in the same way. When training, we replace the sum over all tree pairs in (T, T') in the denominator of (3.6) with a sum over all tree pairs in $(T_{pruned}, T'_{pruned})$.

The parameter ϵ can be set to any value from 0 to k, with lower values resulting in more efficient training, and higher values resulting in better performance. We set ϵ by empirically determining a good speed/performance tradeoff (see Section 3.5.2).

3.4 Joint Selection

At test time, we have a weight vector θ and so selecting optimal trees for the sentence pair (s, s') from a pair of k best lists, (T, T') is straightforward. We just find:

$$\begin{aligned} (t^*, t'^*) &= \operatorname*{argmax}_{(t,t') \in (T,T')} \max_a \mathrm{P}(t, a, t' | s, s', \theta) \\ &= \operatorname*{argmax}_{(t,t') \in (T,T')} \max_a \theta^\top \phi(t, a, t') \end{aligned}$$

Note that with no additional cost, we can also find the optimal alignment between t^* and t'^*:

$$a^* = \operatorname*{argmax}_a \theta^\top \phi(t^*, a, t'^*)$$

3.4.1 Test Set Pruning

Because the size of (T, T') grows as $O(k^2)$, the time spent iterating through all these tree pairs can grow unreasonably long, particularly when reranking a set of sentence pairs the size of a typical MT corpus. To combat this, we use a simple pruning technique to limit the number of tree pairs under consideration.

To prune the list of tree pairs, first we rank them according to the metric:

$$w_{\text{SOURCELL}} \cdot \text{SOURCELL} + w_{\text{TARGETLL}} \cdot \text{TARGETLL}$$

Then, we simply remove all tree pairs whose ranking falls below some empirically determined cutoff. As we show in Section 3.5.3, by using this technique we are able to speed up reranking by a factor of almost 20 without an appreciable loss of performance.

3.5 Experiments

The general experimental setup for evaluating this model's parsing accuracy is described in Section 2.2. For the final evaluation, the model was trained as described in Section 3.3, but since the full iterative training procedure was relatively time-consuming, during development of the model, we had two training setups: rapid and full. In the rapid training setup, only 1000 sentence pairs from the training set were used, and we used fixed alignments for each tree pair rather than iterating (see Section 3.3.1). The full training setup used the iterative training procedure on all 2261 training sentence pairs.[5]

Unless otherwise specified, the maximum value of k was set to 100 for both training and testing, and all experiments used a value of 25 as the ϵ parameter for training set pruning and a cutoff rank of 500 for test set pruning.

3.5.1 Feature Ablation

To verify that all our features were contributing to the model's performance, we did an ablation study, removing one group of features at a time. Table 3.1 shows the F_1 scores

[5]As mentioned in Section 2.2, please note that the development set results in Sections 3.5.1, 3.5.2, 3.5.3, and 3.5.4 all used a slightly different training setup with different baseline grammars. While the dev set results are all internally consistent, they are not directly comparable to those in Section 3.5.5; differences reflect more than just inherent differences between the development and test sets.

	Baseline Parsers		
Features	Chinese F_1	English F_1	Total F_1
Monolingual	84.95	76.75	81.15
	Rapid Training		
Features	Chinese F_1	English F_1	Total F_1
All	86.37	78.92	82.91
−Hard align	85.83	77.92	82.16
−Scaled align	86.21	78.62	82.69
−Head word	**86.47**	**79.00**	**83.00**
−Span diff	86.00	77.49	82.07
−Num children	86.26	78.56	82.69
−Child labels	86.35	78.45	82.68
	Full Training		
Features	Chinese F_1	English F_1	Total F_1
All	**86.76**	79.41	**83.34**
−Head word	86.42	**79.53**	83.22

Table 3.1: Feature ablation study. F_1 on dev set after training with individual feature groups removed. Performance with individual baseline parsers is included for reference.

on the bilingual development data resulting from training with each group of features removed.[6] Note that though head word features seemed to be detrimental in our rapid training setup, earlier testing had shown a positive effect, so we reran the comparison using our full training setup, where we again saw a slight improvement when including these features.

3.5.2 Training Set Pruning

To find a good value of the ϵ parameter for training set pruning we tried several different values, using our rapid training setup and testing on the dev set. The results are shown in Table 3.2. We selected 25 as it showed the best performance/speed tradeoff, on average performing as well as if we had done no pruning at all, while requiring only a quarter the memory and CPU time.

[6]We do not have a test with the basic alignment features removed because they are necessary to compute $a_0(t, t')$.

ϵ	Chinese F_1	English F_1	Total F_1	Tree Pairs
15	85.78	77.75	82.05	1,463,283
20	85.88	77.27	81.90	1,819,261
25	86.37	78.92	82.91	2,204,988
30	85.97	79.18	82.83	2,618,686
40	86.10	78.12	82.40	3,521,423
50	85.95	78.50	82.50	4,503,554
100	86.28	79.02	82.91	8,997,708

Table 3.2: Training set pruning study. F_1 on dev set after training with different values of the ϵ parameter for training set pruning.

Cutoff	Chinese F_1	English F_1	Total F_1	Time (s)
50	86.34	79.26	83.04	174
100	86.61	79.31	83.22	307
200	86.67	79.39	83.28	509
500	86.76	79.41	83.34	1182
1000	86.80	79.39	83.35	2247
2000	86.78	79.35	83.33	4476
10,000	86.71	79.37	83.30	20,549

Table 3.3: Test set pruning study. F_1 on dev set obtained using different cutoffs for test set pruning.

3.5.3 Test Set Pruning

We also tried several different values of the rank cutoff for test set pruning, using the full training setup and testing on the dev set. The results are in Table 3.3. For F_1 evaluation, which is on a very small set of sentences, we selected 500 as the value with the best speed/performance tradeoff. However, when reranking our entire MT corpus, we used a value of 200, sacrificing a tiny bit of performance for an extra factor of 2 in speed.[7]

3.5.4 Sensitivity to k

Since our bitext parser currently operates as a reranker, the quality of the trees is limited by the quality of the k-best lists produced by the baseline parsers. To test this limitation, we evaluated performance on the dev set using baseline k-best lists of varying length. Training parameters were fixed (full training setup with $k = 100$) and test set pruning was disabled for these experiments. The results are in Table 3.4. The relatively modest gains with increasing k, even as the oracle scores continue to improve, indicate that performance is limited more by the model's reliance on the baseline parsers than by search errors that result from the reranking approach.

3.5.5 Parsing Results

Our final evaluation was done using the full training setup and evaluating on the bilingual test data. The results are in Table 3.5. Joint parsing improves F_1 by 2.6 points on out-of-domain English sentences and by 2.4 points on in-domain Chinese sentences.

[7]Using a rank cutoff of 200, the reranking step takes slightly longer than serially running both baseline parsers, and generating k-best lists takes slightly longer than getting 1-best parses, so in total, joint parsing takes about 2.3 times as long as monolingual parsing. With a rank cutoff of 500, total parsing time is scaled by a factor of around 3.8.

	Joint Parsing		Oracle	
k	Chinese F_1	English F_1	Chinese F_1	English F_1
1	84.95	76.75	84.95	76.75
10	86.23	78.43	90.05	81.99
25	86.64	79.27	90.99	83.37
50	86.61	79.10	91.82	84.14
100	86.71	79.37	92.23	84.73
150	86.67	79.47	92.49	85.17

Table 3.4: Sensitivity to k study. Joint parsing and oracle F_1 obtained on dev set using different maximum values of k when generating baseline k-best lists.

Parsing Model	Chinese F_1	English F_1	Total F_1
Monolingual	83.6	81.2	82.5
Bilingual	**86.0**	**83.8**	**84.9**

Table 3.5: Final evaluation. Comparison of F_1 on test set between baseline parsers and joint parser.

3.5.6 Machine Translation Results

We evaluated the effect of the bilingual parsing model on downstream MT performance using the setup described in Section 2.3. Since our MT system uses target-side syntax, we compared trees produced by the monolingual English parser to those produced by the joint model. Results are in Table 3.6. The trees produced by the bilingual model appear to result in a more powerful MT system, albeit one a bit more prone to overfitting, achieving an improvement of 0.7 BLEU on the tuning set which generalized to a 0.3 BLEU improvement on the test set.

3.5.7 Analysis

Overall, the bilingual parsing model was quite successful at its primary goal: leveraging bilingual information to improve parser quality. The bilingual model achieved an overall

Parsing Model	Tuning BLEU	Test BLEU
Monolingual	30.18	30.46
Bilingual	**30.84**	**30.74**

Table 3.6: MT comparison on the Moses syntactic string-to-tree MT system trained with English trees output from either the baseline monolingual parser or our bilingual parsing model.

parser improvement of 2.4 F_1, corresponding to a relative error reduction of about 14%. For the most part, this was achieved by using parsing decisions that are unambiguous in one language to inform higher-ambiguity decisions in the other. For example, the English parsing error shown in Figure 3.1 was fixed by the bilingual model, as it had access to the (correct) Chinese parse.

Unfortunately, though, the resulting improvements in machine translation quality were not quite as strong as we would have hoped. In part, as already mentioned, this was because the more powerful translation rule database that resulted from using the output of the bilingual model also appeared to be more prone to overfitting during parameter optimization (MERT). This will be a recurring issue, as we will see that tuning set improvements from improved trees will consistently be higher than the test set improvements. But more generally, as we will discuss in more detail in Chapter 5, it's not necessarily the case that the most linguistically accurate trees will always be the ones best suited for machine translation. We'll start to address this point in the next chapter where we describe a new model that explicitly learns how to model agreement between the parse trees and the word alignments. This model achieves roughly similar improvements in parse quality, but does so in a way that translates to better MT improvements. This effect is even more pronounced when we also allow modifications to the word alignments so that they match the parses better.

Chapter 4

Joint Parsing and Alignment

We know from earlier work that word aligners can be improved by using syntactic information (DeNero and Klein, 2007; May and Knight, 2007; Fossum et al., 2008), and in the previous chapter we saw how word alignment features can be leveraged to improve parsing performance in a bilingual setting. In the first case, however, parsers do not exploit bilingual information. In the second, word alignment is performed with a model that does not exploit syntactic information. This chapter presents a single, joint model for parsing and word alignment that allows both pieces to influence one another simultaneously.[1]

While building a joint model seems intuitive, there is no easy way to characterize how word alignments and syntactic parses should relate to each other in general. In the ideal situation, each pair of sentences in a bilingual corpus could be syntactically parsed using a synchronous context-free grammar. Of course, real translations are almost always at least partially syntactically divergent. Therefore, it is unreasonable to expect perfect matches of any kind between the two sides' syntactic trees, much less expect that those matches be well explained at a word level. Indeed, it is sometimes the case that large pieces of a sentence pair are completely asynchronous and can only be explained monolingually.

[1]The model described in this chapter was originally presented at NAACL in 2010 (Burkett et al., 2010).

This model exploits synchronization where possible to perform more accurately on both word alignment and parsing, but also allows independent models to dictate pieces of parse trees and word alignments when synchronization is impossible. This notion of “weak synchronization” is parameterized and estimated from data to maximize the likelihood of the correct parses and word alignments. Weak synchronization is closely related to quasi-synchronous grammars (Smith and Eisner, 2006, 2009) and the bilingual parse reranking model presented in the previous chapter, but those models assume that the word alignment of a sentence pair is known and fixed.

To simultaneously model both parses and alignments, this model loosely couples three separate combinatorial structures: monolingual trees in the source and target languages, and a synchronous ITG alignment that links the two languages (but is not constrained to match linguistic syntax). The model has no hard constraints on how these three structures must align, but instead contains a set of “synchronization” features that are used to propagate influence between the three component grammars. The presence of synchronization features couples the parses and alignments, but makes exact inference in the model intractable; we show how to use a variational mean field approximation, both for computing approximate feature expectations during training, and for performing approximate joint inference at test time.

Once again, we train the model on the English Chinese Translation Treebank. When evaluated on parsing and word alignment, this model significantly improves over independently-trained baselines: the monolingual parser of Petrov and Klein (2007) and the discriminative word aligner of Haghighi et al. (2009). It also slightly improves overall parsing performance over the bilingual parsing model from Chapter 3, yielding the highest joint parsing F_1 numbers on this data set. Furthermore, despite the fairly similar parsing performance, using this model to annotate MT training data works substantially better at improving syntactic MT performance: using both the parses and word alignments generated by this model yields a BLEU increase of almost 1.0 over the baseline.

4.1 Task Definition

Once again, given a source-language sentence, s, and a target-language sentence, s', we wish to predict a source tree t, a target tree t', and some kind of alignment a between the phrases in the sentence pair. These structures are illustrated in Figure 4.1.[2]

To facilitate these predictions, we define a conditional distribution $\mathrm{P}(t, a, t'|s, s')$. We begin with a generic conditional exponential form:

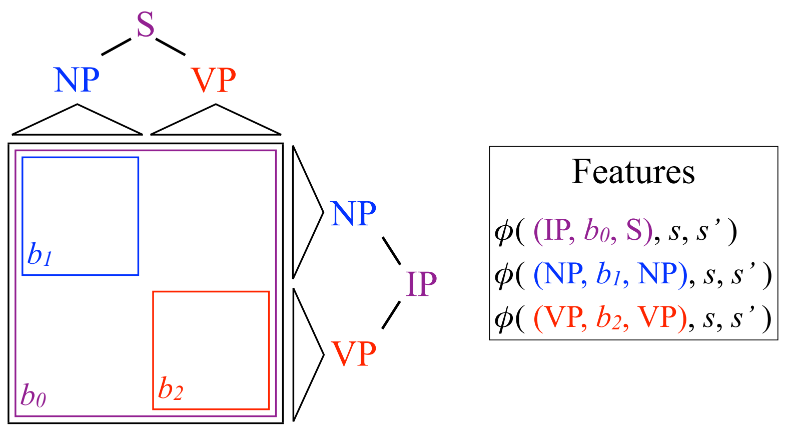
$$\mathrm{P}(t, a, t'|s, s') \propto \exp \theta^\top \phi(t, a, t', s, s') \tag{4.1}$$

Unfortunately, a generic model of this form is intractable, because we cannot efficiently sum over all triples (t, a, t') without some assumptions about how the features $\phi(t, a, t', s, s')$ decompose.

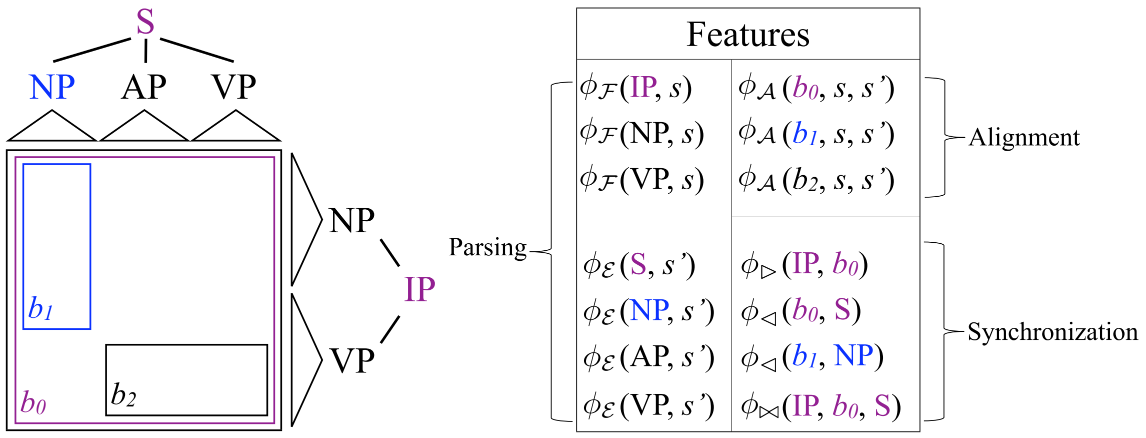
One natural solution is to restrict our candidate triples to those given by a synchronous context free grammar (SCFG) (Shieber and Schabes, 1990). Figure 4.1(a) gives a simple example of generation from a log-linearly parameterized synchronous grammar, together with its features. With the SCFG restriction, we can sum over the necessary structures using the $O(n^6)$ bitext inside-outside algorithm, making $\mathrm{P}(t, a, t'|s, s')$ relatively efficient to compute expectations under.

Unfortunately, an SCFG requires that *all* the constituents of each tree, from the root down to the words, are generated perfectly in tandem. The resulting inability to model any level of syntactic divergence prevents accurate modeling of the individual monolingual trees. We will consider the running example from Figure 4.2 throughout the chapter. Here, for instance, the verb phrase *established in such places as Quanzhou, Zhangzhou, etc.* in English does not correspond to any single node in the Chinese tree. A synchronous grammar has no choice but to analyze this sentence incorrectly, either by ignoring this verb phrase in English or postulating an incorrect Chinese constituent that corresponds to it.

[2]Note that unlike in the previous chapter, the alignment structure a might not directly correspond with either parse tree. In this chapter, a refers to generic alignments that map between phrases in s and s'.


S
NP
VP
NP
IP
VP
b_1
b_2
b_0
Features
ϕ((IP, b_0, S), s, s')
ϕ((NP, b_1, NP), s, s')
ϕ((VP, b_2, VP), s, s')

(a) Synchronous Rule


S
NP
AP
VP
NP
IP
VP
b_1
b_0
b_2
Features
Parsing
$\phi_{\mathcal{F}}$(IP, s)
$\phi_{\mathcal{F}}$(NP, s)
$\phi_{\mathcal{F}}$(VP, s)
$\phi_{\mathcal{E}}$(S, s')
$\phi_{\mathcal{E}}$(NP, s')
$\phi_{\mathcal{E}}$(AP, s')
$\phi_{\mathcal{E}}$(VP, s')
$\phi_{\mathcal{A}}$(b_0, s, s')
$\phi_{\mathcal{A}}$(b_1, s, s')
$\phi_{\mathcal{A}}$(b_2, s, s')
$\phi_{\triangleright}$(IP, b_0)
$\phi_{\triangleleft}$(b_0, S)
$\phi_{\triangleleft}$(b_1, NP)
$\phi_{\bowtie}$(IP, b_0, S)
Alignment
Synchronization

(b) Asynchronous Rule

Figure 4.1: Source trees, t (right), alignments, a (grid), and target trees, t' (top), and feature decompositions for synchronous (a) and weakly synchronous (b) grammars. Features always condition on bispans and/or anchored syntactic productions, but weakly synchronous grammars permit more general decompositions.

Therefore, instead of requiring strict synchronization, our model treats the two monolingual trees and the alignment as separate objects that can vary arbitrarily. However, the model rewards synchronization appropriately when the alignment brings the trees into correspondence.

4.2 Weakly Synchronized Grammars

We propose a joint model which still gives probabilities on triples (t, a, t'). However, instead of using SCFG rules to synchronously enforce the tree constraints on t and t', we only require that each of t and t' be well-formed under separate monolingual CFGs.

In order to permit efficient enumeration of all possible alignments a, we also restrict a to the set of unlabeled ITG bitrees (Wu, 1997), though again we do not require that a relate to t or t' in any particular way. Although this assumption does limit the space of possible word-level alignments, for the domain we consider (Chinese-English word alignment), the reduced space still contains almost all empirically observed alignments (Haghighi et al., 2009).[3] For example, in Figure 4.2, the word alignment is ITG-derivable, and each of the colored rectangles is a bispan in that derivation.

There are no additional constraints beyond the independent, internal structural constraints on t, a, and t'. This decoupling permits derivations like that in Figure 4.1(b), where the top-level syntactic nodes align, but their children are allowed to diverge. With the three structures separated, our first model is a completely factored decomposition of (4.1).

Formally, we represent a source tree t as a set of nodes $\{n\}$, each node representing a labeled span. Likewise, a target tree t' is a set of nodes $\{n'\}$.[4] We represent alignments a as

[3]See Section 4.7.1 for some new terminal productions required to make this true for the dataset used in this work.

[4]For expositional clarity, we describe n and n' as labeled spans only. However, in general, features that depend on n or n' are permitted to depend on the entire rule, and do in our final system.

sets of *bispans* $\{b\}$, indicated by rectangles in Figure 4.1.[5] Using this notation, the initial model has the following form:

$$\mathrm{P}(t,a,t'|s,s') \propto \exp\left(\sum_{n\in t}\theta^\top\phi_{\mathcal{F}}(n,s) + \sum_{b\in a}\theta^\top\phi_{\mathcal{A}}(b,s,s') + \sum_{n'\in t'}\theta^\top\phi_{\mathcal{E}}(n',s')\right) \tag{4.2}$$

Here $\phi_{\mathcal{F}}(n,s)$ indicates a vector of source node features, $\phi_{\mathcal{E}}(n',s')$ is a vector of target node features, and $\phi_{\mathcal{A}}(b,s,s')$ is a vector of alignment bispan features. Of course, this model is completely asynchronous so far, and fails to couple the trees and alignments at all. To permit soft constraints between the three structures we are modeling, we add a set of *synchronization* features.

For $n \in t$ and $b \in a$, we say that $n \triangleright b$ if n and b both map onto the same span of s. We define $b \triangleleft n'$ analogously for $n' \in t'$. We now consider three different types of synchronization features. Source-alignment synchronization features $\phi_{\triangleright}(n,b)$ are extracted whenever $n \triangleright b$. Similarly, target-alignment features $\phi_{\triangleleft}(b,n')$ are extracted if $b \triangleleft n'$. These features capture phenomena like that of bispan b_7 in Figure 4.2. Here the Chinese noun 锟斤拷 synchronizes with the ITG derivation, but the English projection of b_7 is a distituent. Finally, we extract source-target features $\phi_{\bowtie}(n,b,n')$ whenever $n \triangleright b \triangleleft n'$. These features capture complete bispan synchrony (as in bispan b_8) and can be expressed over triples (n,b,n') which happen to align, allowing us to reward synchrony, but not requiring it. All of these licensing conditions are illustrated in Figure 4.1(b).

With these features added, the final form of the model is:

$$\begin{aligned}\mathrm{P}(t,a,t'|s,s') \propto \exp\Bigg(&\sum_{n\in t}\theta^\top\phi_{\mathcal{F}}(n,s) + \sum_{b\in a}\theta^\top\phi_{\mathcal{A}}(b,s,s') + \sum_{n'\in t'}\theta^\top\phi_{\mathcal{E}}(n',s') + \\ &\sum_{n\triangleright b}\theta^\top\phi_{\triangleright}(n,b) + \sum_{b\triangleleft n'}\theta^\top\phi_{\triangleleft}(b,n') + \sum_{n\triangleright b\triangleleft n'}\theta^\top\phi_{\bowtie}(n,b,n')\Bigg)\end{aligned} \tag{4.3}$$

[5] Alignments a link arbitrary spans of s and s' (including non-constituents and individual words). We discuss the relation to word-level alignments in Section 4.3.

We emphasize that because of the synchronization features, this final form does *not* admit any known efficient dynamic programming for the exact computation of expectations. We will therefore turn to a variational inference method in Section 4.5.

4.3 Features

With the model's locality structure defined, we just need to specify the actual feature function, ϕ. We divide the features into three types: parsing features ($\phi_{\mathcal{F}}(n, s)$ and $\phi_{\mathcal{E}}(n', s')$), alignment features ($\phi_{\mathcal{A}}(b, s, s')$) and synchronization features ($\phi_{\triangleright}(n, b)$, $\phi_{\triangleleft}(b, n')$, and $\phi_{\bowtie}(n, b, n')$). We detail each of these in turn here.

4.3.1 Parsing

The monolingual parsing features we use are simply parsing model scores under the parser of Petrov and Klein (2007). While that parser uses heavily refined PCFGs with rule probabilities defined at the refined symbol level, we interact with its posterior distribution via posterior marginal probabilities over unrefined symbols. In particular, to each unrefined anchored production ${}_i\mathrm{A}_j \rightarrow {}_i\mathrm{B}_k\mathrm{C}_j$, we associate a single feature whose value is the marginal quantity $\log \mathrm{P}({}_i\mathrm{B}_k\mathrm{C}_j | {}_i\mathrm{A}_j, s)$ under the monolingual parser. These scores are the same as the variational rule scores of Matsuzaki et al. (2005).[6]

4.3.2 Alignment

We begin with the same set of alignment features as Haghighi et al. (2009), which are defined only for terminal bispans. In addition, we include features on nonterminal bispans

[6] Of course the structure of our model permits any of the additional rule-factored monolingual parsing features that have been described in the parsing literature, but in the present work we focus on the contributions of joint modeling.

including a bias feature, features that measure the difference in size between the source and target spans, features that measure the difference in relative sentence position between the source and target spans, and features that measure the density of word-to-word alignment posteriors under a separate unsupervised word alignment model.

4.3.3 Synchronization

Our synchronization features are indicators for the syntactic types of the participating nodes. We determine types at both a coarse (more collapsed than Treebank symbols) and fine (Treebank symbol) level. At the coarse level, we distinguish between phrasal nodes (e.g. S, NP), synthetic nodes introduced in the process of binarizing the grammar (e.g. S′, NP′), and part-of-speech nodes (e.g. NN, VBZ). At the fine level, we distinguish all nodes by their exact label. We use coarse and fine types for both partially synchronized (source-alignment or target-alignment) features and completely synchronized (source-alignment-target) features. The inset of Figure 4.2 shows some sample features. Of course, we could devise even more sophisticated features by using the input text itself. As we shall see, however, this model gives significant improvements with these simple features alone.

4.4 Learning

This model is trained in a supervised setting, using a bilingual treebank with gold trees and alignments. However, although the model assigns probabilities to entire synchronous derivations of sentences, the available corpora of this type only provide alignments only at the word level (1 by 1 bispans in Figure 4.2). This means that the alignment variable a is not fully observed. Because of this, given a particular word alignment w, we maximize the marginal probability of the set of derivations $\mathcal{A}(w)$ that are consistent with w (Haghighi

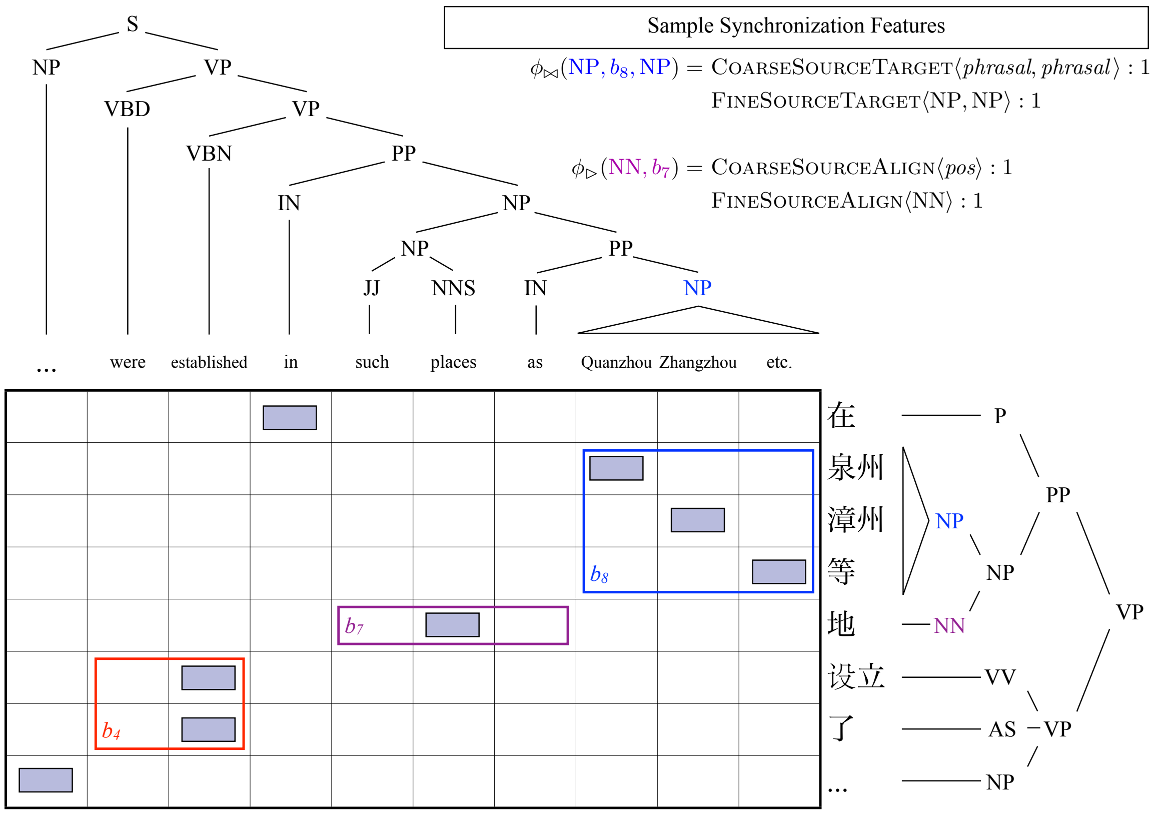

Sample Synchronization Features
$\phi_{\bowtie}(\text{NP}, b_8, \text{NP}) = \textsc{CoarseSourceTarget}\langle \textit{phrasal}, \textit{phrasal} \rangle : 1$
$\textsc{FineSourceTarget}\langle \text{NP}, \text{NP} \rangle : 1$
$\phi_{\triangleright}(\text{NN}, b_7) = \textsc{CoarseSourceAlign}\langle \textit{pos} \rangle : 1$
$\textsc{FineSourceAlign}\langle \text{NN} \rangle : 1$
S
NP
VP
VBD
VP
VBN
PP
IN
NP
NP
PP
JJ
NNS
IN
NP
...
were
established
in
such
places
as
Quanzhou
Zhangzhou
etc.
b_4
b_7
b_8
在
泉州
漳州
等
地
设立
了
...
P
PP
NP
NP
NN
VP
VV
AS
VP
NP

Figure 4.2: An example of a Chinese-English sentence pair with parses, word alignments, and a subset of the full optimal ITG derivation, including one totally unsynchronized bispan (b_4), one partially synchronized bispan (b_7), and and fully synchronized bispan (b_8). The inset provides some examples of active synchronization features (see Section 4.3.3) on these bispans. On this example, the monolingual English parser erroneously attached the lower PP to the VP headed by *established*, and the non-syntactic ITG word aligner misaligned 锟斤拷 to *such* instead of to *etc.* Our joint model corrected both of these mistakes because it was rewarded for the synchronization of the two NPs joined by b_8.

et al., 2009):[7]

$$\mathcal{L}(\theta) = \log \sum_{a \in \mathcal{A}(w_i)} \mathrm{P}(t_i, a, t'_i | s_i, s'_i)$$

We maximize this objective using standard gradient methods (Nocedal and Wright, 1999). As with fully visible log-linear models, the gradient for the i^{th} sentence pair with respect to θ is a difference of feature expectations:

$$\nabla \mathcal{L}(\theta) = \mathrm{E}_{\mathrm{P}(a|t_i,w_i,t'_i,s_i,s'_i)} \left[\phi(t_i, a, t'_i, s_i, s'_i)\right] - \mathrm{E}_{\mathrm{P}(t,a,t'|s_i,s'_i)} \left[\phi(t, a, t', s_i, s'_i)\right] \tag{4.4}$$

We cannot efficiently compute the model expectations in this equation exactly. Therefore we turn next to an approximate inference method.

4.5 Mean Field Inference

Instead of computing the model expectations from (4.4), we compute the expectations for each sentence pair with respect to a simpler, fully factored distribution $\mathrm{Q}(t, a, t') = q(t)q(a)q(t')$. Rewriting Q in log-linear form, we have:

$$\mathrm{Q}(t, a, t') \propto \exp\left(\sum_{n \in t} \psi_n + \sum_{b \in a} \psi_b + \sum_{n' \in t'} \psi_{n'}\right)$$

Here, the ψ_n, ψ_b and $\psi_{n'}$ are variational parameters which we set to best approximate our weakly synchronized model from (4.3):

$$\psi^* = \underset{\psi}{\operatorname{argmin}}\, \mathrm{KL}\left(\mathrm{Q}_\psi || P_\theta(t, a, t' | s, s')\right)$$

Once we have found Q, we compute an approximate gradient by replacing the model expectations with expectations under Q:

$$\mathrm{E}_{\mathrm{Q}(a|w_i)} \left[\phi(t_i, a, t'_i, s_i, s'_i)\right] - \mathrm{E}_{\mathrm{Q}(t,a,t'|s_i,s'_i)} \left[\phi(t, a, t', s_i, s'_i)\right]$$

[7]We also learn from non-ITG alignments by maximizing the marginal probability of the set of minimum-recall error alignments in the same way as Haghighi et al. (2009)

```
Input:    sentence pair $(s, s')$
          parameter vector $\theta$
Output:   variational parameters $\psi$
1. Initialize:
   $\psi_n^0 \leftarrow \theta^\top \phi_{\mathcal{F}}(n, s)$
   $\psi_b^0 \leftarrow \theta^\top \phi_{\mathcal{A}}(b, s, s')$
   $\psi_{n'}^0 \leftarrow \theta^\top \phi_{\mathcal{E}}(n', s')$
   $\mu_n^0 \leftarrow \sum_t q_{\psi^0}(t) I(n \in t)$
   $\mu_b^0 \leftarrow \sum_a q_{\psi^0}(a) I(b \in a)$
   $\mu_{n'}^0 \leftarrow \sum_{t'} q_{\psi^0}(t') I(n' \in t')$
2. While not converged, for each $n, b, n'$ in the monolingual and ITG charts:
     $\psi_n^i \leftarrow \theta^\top \big(\phi_{\mathcal{F}}(n, s) + \sum_{b:n \rhd b} \mu_b^{i-1} \phi_{\rhd}(n, b) +$
                 $\sum_{b:n \rhd b} \sum_{n':b \lhd n'} \mu_b^{i-1} \mu_{n'}^{i-1} \phi_{\bowtie}(n, b, n')\big)$
     $\mu_n^i \leftarrow \sum_t q_{\psi^i}(t) I(n \in t)$ (inside-outside)
     $\psi_b^i \leftarrow \theta^\top \big(\phi_{\mathcal{A}}(b, s, s') + \sum_{n:n \rhd b} \mu_n^{i-1} \phi_{\rhd}(n, b) + \sum_{n':b \lhd n'} \mu_{n'}^{i-1} \phi_{\lhd}(b, n') +$
                 $\sum_{n:n \rhd b} \sum_{n':b \lhd n'} \mu_n^{i-1} \mu_{n'}^{i-1} \phi_{\bowtie}(n, b, n')\big)$
     $\mu_b^i \leftarrow \sum_a q_{\psi^i}(a) I(b \in a)$ (bitext inside-outside)
     $\psi_{n'}^i \leftarrow \theta^\top \big(\phi_{\mathcal{E}}(n', s') + \sum_{b:b \lhd n'} \mu_b^{i-1} \phi_{\lhd}(b, n') +$
                 $\sum_{b:b \lhd n'} \sum_{n:n \rhd b} \mu_b^{i-1} \mu_n^{i-1} \phi_{\bowtie}(n, b, n')\big)$
     $\mu_{n'}^i \leftarrow \sum_{t'} q_{\psi^i}(t') I(n' \in t')$ (inside-outside)
3. Return variational parameters $\psi$
```

Figure 4.3: Structured mean field inference for the weakly synchronized model. $I(n \in t)$ is an indicator value for the presence of node n in source tree t.

Now, we will briefly describe how we compute Q. First, note that the parameters ψ of Q factor along individual source nodes, target nodes, and bispans. The combination of the KL objective and our particular factored form of Q make our inference procedure a structured mean field algorithm (Saul and Jordan, 1996). Structured mean field techniques are well-studied in graphical models, and our adaptation in this section to multiple grammars follows standard techniques (see e.g. Wainwright and Jordan, 2008).

Because all of the synchronization features can be represented as products of indicator features, the mean field updates for ψ take a standard form based on expected feature counts (Xing et al., 2004). This makes it simple to describe the algorithm (shown in Figure 4.3) procedurally. Similar to block Gibbs sampling, we iteratively optimize each component (source parse, target parse, and alignment) of the model in turn, conditioned on the others. Where block Gibbs sampling conditions on fixed trees or ITG derivations, our mean field algorithm maintains uncertainty in the form of monolingual parse forests or ITG forests. The key components to this uncertainty are the expected counts of particular source nodes, target nodes, and bispans under the mean field distribution:

$$\mu_n = \sum_t q_\psi(t) I(n \in t)$$

$$\mu_{n'} = \sum_{t'} q_\psi(t') I(n' \in t')$$

$$\mu_b = \sum_a q_\psi(a) I(b \in a)$$

Since dynamic programs exist for summing over each of the individual factors, these expectations can be computed in polynomial time.

4.5.1 Pruning

Although we can approximate the expectations from (4.4) in polynomial time using our mean field distribution, in practice we must still prune the ITG forests and monolingual parse forests to allow tractable inference. We prune our ITG forests using the same basic idea as Haghighi et al. (2009), but we employ a technique that allows us to be more aggressive. Where Haghighi et al. (2009) pruned bispans based on how many unsupervised HMM alignments were violated, we first train a maximum-matching word aligner (Taskar et al., 2005) using our supervised data set, which has only half the precision errors of the unsupervised HMM. We then prune every bispan which violates at least three alignments from the maximum-matching aligner. When compared to pruning the bitext forest of our model with Haghighi et al. (2009)'s HMM technique, this new technique allows us to maintain the same level of accuracy while cutting the number of bispans in half.

In addition to pruning the bitext forests, we also prune the syntactic parse forests using the monolingual parsing model scores. For each unrefined anchored production ${}_i\mathrm{A}_j \rightarrow {}_i\mathrm{B}_k\mathrm{C}_j$, we compute the marginal probability $\mathrm{P}({}_i\mathrm{A}_{j},{}_i\mathrm{B}_{k},{}_k\mathrm{C}_j|s)$ under the monolingual parser (these are equivalent to the maxrule scores from Petrov and Klein 2007). We only include productions where this probability is greater than 10^{-20}. Note that at training time, we are not guaranteed that the gold trees will be included in the pruned forest. Because of this, we replace the gold trees t_i, t'_i with oracle trees from the pruned forest, which can be found efficiently using a variant of the inside algorithm (Huang, 2008).

4.6 Testing

Once the model has been trained, we still need to determine how to use it to predict parses and word alignments for our test sentence pairs. Ideally, given the sentence pair (s, s'), we would find:

$$\begin{aligned}(t^*, w^*, t'^*) &= \operatorname*{argmax}_{t,w,t'} \mathrm{P}(t, w, t'|s, s') \\ &= \operatorname*{argmax}_{t,w,t'} \sum_{a \in \mathcal{A}(w)} \mathrm{P}(t, a, t'|s, s')\end{aligned}$$

Of course, this is also intractable, so we once again resort to our mean field approximation. This yields the approximate solution:

$$(t^*, w^*, t'^*) = \operatorname*{argmax}_{t,w,t'} \sum_{a \in \mathcal{A}(w)} \mathrm{Q}(t, a, t')$$

However, recall that Q incorporates the model's mutual constraint into the variational parameters, which factor into $q(t)$, $q(a)$, and $q(t')$. This allows us to simplify further, and find the maximum a posteriori assignments under the variational distribution. The trees can be found quickly using the Viterbi inside algorithm on their respective qs:

$$t^* = \operatorname*{argmax}_t q(t)$$

$$t'^* = \operatorname*{argmax}_{t'} q(t')$$

However, finding the optimum w under q:

$$w^* = \operatorname*{argmax}_w \sum_{a \in \mathcal{A}(w)} q(a)$$

is still intractable.

As we cannot find the maximum probability word alignment, we provide two alternative approaches for finding w^*. The first is to just find the Viterbi ITG derivation $a^* = \operatorname{argmax}_a q(a)$ and then set w^* to contain exactly the 1x1 bispans in a^*. The second method, posterior thresholding, is to compute posterior marginal probabilities under q for each 1x1 cell beginning at position i, j in the word alignment grid:

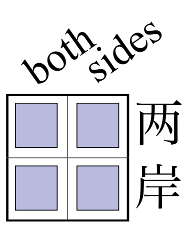
$$m(i,j) = \sum_a q(a) I((i, i+1, j, j+1) \in a)$$

We then include $w(i,j)$ in w^* if $m(w(i,j)) > \tau$, where τ is a threshold chosen to trade off precision and recall. In preliminary experiments, we found that the Viterbi alignment was uniformly worse than posterior thresholding. All the results from Section 4.7.2 use the threshold $\tau = 0.25$. For machine translation experiments, we set τ so that the resulting word alignments had roughly the same density (i.e. fraction of aligned words) as the baseline HMM alignments.[8]

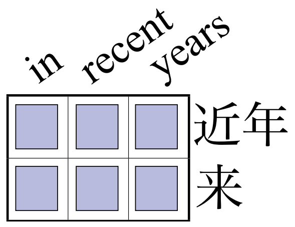
4.7 Experiments

Our experimental setup is once again as described in Chapter 2. For this model, in addition to measuring parsing F_1, we also measure how well the word alignments match

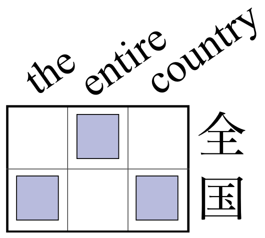
[8]Because of the way word alignments interact with the rule extraction procedure, the density of the alignments can have a direct impact on MT performance. We attempted to control for density as much as possible in order to avoid conflating word alignment density and word alignment quality.



(a) 2x2



(b) 2x3



(c) Gapped 2x3

Figure 4.4: Examples of phrasal alignments that can be represented by our new ITG terminal bispans.

gold-standard annotations according to AER and F_1. In our syntactic MT experiments, we investigate how the two relevant components of the model (English parse trees and word alignments) affect MT performance individually and in tandem.

4.7.1 Dataset-specific ITG Terminals

The gold word alignments in these corpora include significantly more many-to-many word alignments than those used by Haghighi et al. (2009). We are able to produce some of these many-to-many alignments by including new many-to-many terminals in our ITG word aligner, as shown in Figure 4.4. Our terminal productions sometimes capture non-literal translation like *both sides* or *in recent years*. They also can allow us to capture particular, systematic changes in the annotation standard. For example, the gapped pattern from Figure 4.4 captures the standard that the English word *the* is always aligned to the Chinese head noun in a noun phrase. We featurize these non-terminals with features similar to those of Haghighi et al. (2009), and all of the alignment results we report in Section 4.7.2 (both joint and ITG) employ these features.

Parsing Model	Chinese F_1	English F_1	Total F_1
Monolingual	83.6	81.2	82.5
Bilingual Reranker	**86.0**	83.8	84.9
Joint Parsing and Alignment	85.7	**84.5**	**85.1**

Table 4.1: Parsing results. The joint model has the highest reported F_1 for English-Chinese bilingual parsing.

Alignment Model	Precision	Recall	AER	F_1
HMM	86.0	58.4	30.0	69.5
ITG	**86.8**	73.4	20.2	79.5
Joint Parsing and Alignment	85.5	**84.6**	**14.9**	**85.0**

Table 4.2: Word alignment results. This joint model has the highest reported F_1 for English-Chinese word alignment.

4.7.2 Parsing and Word Alignment

We evaluated the model's intrisic accuracy as described in Section 2.2. However, to speed up training, we only used training sentences of length ≤ 50 words, which left us with 1974 of 2261 sentences. We compare our parsing results to the monolingual parsing models and to the bilingual reranker from Chapter 3. The results are in Table 4.1. For word alignment, we compare to the baseline unsupervised HMM word aligner and to the English-Chinese ITG-based word aligner of Haghighi et al. (2009). The results are in Table 4.2.

As can be seen, the joint model makes substantial improvements over the independent models. For parsing, we improve absolute F_1 over the monolingual parsers by 2.1 in Chinese, and by 3.3 in English. Though Chinese parsing results are slightly weaker than those of the bilingual parsing model from the previous chapter, English results are stronger, resulting in an overall F_1 gain, despite this model using a much simpler set of synchronization

Parsing Model	Alignment Model	Tuning BLEU	Test BLEU
Monolingual	HMM	30.18	30.46
Reranker	HMM	30.84	30.74
Joint PA	HMM	31.14	30.91
Monolingual	ITG	30.47	30.64
Monolingual	Joint PA	30.92	31.07
Joint PA	Joint PA	**31.25**	**31.40**

Table 4.3: Machine translation results.

features. For word alignment, we improve absolute F_1 by 5.5 over the non-syntactic ITG word aligner.

4.7.3 Machine Translation

We tested MT performance using the setup described in Section 2.3. Table 4.3 gives the results. We see that the English trees and the word alignments each provide independent improvements in MT performance. However, unsurprisingly, our best overall MT performance so far comes when using the predictions of the joint model jointly.

4.7.4 Analysis

Once again, we see that joint modeling can be very effective at directly improving parsing accuracy, and in this model this finding is extended to word alignment accuracy as well. However, by improving both simultaneously, we were able to achieve substantially better MT improvements: just shy of +1.0 BLEU rather than the +0.3 BLEU we found with the previous model. Part of this may be due to the fact that the English parses and the word alignments were both of higher quality than before. However, there is an additional confounding factor: the parses and alignments output by this model, more than any other parses or alignments we tried, were directly encouraged to agree about phrasal boundaries.

As we will discuss in the following chapter, this agreement may actually be even more important for syntactic MT performance than adherence to guidelines laid out by human annotators.

Chapter 5

Transforming Trees

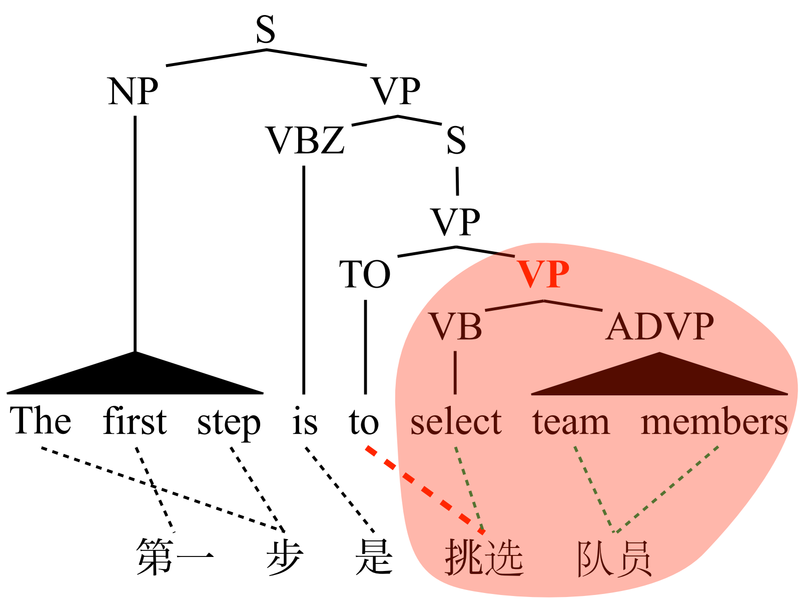
Monolingually, many Treebank conventions are more or less equally good. For example, the English WSJ treebank (Marcus et al., 1993) attaches verbs to objects rather than to subjects, and it attaches prepositional modifiers outside of all quantifiers and determiners. The former matches most linguistic theories while the latter does not, but to a monolingual parser, these conventions are equally learnable. As we've shown in the previous two chapters, leveraging bilingual information allows us to do even better at learning to match treebank annotations. However, once bilingual data is involved, some treebank conventions may entail constraints on rule extraction that are not borne out by semantic alignments. To the extent that there are simply divergences in the syntactic structure of the two languages, it will often be impossible to construct syntax trees that are simultaneously in full agreement with monolingual linguistic theories and with the alignments between sentences in both languages.

To see this, in Figure 5.1a we return to the example English tree from Chapter 1, which was taken from the English side of the English Chinese Translation Treebank (Bies et al., 2007). The lowest VP in this tree is headed by 'select,' which aligns to the Chinese verb ' 挑选.' However, ' 挑选' also aligns to the other half of the English infinitive, 'to,' which, following common English linguistic theory, is outside the VP. Because of this violating alignment, many syntactic machine translation systems (Galley et al., 2004; Huang et al., 2006) won't extract any translation rules for this constituent. However, by applying a simple transformation to the English tree to set up the infinitive as its own constituent, we get the tree in Figure 5.1b, which may be less well-motivated linguistically, but which corresponds better to the Chinese-mediated semantics and permits the extraction of many more syntactic MT rules.

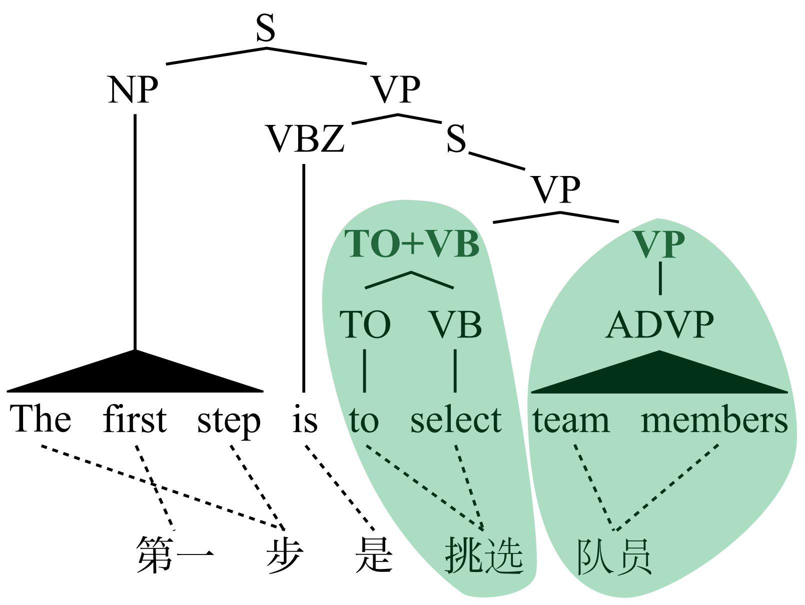
In this work, we develop a method based on *transformation-based learning* (Brill, 1995) for automatically acquiring a sequence of tree transformations of the sort in Figure 5.1. Once the transformation sequence has been learned, it can be deterministically applied to any parsed sentences, yielding new parse trees with constituency structures that agree better with the bilingual alignments yet remain consistent across the corpus. In particular, we use this method to learn a transformation sequence for the English trees in our Chinese to English MT training data. By using this technique in tandem with simple binarization, which also can help with syntactic rule extraction (Wang et al., 2007), we achieve aggregate improvements of up to +1.8 BLEU.[1]

There are two main ideas that contribute to the success of this system and that appear elsewhere in the recent machine translation literature. First is the basic notion we've already laid out: simple tree transformations can result in better syntactic translation rules. We will directly transform input syntactic annotations that can be fed into any syntactic MT system, but Zhao et al. (2011) built a particular rule extraction procedure that was trained to directly transform the subtrees making up individual translation rules by using a manually constructed set of transformations similar to those learned by our system. The second, and perhaps more important idea, is to optimize a proxy evaluation metric that is more closely related to the final goal task than parsing F_1. Katz-Brown et al. (2011) retrained

[1]The technique described in this chapter was originally presented at EMNLP in 2012 (Burkett and Klein, 2012b).



(a) Before



(b) After

Figure 5.1: An example tree transformation merging a VB node with the TO sibling of its parent VP. Before the transformation (a), the bolded VP cannot be extracted as a translation rule, but afterwards (b), both this VP and the newly created TO+VB node are extractable.

a parser to directly optimize a word reordering metric in order to improve a downstream machine translation system that uses dependency parses in a preprocessing reordering step. Again, we instead take a more general approach, as our goal is to create trees that are more compatible with a wide range of syntactically-informed translation systems, particularly those that extract translation rules based on syntactic constituents. To that end, we optimize a metric that formalizes the notion of *agreement* between parses and word alignments that we have been discussing in less precise terms up to this point.

5.1 Agreement

Our primary goal in adapting parse trees is to improve their agreement with a set of external word alignments. Thus, our first step is to define an *agreement score* metric to operationalize this concept.

Central to the definition of our agreement score is the notion of an *extractable node*. Intuitively, an extractable English[2] tree node (also often called a "frontier node" in the literature), is one whose span aligns to a contiguous span in the foreign sentence.

Formally, we assume a fixed word alignment $w = \{(i, j)\}$, where $(i, j) \in w$ means that English word i is aligned to foreign word j. For an English span $[k, \ell]$ (inclusive), the set of aligned foreign words is:

$$\textit{fset}([k, \ell]) = \{j \mid \exists\, i : k \leq i \leq \ell; (i, j) \in w\}$$

We then define the aligned foreign span as:

$$\textit{fspan}([k, \ell]) = [\min(\textit{fset}([k, \ell])), \max(\textit{fset}([k, \ell]))]$$

[2]For expositional clarity, we will refer to "English" and "foreign" sentences/trees, but our definitions are in no way language dependent and apply equally well to any language pair.

The aligned English span for a given foreign span $[s, t]$ is defined analogously:

$$\textit{eset}([s,t]) = \{i \mid \exists\, j : s \leq j \leq t; (i,j) \in w\}$$
$$\textit{espan}([s,t]) = [\min(\textit{eset}([s,t])), \max(\textit{eset}([s,t]))]$$

Finally, we define $[k, \ell]$ to be extractable if and only if it has at least one word alignment and its aligned foreign span aligns back to a subspan of $[k, \ell]$:

$$\textit{fset}([k,\ell]) \neq \emptyset \wedge \textit{espan}(\textit{fspan}([k,\ell])) \subseteq [k,\ell]$$

With this definition of an extractable span, we can now define the agreement score $g_w(t)$ for an English tree t, conditioned on an alignment w:[3]

$$g_w(t) = \sum_{\substack{[k,\ell] \in t: \\ |[k,\ell]|>1}} sign([k,\ell]) \tag{5.1}$$

Where

$$sign([k,\ell]) = \begin{cases} 1 & [k,\ell] \text{ is extractable} \\ -1 & \text{otherwise} \end{cases}$$

Importantly, the sum in Equation 5.1 ranges over all *unique* spans in t. This is simply to make the metric less gameable, preventing degenerate solutions such as an arbitrarily long chain of unary productions over an extractable span. Also, since all individual words are generated by preterminal part-of-speech nodes, the sum skips over all length 1 spans.

As a concrete example of agreement score, we can return to Figure 5.1. The tree in Figure 5.1a has 6 unique spans, but only 5 are extractable, so the total agreement score is 5 - 1 = 4. After the transformation, though, the tree in Figure 5.1b has 6 extractable spans, so the agreement score is 6.

[3]Unextractable spans are penalized in order to ensure that space is saved for the formation of extractable ones.

Parsing Model	Alignment Model	Average Agreement	Test BLEU
Monolingual	HMM	5.01	30.46
Reranker	HMM	5.34	30.74
Joint PA	HMM	5.51	30.91
Monolingual	ITG	4.81	30.64
Monolingual	Joint PA	6.69	31.07
Joint PA	Joint PA	7.30	31.40

Table 5.1: Relationship between agreement score and machine translation performance. Agreement scores are averaged per sentence.

5.1.1 Relation to MT Performance

Now that we have a formal definition of agreement, it is interesting to see how well it relates to the differences in MT performance we've seen so far. Table 5.1 shows the average per-sentence agreement scores of the variously annotated MT training data we've tried alongside the headline BLEU numbers from Table 4.3. With the exception of the ITG alignments, there is a pretty consistent qualitative relationship between agreement score and MT performance. This suggests that modifying our annotations to directly optimize agreement score may yield additional improvements in translation quality.

5.2 Transformation-Based Learning

Transformation-based learning (TBL) was originally introduced via the Brill part-of-speech tagger (Brill, 1992) and has since been applied to a wide variety of NLP tasks, including binary phrase structure bracketing (Brill, 1993), PP-attachment disambiguation (Brill and Resnik, 1994), base NP chunking (Ramshaw and Marcus, 1995), dialogue act tagging (Samuel et al., 1998), and named entity recognition (Black and Vasilakopoulos, 2002).

The generic procedure is simple, and requires only four basic inputs: a set of training sentences, an initial state annotator, an inventory of atomic transformations, and an evaluation metric. First, you apply the initial state annotator (here, the source of original trees) to your training sentences to ensure that they all begin with a legal annotation. Then, you test each transformation in your inventory to see which one will yield the greatest improvement in the evaluation metric if applied to the training data. You greedily apply this transformation to the full training set and then repeat the procedure, applying transformations until some stopping criterion is met (usually either a maximum number of transformations, or a threshold on the marginal improvement in the evaluation metric).

The output of the training procedure is an ordered set of transformations. To annotate new data, you simply label it with the same initial state annotator and then apply each of the learned transformations in order. This process has the advantage of being quite fast (usually linear in the number of transformations and the length of the sentence; for parsing, the cost will typically be dominated by the cost of the initial state annotator), and, unlike the learned parameters of a statistical model, the set of learned transformations itself can often be of intrinsic linguistic interest.

For our task, we have already defined the evaluation metric (Section 5.1) and the initial state annotator will either be the gold Treebank trees or a Treebank-trained PCFG parser. Thus, to fully describe our system, it only remains to define the set of possible tree transformations.

5.3 Tree Transformations

The definition of an atomic transformation consists of two parts: a rewrite rule and the triggering environment (Brill, 1995). Tree transformations are best illustrated visually, and so for each of our transformation types, both parts of the definition are represented schematically in Figures 5.2-5.7. We have also included a real-world example of each type of transformation, taken from the English Chinese Translation Treebank.

Altogether, we define six types of tree transformations. Each class of transformation takes between two and four syntactic category arguments, and most also take a DIRECTION argument that can have the value *left* or *right*.[4] We refer to the nodes in the schematics whose categories are arguments of the transformation definition as *participating* nodes. Basically, a particular transformation is triggered anywhere in a parse tree where all participating nodes appear in the configuration shown. The exact rules for the triggering environment are:

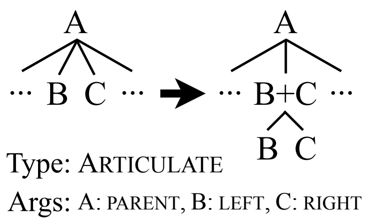
1. Each participating node must appear in the schematically illustrated relationship to the others. The non-participating nodes in the schematic do not have to appear. Similarly, any number of additional nodes can appear as siblings, parents, or children of the explicitly illustrated nodes.

2. Any node that will gain a new child as a result of the transformation must already have at least one nonterminal child. We have drawn the schematics to reflect this, so this condition is equivalent to saying that any participating node that is drawn with children must have a phrasal syntactic category (i.e. it cannot be a POS).

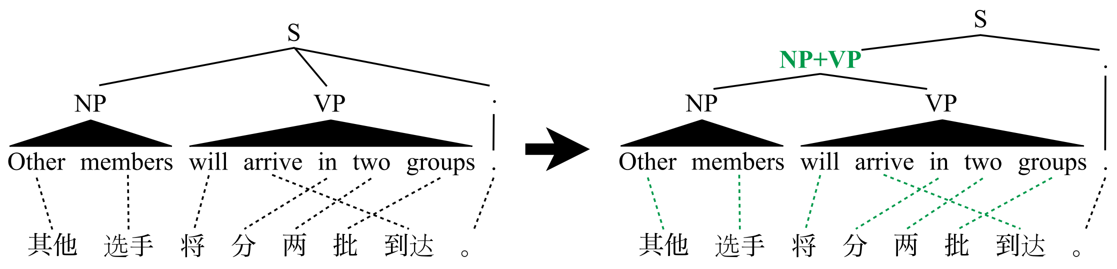
3. Repeated mergings are not allowed. That is, the newly created nodes that result from an ARTICULATE or ADOPT transformation cannot then participate as the LEFT or RIGHT argument of a subsequent ARTICULATE transformation or as the AUNT or TARGET argument of a subsequent ADOPT transformation. This is simply to prevent the unrestrained proliferation of new syntactic categories.

The rewrite rule for a transformation is essentially captured in the corresponding schematic. Additional nodes that do not appear in the schematic are generally handled

[4]To save space, the schematic for each of these transformations is only shown for the *left* direction, but the *right* version is simply the mirror image.

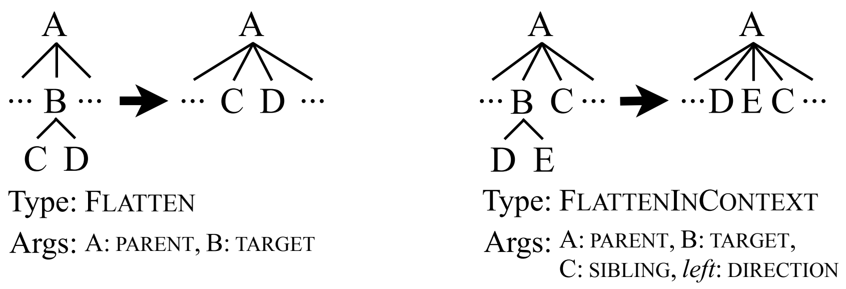


(a) Schematic

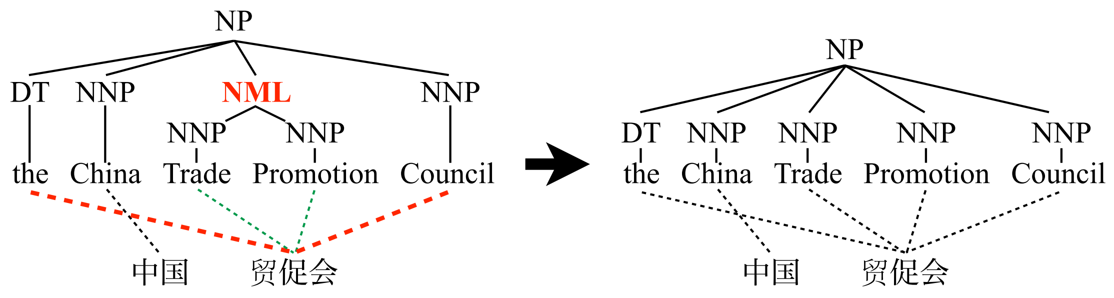

S
NP
VP
.
Other members will arrive in two groups .
其他 选手 将 分 两 批 到达 。
S
NP+VP
NP
VP
.
Other members will arrive in two groups .
其他 选手 将 分 两 批 到达 。

(b) Example: ARTICULATE⟨S, NP, VP⟩

Figure 5.2: ARTICULATE transformations.

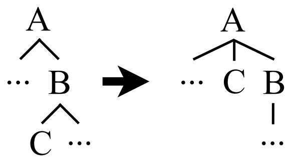

A
⋯ B ⋯
C D
A
⋯ C D ⋯
Type: FLATTEN
Args: A: PARENT, B: TARGET
A
⋯ B C ⋯
D E
A
⋯ D E C ⋯
Type: FLATTENINCONTEXT
Args: A: PARENT, B: TARGET, C: SIBLING, *left*: DIRECTION

(a) Schematic


NP
DT
NNP
NML
NNP
NNP
NNP
the China Trade Promotion Council
中国 贸促会
NP
DT NNP NNP NNP NNP
the China Trade Promotion Council
中国 贸促会

(b) Example: FLATTENINCONTEXT⟨NP, NML, NNP, *left*⟩

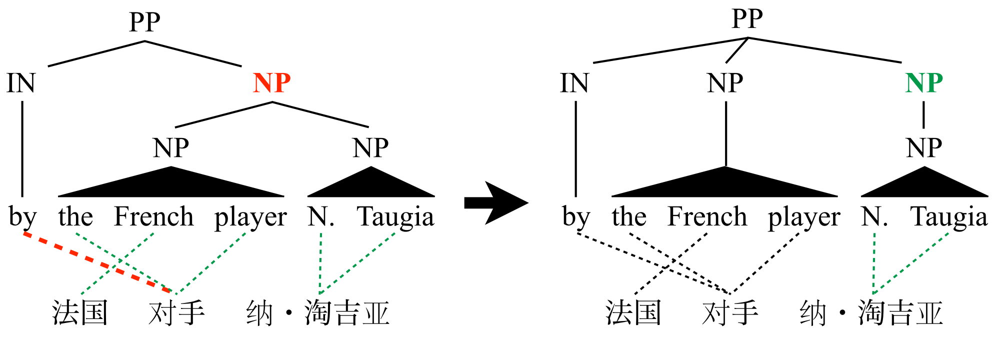
Figure 5.3: FLATTEN transformations.



Type: PROMOTE

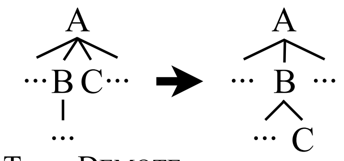
Args: A: GRANDPARENT, B: PARENT,
C: CHILD, *left*: DIRECTION

(a) Schematic


PP
IN
NP
NP
NP
by the French player N. Taugia
法国 对手 纳・淘吉亚
PP
IN
NP
NP
NP
by the French player N. Taugia
法国 对手 纳・淘吉亚

(b) Example: PROMOTE⟨PP, NP, NP, *left*⟩

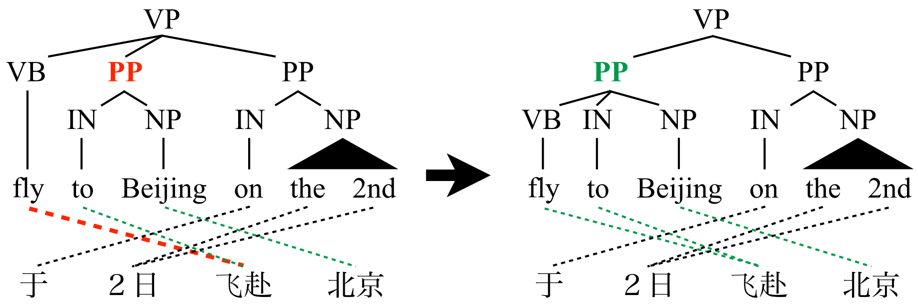
Figure 5.4: PROMOTE transformations.



Type: DEMOTE

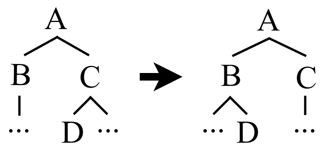
Args: A: PARENT, B: DEMOTER,
C: DEMOTED, *left*: DIRECTION

(a) Schematic



(b) Example: DEMOTE⟨VP, PP, VB, *right*⟩

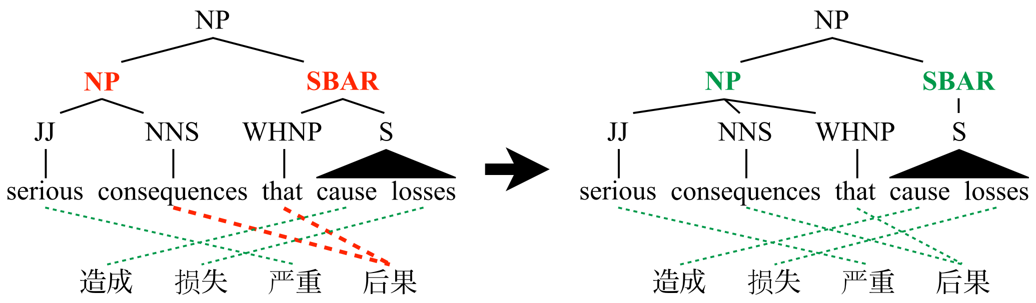
Figure 5.5: DEMOTE transformations.



Type: Transfer

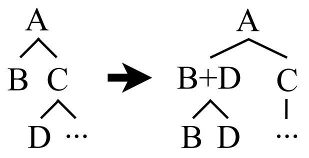
Args: A: grandparent, B: aunt,
C: parent, D: target,
left: direction

(a) Schematic


NP
NP
SBAR
JJ
NNS
WHNP
S
serious consequences that cause losses
造成 损失 严重 后果
NP
NP
SBAR
JJ
NNS
WHNP
S
serious consequences that cause losses
造成 损失 严重 后果

(b) Example: Transfer⟨NP, NP, SBAR, WHNP, *left*⟩

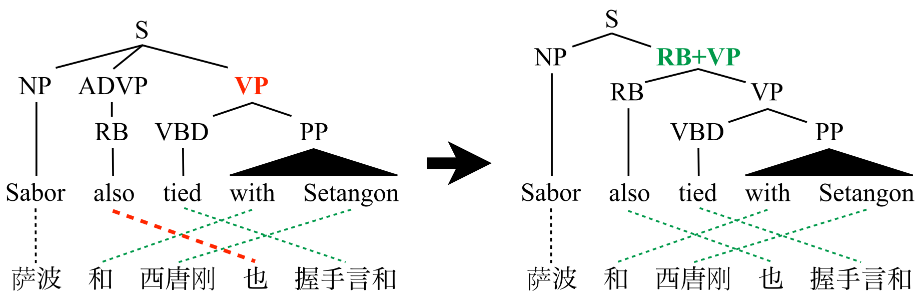
Figure 5.6: Transfer transformations.



Type: Adopt

Args: A: grandparent, B: aunt,
C: parent, D: target,
left: direction

(a) Schematic


S
NP
ADVP
VP
RB
VBD
PP
Sabor also tied with Setangon
萨波 和 西唐刚 也 握手言和
S
NP
RB+VP
RB
VP
VBD
PP
Sabor also tied with Setangon
萨波 和 西唐刚 也 握手言和

(b) Example: Adopt⟨S, VP, ADVP, RB, *right*⟩

Figure 5.7: Adopt transformations.

in the obvious way: unillustrated children or parents of illustrated nodes remain in place, while unillustrated siblings of illustrated nodes are handled identically to their illustrated siblings. The only additional part of the rewrite that is not shown explicitly in the schematics is that if the node in the PARENT position of a TRANSFER or ADOPT transformation is left childless by the transformation (because the TARGET node was its only child), then it is deleted from the parse tree. In the case of a transformation whose triggering environment appears multiple times in a single tree, transformations are always applied leftmost/bottom-up and exhaustively.[5]

In principle, our transformation inventory consists of all possible assignments of syntactic categories to the arguments of each of the transformation types (subject to the triggering environment constraints). In practice, though, we only ever consider transformations whose triggering environments appear in the training corpus (including new triggering environments that appear as the result of earlier transformations). While the theoretical space of possible transformations is exponentially large, the set of transformations we actually have to consider is quite manageable, and empirically grows substantially sublinearly in the size of the training set.

5.4 Results and Analysis

There are two ways to use this procedure. One is to apply it to the entire data set, with no separate training phase. Given that the optimization has no notion of gold transformations, this procedure is roughly like an unsupervised learner that clusters its entire data. Another way is to learn annotations on a subset of data and apply it to new data. We choose the latter primarily for reasons of efficiency and simplicity: many common use cases are easiest to

[5]The transformation is repeatedly applied at the lowest, leftmost location of the parse tree where the triggering environment appears, until the triggering environment no longer appears anywhere in the tree.

Transformations	Total Spans	Extractable Spans	Agreement Score
0	13.15	9.78	6.40
10	12.57	10.36	8.15
50	13.41	11.38	9.35
200	14.03	11.96	9.89
1584	14.58	12.36	10.15
2471	14.65	12.35	10.06

Table 5.2: Average span counts and agreement scores on the English Chinese Translation Treebank development set. The highest agreement score was attained at 1584 transformations, but most of the improvement happened much earlier.

manage when annotation systems can be trained once offline and then applied to new data as it comes in.

Since we intend for our system to be used as a pre-trained annotator, it is important to ensure that the learned transformation sequence achieves agreement score gains that generalize well to unseen data. To minimize errors that might be introduced by the noise in automatically generated parses and word alignments, and to maximize reproducibility, we conducted our initial experiments on the English Chinese Translation Treebank. For this dataset, the initial state annotations (parse trees) are the gold-standard parses that were manually created by trained human annotators; similarly, the gold-standard word alignments are used to compute the agreement score.[6] We trained the system on the training set, and evaluated on the development set.

The improvements in agreement score are shown in Figure 5.8, with a slightly more detailed breakdown at a few fixed points in Table 5.2. While the system was able to find up to 2471 transformations that improved the training set agreement score, the majority of the improvement, and especially the majority of the improvement that generalized to the test set, was achieved within the first 200 or so transformations. We also see from Table 5.2

[6]Note that unlike in the previous chapters, no monolingual parsers or unsupervised word aligners were used for these experiments. The results in this section are all about directly modifying the gold-standard trees to improve agreement with gold-standard word alignments.

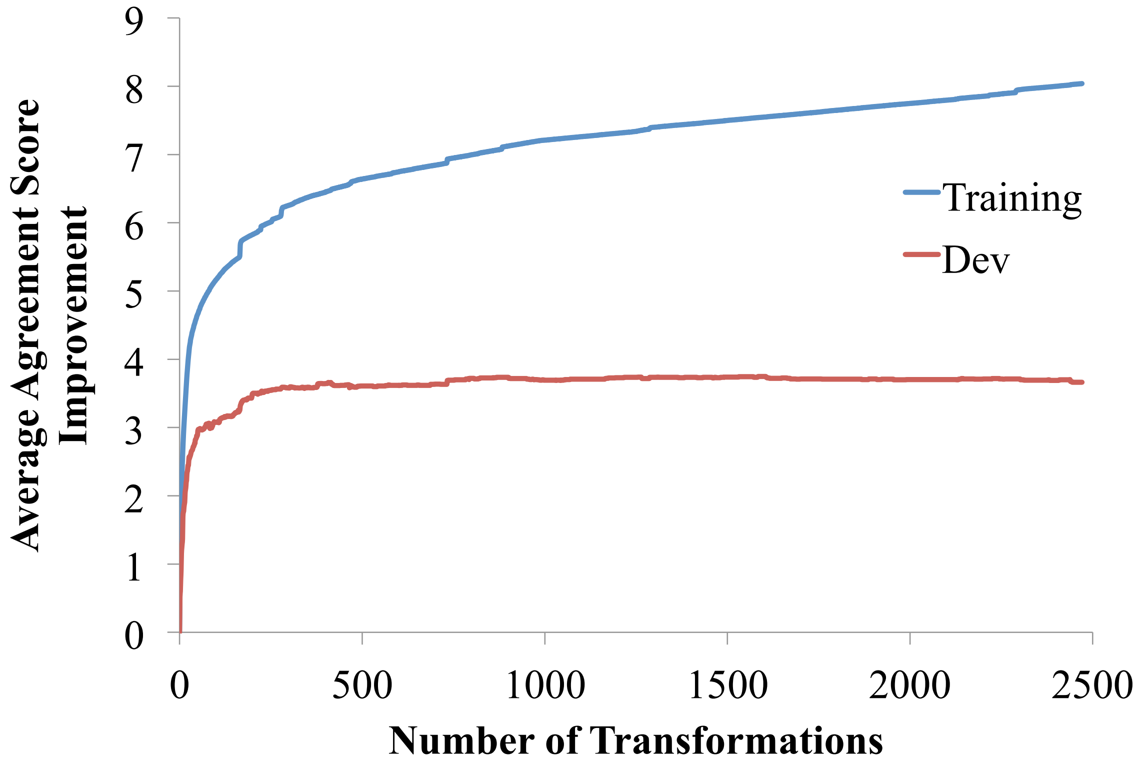


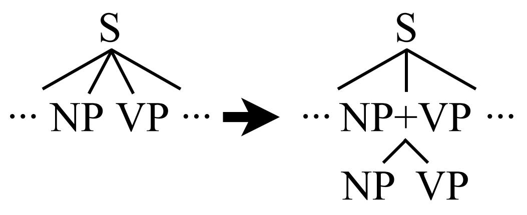
Figure 5.8: Transformation results on the English Chinese Translation Treebank. The value plotted is the average (per-sentence) improvement in agreement score over the baseline trees.

1	ARTICULATE⟨S,NP,VP⟩
2	FLATTENINCONTEXT⟨PP,NP,IN,*right*⟩
3	PROMOTE⟨VP,VP,VBN,*left*⟩
4	ADOPT⟨VP,TO,VP,VB,*left*⟩
5	ADOPT⟨PP,VBG,PP,IN,*left*⟩
6	FLATTEN⟨VP,VP⟩
7	ARTICULATE⟨VP,VBD,NP⟩
8	FLATTENINCONTEXT⟨PP,NML,NNP,*left*⟩
9	ARTICULATE⟨NP,NNP,NNS⟩
10	ARTICULATE⟨S,NP,ADVP⟩
11	TRANSFER⟨NP,NP,SBAR,WHNP,*left*⟩
12	FLATTENINCONTEXT⟨NP,NML,NNP,*left*⟩
13	ARTICULATE⟨NP,NN,NNS⟩
14	TRANSFER⟨NP,NP+,,SBAR,WHNP,*left*⟩
15	ADOPT⟨PP,IN,PP,IN,*left*⟩
16	PROMOTE⟨S,VP,CC+VP,*right*⟩
17	ARTICULATE⟨VP,VBZ,VBN⟩
18	ARTICULATE⟨VP,VBD,PP⟩
19	ARTICULATE⟨VP,MD,ADVP⟩
20	ADOPT⟨PP,SYM,QP,CD,*right*⟩

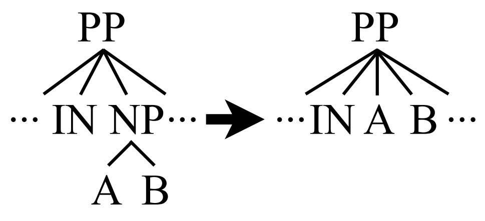
Table 5.3: The first 20 learned transformations, excluding those that only merged punctuation or conjunctions with adjacent phrases. The first 5 are illustrated in Figure 5.9.

that, though the first few transformations deleted many non-extractable spans, the overall trend was to produce more finely articulated trees, with the full transformation sequence increasing the number of spans by more than 10%.

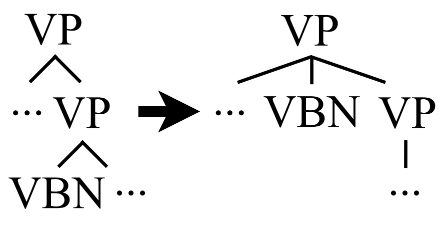
As discussed in Section 5.2, one advantage of TBL is that the learned transformations can themselves often be interesting. For this task, some of the highest scoring transformations did uninteresting things like conjoining conjunctions or punctuation, which are often either unaligned or aligned monotonically with adjacent phrases. However, by filtering out all ARTICULATE transformations where either the LEFT or RIGHT argument is "CC", "-RRB-", ",", or "." and taking the top 20 remaining transformations, we get the list in Table 5.3, the first 5 of which are also illustrated in Figure 5.9. Some of these (e.g. #1, #7, #10) are additional ways of creating new spans when English and Chinese phrase structures



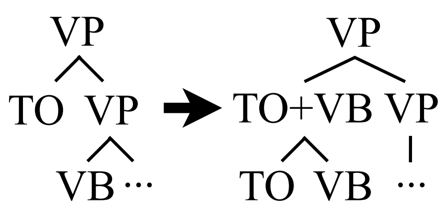
(a) Articulate⟨S,NP,VP⟩



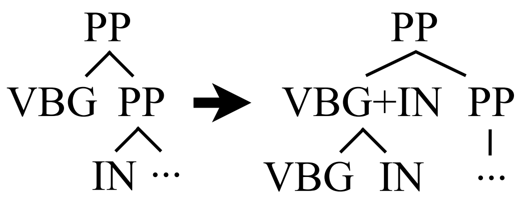
(b) FlattenInContext⟨PP,NP,IN,*right*⟩



(c) Promote⟨VP,VP,VBN,*left*⟩



(d) Adopt⟨VP,TO,VP,VB,*left*⟩



(e) Adopt⟨PP,VBG,PP,IN,*left*⟩

Figure 5.9: Illustrations of the top 5 transformations from Table 5.3.

roughly agree, but many others do recover known differences in English and Chinese syntax. For example, many of these transformations directly address compound verb forms in English, which tend to align to single words in Chinese: #3 (past participle constructions), #4 (infinitive), #6 (all), and #17 (present perfect). We also see differences between English and Chinese internal NP structure (e.g. #9, #12, #13).

5.5 Machine Translation

The ultimate goal of our system is to improve the agreement between the automatically generated parse trees and word alignments that are used as training data for syntactic machine translation systems. Given the amount of variability between the outputs of different parsers and word aligners (or even the same systems with different settings), the best way to improve agreement is to learn a transformation sequence that is specifically tuned for the same annotators (parsers and word aligners) we are evaluating with. In particular, we found that though training on the English Chinese Translation Treebank produces clean, interpretable rules, preliminary experiments showed little to no improvement from using these rules for MT, primarily because actual alignments are not only noisier but also systematically different from gold ones. Thus, all rules used for MT experiments were learned from automatically annotated text.

As mentioned in Section 5.4, we could, in principle train on all 506k sentences of our MT training data. However, this would be quite slow: each iteration of the training procedure requires iterating through all n training sentences[7] once for each of the m candidate transformations, for a total cost of $O(nm)$ where m grows (albeit sublinearly) with n. Because the agreement score metric is additive, the most useful transformations (and hence the first ones found by the greedy learning procedure) are typically the ones that are triggered the most frequently, so any reasonably sized training set is likely to contain them. Thus, it is not actually likely that dramatically increasing the size of the training set will yield particularly large gains.

To train our TBL system, we therefore extracted a random subset of 3000 sentences to serve as a training set.[8] We also extracted an additional 1000 sentence test set to use for

[7]By using a simple hashing scheme to keep track of triggering environments, this cost can be reduced greatly but is still linear in the number of training sentences.

[8]The sentences were shorter on average than those in the English Chinese Translation Treebank, so this training set contains roughly the same number of words as that used in the experiments from Section 5.4.

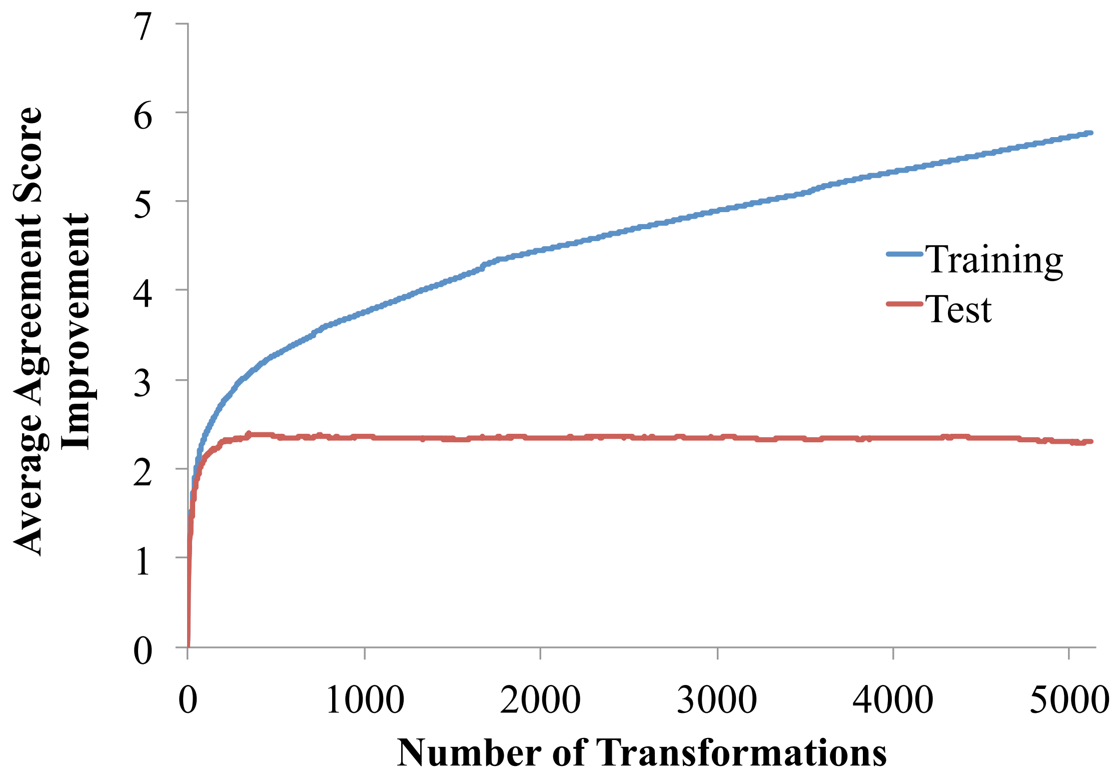


Figure 5.10: Transformation results on a subset of the MT training data. The training and test sets are disjoint in order to measure how well the learned transformation sequence generalizes. Once again, we plot the average improvement over the baseline trees.

rapidly evaluating agreement score generalization. Figure 5.10 illustrates the improvements in agreement score for the automatically annotated data, analogous to Figure 5.8. The same general patterns hold, although we do see that the automatically annotated data is more idiosyncratic and so more than twice as many transformations are learned before training set agreement stops improving, even though the training set sizes are roughly the same.[9] Furthermore, test set generalization in the automatic annotation setting is a little bit worse, with later transformations tending to actually hurt test set agreement.

Our MT setup is once again as described in Section 2.3. Word alignments were the

[9]Note that the training set improvement curves don't actually flatten out because training halts once no improving transformation exists.

ρ	Transformations	Agreement Score	Tuning BLEU	Test BLEU
∞	0	5.01	30.18	30.46
50	30	6.76	31.02	31.33
25	65	7.12	31.14	31.10
10	145	7.40	**31.59**	31.72
1	5122	**7.52**	31.52	**31.77**

Table 5.4: Machine translation results. Agreement scores are taken from the test data used to generate Figure 5.10. Note that using 0 transformations just yields the original baseline trees.

same for all experiments; the only difference was the English trees, for which we tested a range of transformation sequence prefixes (including a 0-length prefix, which just yields the original parser output, as a baseline). The various prefixes were selected by cutting off training whenever the best available transformation failed to increase the (total) training set agreement score by some minimum value, ρ. Table 5.4 shows the results of these experiments. As suggested by Figure 5.10, most of the improvement in agreement score comes early on, though gains do continue. We also see BLEU improvements following a similar trend, albeit much less smoothly. The eventual improvement of +1.3 BLEU is the best we have seen so far.

5.5.1 Binarization

Previous work has shown that because more tree nodes means more potential rules, creating more finely articulated trees through simple binarization is actually enough to yield syntactic MT improvements (Wang et al., 2007). Since the transformed trees tended to be more finely articulated, one possible objection to our results is that perhaps the system is getting all its gains from just adding more nodes. To rule out this explanation we conducted an additional experiment where we equalized the span count by also testing binarized versions of the baseline and transformed trees, using the left-branching and right-branching

Trees	Binarization		
	None	Left	Right
Baseline	30.46	31.15	31.27
Transformed	31.77	**32.26**	31.87

Table 5.5: Test set BLEU after binarizing both baseline and transformed trees.

Method	Average Agreement	Tuning BLEU	Test BLEU
Baseline	5.01	30.18	30.46
Bilingual Parsing	5.34	30.84	30.74
Joint PA	7.30	31.25	31.40
Tree Transformation	7.52	31.52	31.77

Table 5.6: Summary of the main results of this thesis.

binarization scripts included with Moses.[10] As Table 5.5 shows, binarization does indeed improve MT performance, particularly for the baseline trees. However, the tree transformations are still helpful above and beyond simple binarization, with transformed, binarized trees yielding an overall +1.8 BLEU improvement over just direct use of the parser output.

5.5.2 Comparison with Supervised Parsing Models

The headline MT findings from the three annotation systems described in this thesis are summarized in Table 5.6. Once we include the tree transformation system, the trend from Table 5.1 still holds, with larger agreement score improvements resulting in larger syntactic MT improvements. However, given that the joint models improve agreement in a fairly different way from the tree transformation system, one obvious next step is to see if these methods can be combined.

To test whether combination of these systems is helpful, we tried learning tree transformations from the output of each of the joint models. We then applied the appropriate set of

[10]Binarized trees are guaranteed to have $k - 1$ unique spans for sentences of length k.

Method	Average Agreement	Tuning BLEU	Test BLEU
Baseline	5.01	30.18	30.46
+ Transformation	7.52	31.52	31.77
Bilingual Parsing	5.34	30.84	30.74
+ Transformation	7.84	31.50	31.51
Joint PA	7.30	31.25	31.40
+ Transformation	9.91	31.62	31.34

Table 5.7: Combining joint parsing models with tree transformation.

learned transformations to reannotate the output of these models. The results are shown in Table 5.7. Unfortunately, combination is not as successful as we had hoped. The output of the bilingual reranker does benefit from learned transformations, but the aggregate result is not quite as good for MT performance as simply learning to transform the output of the baseline monolingual English parser. Learning transformations for the output of the joint parsing and alignment model yields the highest total agreement score by far, but though tuning set MT performance improves, test set performance does not.

Chapter 6

Conclusions

This thesis has investigated two closely related questions: First, how can bilingual information, especially bilingual annotated treebanks, be leveraged to improve the quality of automatic statistical parsers? And second, how do these improved syntactic annotations affect the quality of syntactic machine translation systems? In particular, what kinds of improvements are most beneficial? The thesis has three main findings:

1. When bilingual information is available it can lead to fairly impressive improvements in parsing quality, as measured by parser F_1. By using a latent alignment between syntactic structures in parallel texts, it's possible to automatically learn which features of syntactic agreement are most helpful in correctly disambiguating difficult parsing decisions. The resulting model is too complex for exact inference, but even a highly approximate reranking approach yields substantial gains.

2. By including alignment as a first-class component of the statistical model rather than merely as latent structure, the quality of word alignments can simultaneously be improved along with the parse trees. Furthermore, imposing an additional structural constraint on the alignment makes it possible to develop a variational approximate inference scheme that is more principled and accurate than the reranking approach

required by the previous model. Finally, modeling parse trees along with alignments in this way yields improved annotations that are *much* more useful for the goal of improving downstream syntactic MT.

3. If the end goal of your improved annotations is primarily to improve syntactic MT, fidelity to treebank annotation guidelines is not always the best predictor of success. Our most successful approach from the perspective of MT performance was to ignore gold-standard parses and directly optimize the agreement between the parse trees and word alignments. Our specific method also had the benefit of relying on deterministic tree transformations that are extremely fast to apply, thus resulting in a negligible additional computational cost when syntactically annotating an entire MT training corpus (a process that can be quite expensive when using complex statistical models).

6.1 Future Directions

While the main findings of this thesis are an important first step in understanding the relationship between syntactic annotation quality and machine translation performance, the results presented here raise some additional questions that bear further investigation.

One slightly curious finding has to do with the disparity between tuning and test performance. As we predicted, the new syntactic annotations resulted in higher BLEU scores on both the tuning and test data. However, in general, the tuning set improvements were quite a bit higher than those on the test set. This in itself is a relatively banal finding – optimizer overfitting is hardly an uncommon phenonomenon – but Table 4.3 shows a different pattern of results from experiments where we improved the word alignments and held the trees constant. In these word alignment experiments, which are otherwise identical to the other MT experiments in this thesis, we saw tuning and test set improvements that were much closer together in magnitude, suggesting that the overfitting effect is stronger when parses are improved than when word alignments are improved. These results alone are probably not enough to be conclusive, but given the large recent interest in methods for overcoming MT optimizer instability (see for example Chiang et al., 2008; Macherey et al., 2008; Pauls et al., 2009; Clark et al., 2011) , it seems to be worth investigating the interaction between parameter optimization and syntactic MT specifically.

The final result of this thesis was the somewhat disappointing finding that the two basic approaches presented (statistical modeling to improve parser performance and tree transformations to improve agreement with alignements) do not automatically stack together to achieve even stronger MT performance. One possibility for this result is that the agreement score metric we defined is designed specifically for isolating problems that appear in monolingual parses. When the starting point is parses that were generated from a joint model, the initial agreement is already much higher, so perhaps the remaining disagreements that are relevant to MT performance are not adequately captured by continuing to optimize agreement score. However, we do see in Table 5.7 that the annotations with the highest agreement score (Joint PA + Transformation) also yield the highest BLEU score on the *tuning* set,[1] so another possibility is that we're just running up against the limits of the parameter optimization, as suggested above. A related possibility is that the parameterization of the MT system is not rich enough to fully capture the effects of higher agreement, so perhaps incorporating additional feature engineering a la Chiang et al. (2009) would be beneficial.

Finally, one simplifying assumption made throughout this work is that syntactic information is provided to the MT system via one-best syntactic parse trees. We've shown that by improving the ways in which parsing ambiguities are resolved, we can achive better MT performance. As discussed in Chapter 1, though, there has also been a great deal of investigation into an alternative approach of simply maintaining a higher degree of ambiguity through the MT pipeline, especially by retaining entire parse forests rather than just

[1]Albeit with only a very slight advantage over the next-highest score that could also be due to chance.

one-best trees (Mi and Huang, 2008; Zhang et al., 2009). A natural extension of the current work is to see if the techniques described here can be applied in a more highly ambiguous setting, for example learning to apply contextually appropriate transformations that operate on structures within a parse forest rather than just on a single tree.

6.2 Impact

Clearly, the use of linguistic syntax is an important and powerful way of improving the quality of machine translation systems. And in recent years, incorporation of syntactic information has become much more widespread: the two most popular open-source MT toolkits, Moses (Koehn et al., 2007) and Joshua (Li et al., 2009), have both recently added syntactic pipelines to their basic functionality, and Google Translate, the world's most popular public-facing translation service, has also begun using syntactic information to train production models for some language pairs (Zhang et al., 2011). As the impact of syntax on MT grows and syntactic MT systems become more widespread and diverse, it is becoming more and more important that we have general-purpose techniques to produce appropriate syntactic annotations for training *any* syntactically-informed MT system. Taken together, this work provides several positive steps towards this goal, and we hope and anticipate that continued efforts along these lines will yield even greater rewards.

Bibliography

Anthony Aue, Arul Menezes, Bob Moore, Chris Quirk, and Eric Ringger. Statistical machine translation using labeled semantic dependency graphs. In *TMI*, 2004.

Taylor Berg-Kirkpatrick and Dan Klein. Phylogenetic grammar induction. In *ACL*, 2010.

Ann Bies, Martha Palmer, Justin Mott, and Colin Warner. English Chinese translation treebank v 1.0. Web download, 2007. LDC2007T02.

Daniel M. Bikel and David Chiang. Two statistical parsing models applied to the chinese treebank. In *Second Chinese Language Processing Workshop*, 2000.

Alexandra Birch, Chris Callison-Burch, and Miles Osborne. Constraining the phrase-based, joint probability statistical translation model. In *AMTA*, 2006.

William J. Black and Argyrios Vasilakopoulos. Language independent named entity classification by modified transformation-based learning and by decision tree induction. In *COLING*, 2002.

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. A gibbs sampler for phrasal synchronous grammar induction. In *ACL-IJCNLP*, 2009.

Eric Brill. A simple rule-based part of speech tagger. In *Proceedings of the workshop on Speech and Natural Language*, 1992.

Eric Brill. Automatic grammar induction and parsing free text: A transformation-based approach. In *ACL*, 1993.

Eric Brill. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.

Eric Brill and Philip Resnik. A transformation-based approach to prepositional phrase attachment disambiguation. In *COLING*, 1994.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–311, 1994.

David Burkett and Dan Klein. Two languages are better than one (for syntactic parsing). In *EMNLP*, 2008.

David Burkett and Dan Klein. Fast inference in phrase extraction models with belief propagation. In *NAACL:HLT*, 2012a.

David Burkett and Dan Klein. Transforming trees to improve syntactic convergence. In *EMNLP*, 2012b.

David Burkett, John Blitzer, and Dan Klein. Joint parsing and alignment with weakly synchronized grammars. In *NAACL:HLT*, 2010.

Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *ACL*, 2005.

Colin Cherry and Dekang Lin. Inversion transduction grammar for joint phrasal translation modeling. In *NAACL Workshop on Syntax and Structure in Statistical Translation*, 2007.

David Chiang. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2): 201–228, 2007.

David Chiang. Learning to translate with source and target syntax. In *ACL*, 2010.

David Chiang, Yuval Marton, and Philip Resnik. Online large-margin training of syntactic and structural translation features. In *EMNLP*, 2008.

David Chiang, Kevin Knight, and Wei Wang. 11,001 new features for statistical machine translation. In *NAACL-HLT*, 2009.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. Better hypothesis testing for statistical machine translation: controlling for optimizer instability. In *ACL:HLT*, 2011.

Michael Collins. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637, 2003.

John DeNero and Dan Klein. Tailoring word alignments to syntactic machine translation. In *ACL*, 2007.

John DeNero and Dan Klein. Discriminative modeling of extraction sets for machine translation. In *ACL*, 2010.

John DeNero, Alexandre Bouchard-Côté, and Dan Klein. Sampling alignment structure under a Bayesian translation model. In *EMNLP*, 2008.

Yuan Ding and Martha Palmer. Machine translation using probabilistic synchronous dependency insertion grammars. In *ACL*, 2005.

Jason Eisner. Learning non-isomorphic tree mappings for machine translation. In *ACL*, 2003.

Victoria Fossum, Kevin Knight, and Steven Abney. Using syntax to improve word alignment for syntax-based statistical machine translation. In *ACL MT Workshop*, 2008.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What's in a translation rule? In *HLT-NAACL*, 2004.

Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. Better word alignments with supervised ITG models. In *ACL-IJCNLP*, 2009.

Liang Huang. Forest reranking: Discriminative parsing with non-local features. In *ACL:HLT*, 2008.

Liang Huang, Kevin Knight, and Aravind Joshi. Statistical syntax-directed translation with extended domain of locality. In *HLT-NAACL*, 2006.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. Bootstrapping parsers via syntactic projection across parallel texts. *Special Issue of the Journal of Natural Language Engineering on Parallel Texts*, 11(3):311–325, 2005.

Jason Katz-Brown, Slav Petrov, Ryan McDonald, Franz Och, David Talbot, Hiroshi Ichikawa, Masakazu Seno, and Hideto Kazawa. Training a parser for machine translation reordering. In *EMNLP*, 2011.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *ACL*, 2003.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *ACL*, 2007.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 135–139, Athens, Greece, March 2009. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W/W09/W09-0x24`.

Percy Liang, Ben Taskar, and Dan Klein. Alignment by agreement. In *HLT-NAACL*, 2006.

Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. Lattice-based minimum error rate training for statistical machine translation. In *EMNLP*, 2008.

Daniel Marcu and Daniel Wong. A phrase-based, joint probability model for statistical machine translation. In *EMNLP*, 2002.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2): 313–330, 1993.

Yuval Marton and Philip Resnik. Soft syntactic constraints for hierarchical phrase-based translation. In *ACL:HLT*, 2008.

Takuya Matsuzaki, Yusuki Miyao, and Jun'ichi Tsujii. Probabilistic CFG with latent annotations. In *ACL*, 2005.

Jon May and Kevin Knight. Syntactic re-alignment models for machine translation. In *EMNLP*, 2007.

Haitao Mi and Liang Huang. Forest-based translation rule extraction. In *EMNLP*, 2008.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999. ISBN 0387987932.

Franz Josef Och. Miminal error rate training in statistical machine translation. In *ACL*, 2003.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. Syntax for statistical machine translation. Technical report, CLSP, Johns Hopkins University, 2003.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. Research report, IBM, 2001. RC22176.

Adam Pauls, John DeNero, and Dan Klein. Consensus training for consensus decoding in machine translation. In *EMNLP*, 2009.

Slav Petrov and Dan Klein. Improved inference for unlexicalized parsing. In *HLT-NAACL*, 2007.

Chris Quirk, Arul Menezes, and Colin Cherry. Dependency treelet translation: Syntactically informed phrasal SMT. In *ACL*, 2005.

Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. In *ACL Workshop on Very Large Corpora*, 1995.

Ken Samuel, Sandra Carberry, and K. Vijay-Shanker. Dialogue act tagging with transformation-based learning. In *COLING*, 1998.

Lawrence Saul and Michael Jordan. Exploiting tractable substructures in intractable networks. In *NIPS 1995*, 1996.

Libin Shen, Jinxi Xu, and Ralph Weischedel. A new string-to-dependency machine translation algorithm with a target dependency language model. In *ACL:HLT*, 2008.

Stuart M. Shieber and Yves Schabes. Synchronous tree-adjoining grammars. In *ACL*, 1990.

David A. Smith and Jason Eisner. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *HLT-NAACL*, 2006.

David A. Smith and Jason Eisner. Parser adaptation and projection with quasi-synchronous grammar features. In *EMNLP*, 2009.

David A. Smith and Noah A. Smith. Bilingual parsing with factored estimation: using English to parse Korean. In *EMNLP*, 2004.

Benjamin Snyder, Tahira Naseem, and Regina Barzilay. Unsupervised multilingual grammar induction. In *ACL-ICJNLP*, 2009.

Ben Taskar, Simon Lacoste-Julien, and Dan Klein. A discriminative matching approach to word alignment. In *EMNLP*, 2005.

Leslie G. Valiant. The complexity of computing the permanent. In *Theoretical Computer Science 8*, 1979.

Martin J Wainwright and Michael I Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc., Hanover, MA, USA, 2008. ISBN 1601981848, 9781601981844.

Wen Wang, Andreas Stolcke, and Jing Zheng. Reranking machine translation hypotheses with structured and web-based language models. In *IEEE ASRU Workshop*, 2007.

Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404, 1997.

Fei Xia, Martha Palmer, Nianwen Xue, Mary Ellen Okurowski, John Kovarik, Fu dong Chiou, Shizhe Huang, Tony Kroch, and Mitch Marcus. Developing guidelines and ensuring consistency for chinese text annotation. In *LREC*, 2000.

Eric P. Xing, Michael I. Jordan, and Stuart Russell. Graph partition strategies for generalized mean field inference. In *UAI*, 2004.

Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2): 207–238, 2005.

Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *ACL*, 2001.

Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. Bayesian learning of non-compositional phrases with synchronous parsing. In *ACL:HLT*, 2008.

Hao Zhang, Licheng Fang, Peng Xu, and Xiaoyun Wu. Binarized forest to string translation. In *ACL:HLT*, 2011.

Hui Zhang, Min Zhang, Haizhou Li, Aiti Aw, and Chew Lim Tan. Forest-based tree sequence to string translation model. In *ACL-IJCNLP*, 2009.

Bing Zhao, Young-Suk Lee, Xiaoqiang Luo, and Liu Li. Learning to transform and select elementary trees for improved syntax-based machine translations. In *ACL:HLT*, 2011.

Andreas Zollmann, Ashish Venugopal, Stephan Vogel, and Alex Waibel. The CMU-AKA syntax augmented machine translation system for IWSLT-06. In *IWSLT*, 2006.