

Integrating Parsing and Word Alignment in Syntax-Based Machine Translation

by

Victoria L. Fossum

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2010

Doctoral Committee:

Professor Steven P. Abney, Chair

Professor Martha E. Pollack

Professor Dragomir R. Radev

Professor Kevin Knight, University of Southern California/Information Sciences Institute

© Victoria L. Fossum 2010
All Rights Reserved

Table of Contents

List of Figures	v
List of Tables	vii
Abstract	ix
Chapter	
1 Introduction	1
1.1 Motivation	1
1.2 Background	11
1.2.1 Statistical Machine Translation	11
1.2.2 Syntax-Based Statistical Machine Translation	12
1.2.3 Word Alignment	17
1.2.4 Parsing	21
1.2.5 Rule Extraction	23
1.2.6 Parameter Estimation	28
1.2.7 Tuning with Minimum Error Rate Training	29
1.2.8 Decoding	29
1.2.9 Evaluation Metrics	30
1.3 Thesis Outline	35
2 Improving Parsing Accuracy	38
2.1 Background	39
2.2 Combining Constituent Parsers of English	41
2.2.1 Parse Selection	43
2.2.2 Parse Hybridization	50
2.2.3 Methods	53
2.2.4 Summary of Contributions	60
2.3 Using Bilingual Word Alignments to Resolve English Syntactic Ambiguity	61
2.3.1 Related Work	63
2.3.2 Our Work	66
2.3.3 Bilingual Alignments and PP-attachment Ambiguity	66

2.3.4	Methods	69
2.3.5	Results	71
2.3.6	Conclusion	72
2.4	Summary of Contributions	73
3	Improving Word Alignment Accuracy Using Parsing	75
3.1	Background	75
3.1.1	Related Work	78
3.2	Methods	80
3.2.1	Link Deletion Algorithm	80
3.2.2	Features	81
3.2.3	Constraints	85
3.2.4	Discriminative Training Using Averaged Perceptron	86
3.3	Experimental Setup	89
3.3.1	Data Sets	89
3.3.2	Evaluation Metrics	89
3.3.3	Experiments	91
3.4	Results	92
3.4.1	Chinese-English	93
3.4.2	Arabic-English	93
3.4.3	Varying the Relative Contribution of Precision and Recall to Alignment F-Measure	93
3.5	Discussion	94
3.5.1	Size of Discriminative Training Set	94
3.5.2	Effect of Link Deletion on Extracted Rules	96
3.6	Summary of Contributions	96
4	Using Parsing and Word Alignment to Improve Accuracy of Both Processes Simultaneously	98
4.1	Background	98
4.1.1	Related Work	100
4.1.2	Our Work	103
4.2	Methods	103
4.2.1	Features	104
4.2.2	Discriminative Training	110
4.2.3	Linear Regression	111
4.3	Experimental Setup	111
4.4	Results	111
4.5	Summary of Contributions	113
5	Conclusion	114
5.1	Future Work	116
5.1.1	Using Bilingual Word Alignments to Improve Parsing Accuracy	116
5.1.2	Using Parsing to Improve Word Alignment Accuracy	116

5.1.3	Using Parsing and Word Alignment to Improve Accuracy of Both Processes Simultaneously	117
5.1.4	A New Evaluation Metric: Rule F-Measure	117
Appendix	119
Bibliography	123

List of Figures

Figure

1.1	Example Alignment and Extracted Translation Rules	4
1.2	Incorrect Alignment Link and Extracted Translation Rules	6
1.3	Example Parse and Extracted Translation Rules	9
1.4	Parse with PP-Attachment Error and Extracted Translation Rules . .	9
1.5	Example Generation	14
1.6	Example Decoding	15
1.7	Syntax-Based Statistical Machine Translation Pipeline	17
1.8	Word Alignment	18
1.9	Parse Tree	22
1.10	Alignment and Parse Graph	24
1.11	Alignment and Parse Graph with Spans and Complement Spans. . . .	25
1.12	Alignment and Parse Graph with Frontier Nodes and Extracted Minimal Rules	28
1.13	Alignment and Parse Graph with Example of Extracted Composed Rule	28
1.14	Before and After Fully-Connecting an Alignment	32
1.15	Computing Rule F-Measure	36
2.1	PP-attachment ambiguity in English	39
2.2	Parse with PP-Attachment Error and Extracted Translation Rules . .	41
2.3	Parse with Correct PP-Attachment and Extracted Translation Rules	42
2.4	Output of Charniak Parser	54
2.5	Output of Constituent Recombination	54
2.6	Output of Context-Free Production Recombination	55
2.7	PP-attachment ambiguity in English	62

2.8	Resolving PP-attachment ambiguity using Chinese-English word alignments	62
2.9	PP-attachment ambiguity in English	68
2.10	Resolving PP-attachment ambiguity using Chinese-English word alignments	68
3.1	Impact of incorrect alignment links upon extraction of tree-to-string transducer rules.	77
4.1	Example (Parse, Alignment) Pair and Extracted Rules	105

List of Tables

Table

1.1	Impact of Alignment and Parse Tree Quality on Precision, Recall, and F-Measure of Extracted Syntax-Based Translation Rules	8
2.1	Oracle Results for Parse Selection and Parse Hybridization	43
2.2	Data Sets Used in Parser Combination: Parsing Experiments	55
2.3	F-Measure of 1-best Output of Individual Parsers	55
2.4	Precision, Recall, and F-score Results from Parse Selection	56
2.5	Precision, Recall, and F-score Results from Constituent Recombination	56
2.6	Precision, Recall, and F-score Results from Context-Free Production Recombination	56
2.7	Data Sets Used in Parser Combination: Syntax-Based MT Experiments.	57
2.8	BLEU Scores Using Parser Combination in Arabic-English Syntax-Based MT with Minimal Rules and No Binarization	58
2.9	BLEU Scores Using Parser Combination in Arabic-English Syntax-Based MT with Composed Rules and Head-Out Binarization	58
2.10	BLEU Scores Using Parser Combination in Arabic-English Syntax-Based MT with Composed Rules and EM Binarization	59
2.11	PP-Attachment Accuracy of Baseline Collins Parser	72
2.12	PP-attachment Accuracy of Perceptron Classifier vs. Baseline Collins Parser	72
2.13	Feature Ablation: Accuracy of PP-Attachment Classifier with Individual Features Removed	73
3.1	Data Sets Used in Alignment Link Deletion: Alignment Experiments.	91
3.2	Data Sets Used in Alignment Link Deletion: Translation Experiments.	91
3.3	Results of Link Deletion.	94

3.4	Rule Precision, Recall, and F-Measure of Rules Extracted from 400 Sentence Pairs of Chinese-English data	95
4.1	Survey of Recent Work in Exploring Multiple Alignments, Parses, or Strings in Training a Machine Translation System	102
4.2	Data Sets Used in (Parse, Alignment) Pair Reranking	111
4.3	Results of Oracle (Parse, Alignment) Pair Reranking Experiments on 2248 Sentences of Arabic-English Parallel Data.	112
4.4	Results of (Parse, Alignment) Pair Reranking with 100-best Alignments and 10-best Parses.	113
A.1	Arabic-English Data Sets	121
A.2	Chinese-English Data Sets	122

Abstract

Integrating Parsing and Word Alignment in Syntax-Based Machine Translation

by

Victoria L. Fossum

Chair: Steven P. Abney

Training a state-of-the-art syntax-based statistical machine translation (MT) system to translate from a *source* language into a *target* language requires a large *parallel corpus* of example sentences in the source language translated into the target language by a human; a *word alignment* (word-to-word correspondence between each source-target sentence pair); and a *parse tree* (syntactic representation) of each sentence in the source language, target language, or both. From these resources, the string-to-tree syntax-based MT system used in this thesis [34, 33] acquires rules governing the process of translating a source string into a target parse tree. After training, these rules are used to translate previously unseen source sentences into the target language.

The parallel corpora used to train current state-of-the-art systems are too large for manual annotation; instead, word alignment and parsing must be performed automatically. There are two problems with current approaches to automatic word alignment and parsing. First, both processes introduce errors that propagate through the pipeline. Improving the accuracy of either process can therefore improve translation

quality. Second, the two processes are typically performed independently. Since each process produces constraints that can be used to guide the other, we can improve the accuracy of both processes by integrating them more closely. Word alignment and parsing jointly determine the set of translation rules acquired by a system during training, so it is desirable to optimize them both in order to produce the best translation rules possible.

In this thesis, we address these two problems as follows. First, we recombine the output of multiple parsers, improving parse and translation quality. Second, we use features of the word alignment to correct parse errors. Third, we use features of the parse trees to correct word alignment errors, improving alignment and translation quality. Fourth, we integrate word alignment and parsing by producing n -best lists of candidates for each process, and discriminatively reranking (word alignment/parse tree) pairs to optimize the quality of the extracted translation rules.

Our results demonstrate that integrating word alignment and parsing improves the accuracy of each process, and in some cases improves translation quality relative to a state-of-the-art syntax-based MT system.

Chapter 1

Introduction

1.1 Motivation

Machine translation (MT) is the process of translating from one language to another automatically. Machine translation systems must answer the following question: given a sentence f in some source language (such as Chinese or Arabic), what is the most likely translation e of that sentence in some target language (such as English)? In order to produce a correct translation, machine translation systems must succeed in two areas: first, they must choose the correct words in the target language; and second, they must output those words in the correct order.

If natural language were unambiguous (as, for example, programming languages are designed to be), there would be only one possible interpretation of the source sentence, and only one possible translation of that source sentence in the target language. Unfortunately for our purposes, natural language is rife with ambiguity. Many words have multiple possible meanings (lexical ambiguity), and many sentences have multiple possible grammatical interpretations (syntactic ambiguity). Moreover, ambiguity is not necessarily preserved across languages; constructions that are unambiguous in one language may be ambiguous in another.

In order to handle the ambiguity inherent in natural language, the human brain relies upon a sophisticated database of world knowledge. Using such word knowledge, humans are able to quickly and accurately select the most likely interpretation of a sentence. In the absence of such world knowledge, a machine translation system can use probabilistic inference in order to resolve ambiguity. Such a system is called a statistical machine translation system. These systems must learn, through exposure to many examples of translations from the source language to the target language, the types of constructions that occur in both languages; the frequencies with which such constructions occur; and the translational correspondences between them.

Because the set of patterns that a system must acquire in order to adequately model translation is so vast, it is infeasible for a human to describe the set of such patterns explicitly and completely. Instead, statistical machine translation systems must acquire such patterns automatically from a large parallel corpus consisting of sentences in the source language and their translations in the target language. The factors that distinguish different approaches to statistical machine translation include: the type of patterns acquired from such a parallel corpus during training (the translation model); the methods used to associate probabilities with each pattern (parameter estimation); and the way in which the patterns are applied to unseen source sentences during testing to produce the most likely target translations (decoding).

In word- or phrase-based translation models, the fundamental pattern, or unit of translational correspondence, consists of a *word* or *multiple words* in the source language; a word or multiple words in the target language; and a word-to-word correspondence between them. In syntax-based translation models, the fundamental unit of translational correspondence is modified to consist of a *tree* rather than a word or phrase in the source language, the target language, or both. Depending on whether syntactic trees are used on the source side, the target side, or both, we refer to the system as tree-to-string, string-to-tree, or tree-to-tree.

Whatever the type of translational correspondence being modelled, all statistical translation systems face the following question: how can such patterns, or rules, be acquired automatically from a parallel corpus of source-target sentence pairs? In other words, given a set of sentences in the source language and their translations in the target language, how can we determine translational correspondences within a sentence pair? Nearly all statistical machine translation systems rely upon a *word alignment*, or an explicit word-to-word correspondence, between words in each source-target sentence pair in order to learn these translational correspondences. In addition, syntax-based systems require a syntactic analysis, or *parse tree*, of each source sentence, target sentence, or both. Using these word alignments and parse trees, the system can extract patterns of translational correspondence that can later be used to translate unseen sentences.

Figure 1.1 illustrates an example source-target sentence pair, word alignment, and target parse tree, along with the translation rules extracted from this sentence pair.¹ Each rule’s right hand side denotes the foreign string to which this rule applies; each rule’s left hand side denotes the English tree that results from this rule’s application.

The rule:

allows an English subtree with a root node IN and a child node “from” to be translated to the Chinese word 从. Rules can contain variables as well as lexical items on both the left and right hand sides. For example, the rule:

contains a variable, x_0 , that appears on the left and right hand sides of the rule. The variable x_0 is co-indexed on the left and right hand sides of the rule to indicate a translational correspondence. The left hand side of the rule specifies an English tree fragment containing a node $x_0:PP$, where $x_0:PP$ can be *any* PP. The right hand side of the rule specifies a Chinese sequence of words x_0 出发, where x_0 must be a translation of the English PP.

¹We describe rule extraction in detail in Chapter 1, Section 1.2.5.

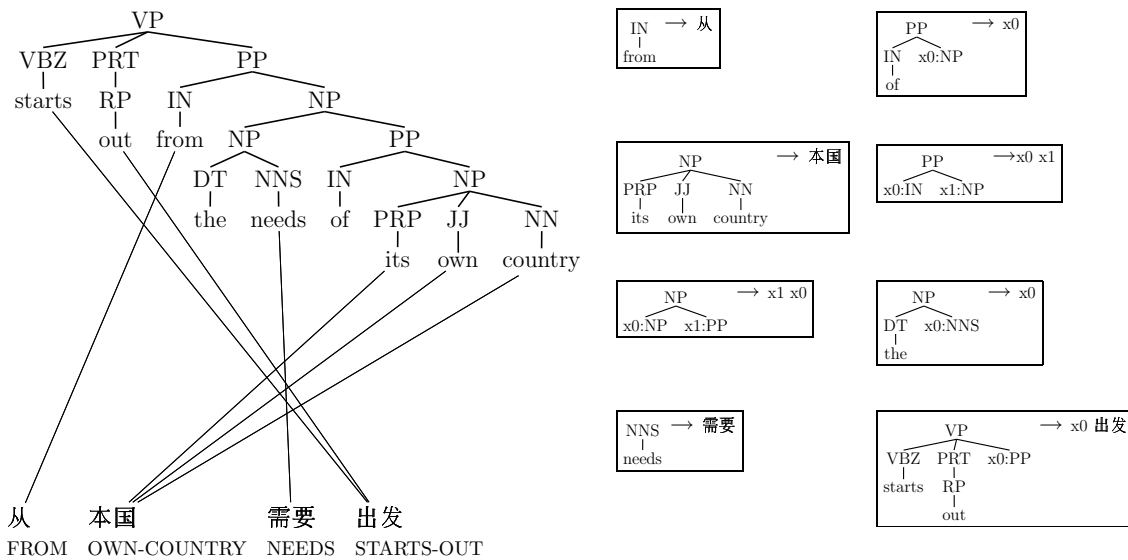
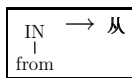
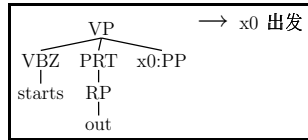


Figure 1.1: Example Alignment and Extracted Translation Rules



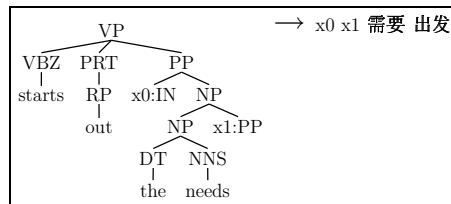
Because word alignments and syntactic parse trees define the fundamental unit of translation for a syntax-based system, it is crucial that these annotations be as accurate as possible. Unfortunately, because current state-of-the-art systems require hundreds of thousands or even millions of parallel sentence pairs of training data, it is infeasible to produce these annotations manually. Instead, they must be generated automatically. There are two problems with the way that automatic word alignment and parsing are currently performed in state-of-the-art syntax-based statistical machine translation systems.

The first problem is that automatic word alignments and automatic parse trees are prone to errors; these errors then cause the system to extract potentially incorrect rules of translational correspondence. In some cases, these errors may cause the system to infer a translational equivalence that simply does not exist between the two languages. In other cases, the effect is more subtle: these errors may force the system to extract rules that are not incorrect, but which fail to identify *minimal* units

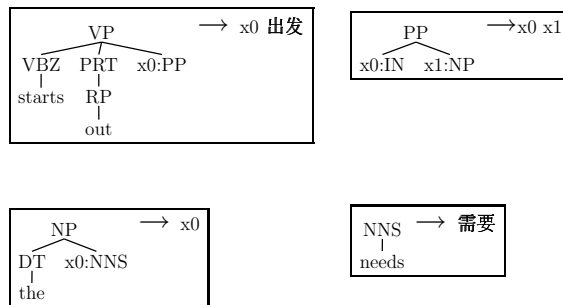


of translational correspondence. These rules incorporate unnecessarily large amounts of context, and thus have limited applicability to unseen sentences at test time.

Figure 1.2 illustrates an error in Chinese-English word alignment, and the effect it has upon the translation rules extracted by a syntax-based machine translation system. In Figure 1.2, the dotted link between “needs” and 出发 is incorrect. The largest rule in this set:



can be applied to Chinese strings containing the Chinese translation of an English IN; the Chinese translation of an English PP; and the Chinese words 需要出发. If a Chinese sentence matches this context exactly, it can be translated to the English subtree shown on the left hand side of the translation rule. This rule includes more context than, for example, the following rules extracted from the correct alignment shown in Figure 1.1:



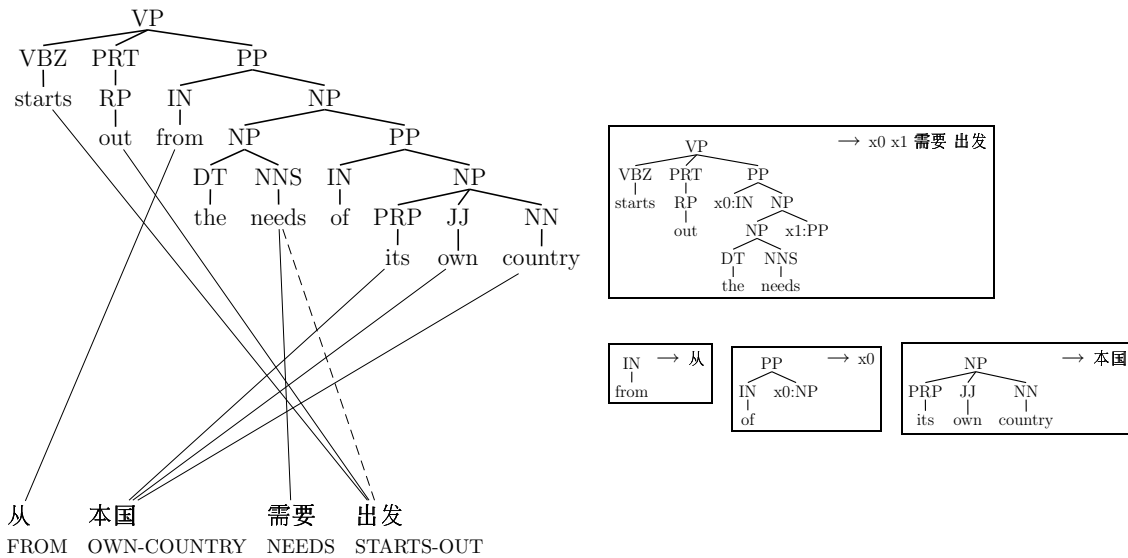
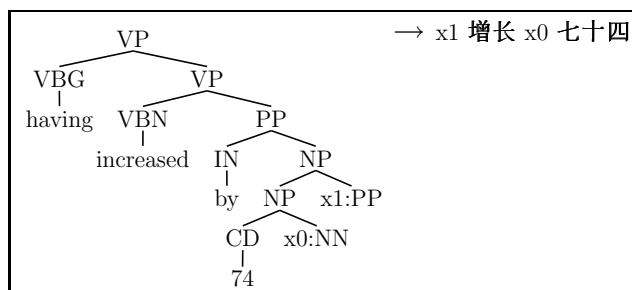


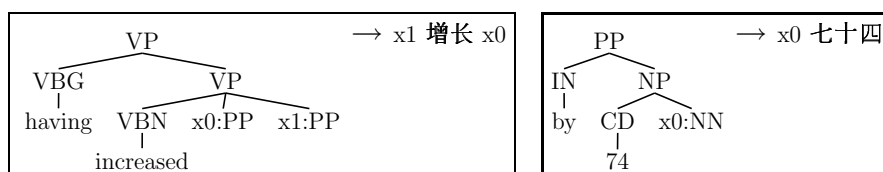
Figure 1.2: Incorrect Alignment Link and Extracted Translation Rules

which cover the same Chinese and English data but are more modular, can be applied in more contexts, and reflect truly *minimal* units of translational correspondence.

Errors in English parse trees can create similar problems for rule extraction. As with alignment errors, parse errors can cause the system to extract translational correspondences that are incorrect. In other cases, parse errors may cause the system to extract rules that are not incorrect, but that fail to reflect *minimal* units of translational correspondence. To illustrate, Figure 1.3 displays an example sentence pair with a correct English parse tree, and the extracted rules. By contrast, Figure 1.4 illustrates an error in the English parse tree (the PP headed by “compared” should actually modify the English VP headed by “increased”, and not the English NP “74 percent”), and the effect it has upon the extracted rules. The largest rule in the set extracted from this incorrect English parse:



can only be applied to Chinese sentences whose context exactly matches the right hand side of the rule. This rule specifies more context than, for example, the following rules extracted from the correct English parse tree shown in Figure 1.3:



which are more modular, can be applied in more contexts, and reflect truly *minimal* units of translational correspondence.

Whether they are extracted because of alignment errors or parse errors, translation rules which contain incorrect translational correspondences or which require excessive amounts of context in order to be applicable can ultimately produce translations of sub-optimal quality when applied during testing. In order to quantify the overall impact of parse and alignment quality upon translation rule quality, we extract a gold-standard set of rules by applying the minimal rule extraction algorithm described in Galley et al. [34] to a parallel Chinese-English corpus with gold alignments and gold English parse trees. We then extract rules from the same corpus using two different sources of automatically produced alignments (GIZA++ with *union* symmetrization and GIZA++ with *refined* symmetrization)² and the Collins parser [17], and compute the precision, recall, and f-measure of the extracted rules against the gold-standard rule set.³ Table 1.1 illustrates that errors introduced by an automatic parser cause

²For more details on alignment models and symmetrization heuristics, please refer to Chapter 1, Section 1.2.3.

³For more details on how to compute rule f-measure, please refer to Chapter 1, Section 1.2.9.

		Parse					
		gold			Collins		
		P	R	F	P	R	F
Alignment	gold	100.0	100.0	100.0	73.3	76.0	74.6
	GIZA++ refined	56.0	65.6	60.4	44.0	54.5	48.7
	GIZA++ union	63.6	52.3	57.4	50.7	44.2	47.2

Table 1.1: Impact of Alignment and Parse Tree Quality on Precision, Recall, and F-Measure of Extracted Syntax-Based Translation Rules

a significant decrease in rule f-measure, from 100.0% using the gold parses to 74.6% using the automatic parses. Errors introduced by an automatic aligner cause an even greater decrease in rule f-measure (from 100.0% using gold alignments to 60.4% or 57.4% using automatic alignments). Using both automatic parses *and* automatic alignments (the settings under which we typically train a statistical MT system) causes rule f-measure to drop to 47.2%. Thus, improving parse and alignment quality has significant potential to improve the quality of extracted translation rules, and thereby also the quality of syntax-based translation.

The second problem with current approaches to word alignment and parsing for syntax-based statistical machine translation systems is that word alignment and parsing are performed independently of each other. Since each process produces constraints that can potentially be used to guide the other, we can expect to improve the accuracy of each process by integrating them more closely. For example, the PP-attachment error made by the English parser in Figure 1.4 can be corrected using the bilingual word alignment. Since PP-attachment is ambiguous in English, but not in Chinese, the word alignment provides an additional constraint upon the English syntactic analysis which can be used to disambiguate the PP-attachment in English. Similarly, the presence of the incorrect alignment link “needs”-出发 in Figure 1.2 can be corrected using the English parse tree. Since “starts out” and “needs” are

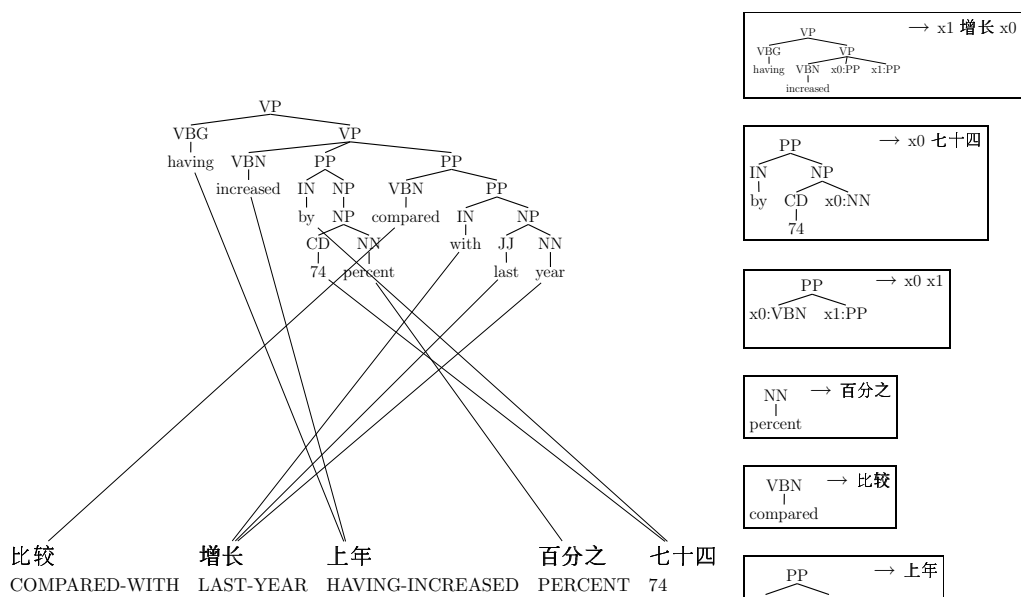


Figure 1.3: Example Parse and Extracted Translation Rules

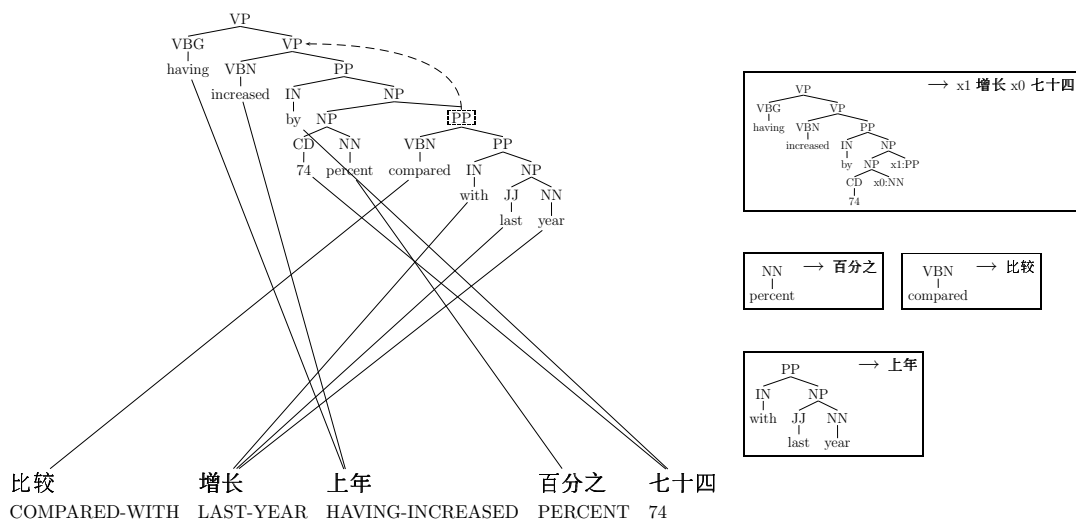


Figure 1.4: Parse with PP-Attachment Error and Extracted Translation Rules

syntactically distant from each other in the English parse tree (meaning that, if one traverses the tree from “starts out” to “needs”, many constituents must be visited along the way), and syntactically distant nodes in the English parse tree are unlikely to be aligned to the same Chinese word, we can infer that the link from “needs” to the Chinese 出发 is likely to be incorrect.

In this thesis, we address the problems of word alignment and syntactic parsing within the context of string-to-tree syntax-based machine translation. Specifically, we address the following questions:

- How can we improve the accuracy of syntactic parsing?
- How can we improve the accuracy of word alignment?
- How can we use each process to constrain the other?
- What is the impact of improving parsing and word alignment accuracy upon syntax-based machine translation?

We answer these questions as follows. First, we improve upon the accuracy of state-of-the-art constituent parsers by recombining the output of multiple parsers (Chapter 2). Second, we show that word alignments can be used to improve parsing accuracy by using bilingual Chinese-English word alignment features to identify and correct syntactic parsing errors in English prepositional phrase attachment (Chapter 2). Third, we show that parses can be used to improve word alignment precision by using syntactic features to identify and delete incorrect word alignment links (Chapter 3). Fourth, we integrate word alignment and parsing by producing n -best lists of candidates for both word alignment and parsing, and discriminatively reranking (word alignment, syntactic parse tree) pairs to optimize the quality of the extracted translation rules (Chapter 4). We evaluate the impact of these improvements upon alignment quality, parse quality, and syntax-based translation quality.

Our results demonstrate that integrating word alignment and syntactic parsing can indeed improve the accuracy of each process, and in some cases leads to an improvement in translation quality relative to a state-of-the-art syntax-based statistical machine translation system.

1.2 Background

1.2.1 Statistical Machine Translation

The goal of any statistical machine translation system is to answer the following question: given a sentence f in the source language, what is the most likely translation e of that sentence in the target language, $\operatorname{argmax}_e [P(e|f)]$?

In the noisy channel model of machine translation, we decompose $P(e|f)$ as follows using Bayes' Rule, which allows us to estimate each component model separately: $P(e|f) = P(f|e) \times \frac{P(e)}{P(f)}$. Since $P(f)$ is fixed for any given source sentence f , we can eliminate it from consideration. We are now left with:

$$\operatorname{argmax}_e [P(e|f)] = P(f|e) \times P(e)$$

$P(f|e)$ is the translation model, $P(e)$ is the language model, and searching over the space of possible target translations to compute $\operatorname{argmax}_e P(e|f)$ is the decoding problem.

In practice, most statistical MT systems do not actually optimize the quantity $P(f|e) \times P(e)$. Instead, they model the posterior probability $P(e|f)$ directly, and incorporate the language model $P(e)$ and the translation model $P(e|f)$ as features h with weights λ in a log-linear model which can be used to score each (e, f) pair:

$$P(e|f) = \frac{\exp[\sum_{m=1}^M \lambda_m \times h_m(e, f)]}{\sum_{e'} \exp[\sum_{m=1}^M \lambda_m \times h_m(e', f)]}$$

This formulation allows additional features besides the translation model and language model to be incorporated into the score of each (e, f) pair in a straightforward way. The feature weights λ_i of each feature h_i can be set discriminatively to optimize translation quality.

Language Models

Typically, the probability $P(e)$ of each possible English translation is modelled according to an n -gram language model:

$$P(e_1^I) = \prod_{i=1}^I P(e_i | e_{i-1}, e_{i-2}, \dots, e_{i-n})$$

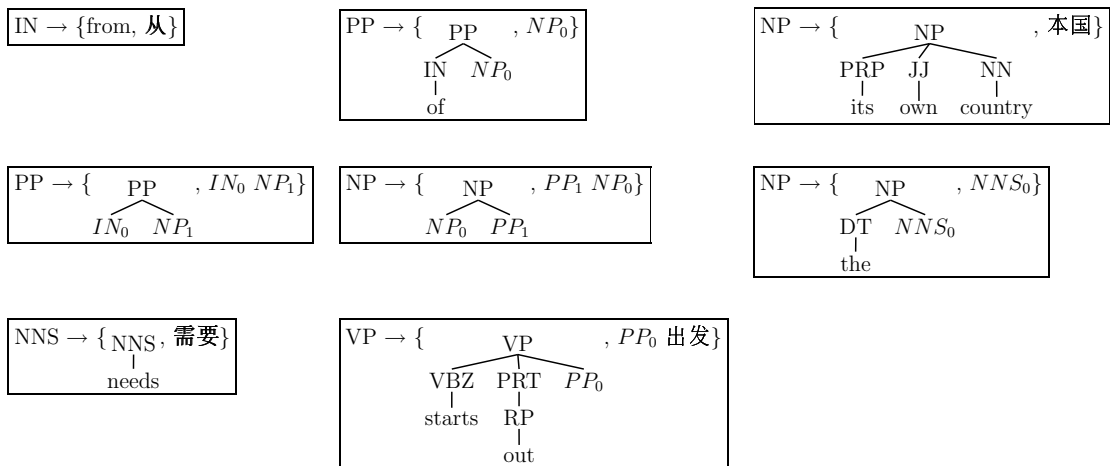
Syntax-Based Translation Models

In syntax-based models, the basic unit of translation consists of a tree fragment in one language; either a sequence of words or a tree fragment in the other language; and a word alignment between them. The benefits of incorporating syntax into MT are twofold: first, it facilitates the use of linguistically motivated syntax to guide the translation process, and second, it facilitates the modelling of long-distance re-ordering patterns. Syntax-based systems may parse the data on the target side (string-to-tree), source side (tree-to-string), or both (tree-to-tree).

1.2.2 Syntax-Based Statistical Machine Translation

String-to-Tree Syntax-Based Models

Galley et al. [34, 33] describe the syntax-based MT system which is the platform for all experiments in this thesis. Formally, this model can be thought of as a generalization of a synchronous context-free grammar (SCFG), in which a context-free grammar (CFG) generates not one, but *two* output strings. We can equivalently represent the rules shown in Figure 1.1 using SCFG notation:



Where two non-terminal symbols are co-indexed, those two symbols are translational equivalents of each other, and must be expanded simultaneously when applying an SCFG rule during generation.

For reference, Figure 1.5 illustrates the process by which an English tree and foreign string are generated under this model; Figure 1.6 illustrates how to apply these translation rules to an input foreign string during translation to construct an output English tree.

Tree-to-String Syntax-Based Models

Yamada and Knight [104] propose a tree-to-string model which applies the following operations to nodes in the source parse tree: child re-ordering, inserting words at a node, and translating leaf nodes. The parameters for each of these probability models are estimated directly from a bilingual corpus using EM (in contrast to the string-to-tree model of Galley et al. [34], in which the bilingual corpus is automatically word-aligned before translation rules are extracted). Re-orderings in the Yamada and Knight [104] model are restricted to child re-orderings, and are therefore not as powerful as the re-orderings allowed by Galley et al. [34, 33].

Huang et al. [38] and Liu et al. [66] present tree-to-string models. The model presented in Huang et al. [38] admits a linear-time dynamic programming decoding

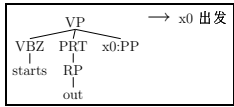
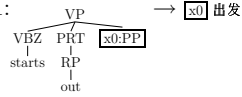
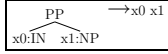
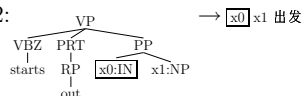
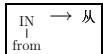
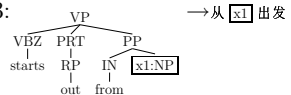
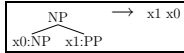
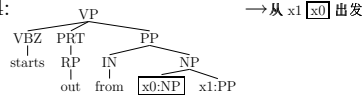
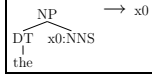
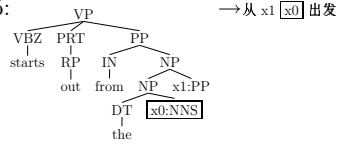
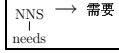
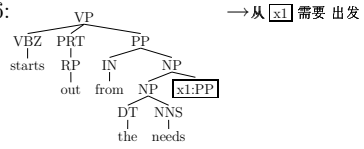
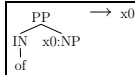
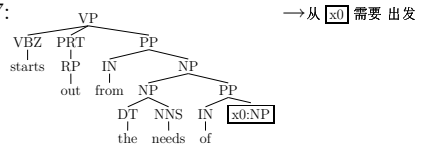
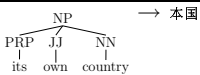
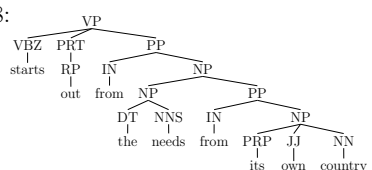
Top-Down Generation of English Tree and Foreign String	Rule to Apply Next
STEP 0: $\boxed{\text{VP}} \rightarrow \square$	
STEP 1: 	
STEP 2: 	
STEP 3: 	
STEP 4: 	
STEP 5: 	
STEP 6: 	
STEP 7: 	
STEP 8: 	

Figure 1.5: Example Generation

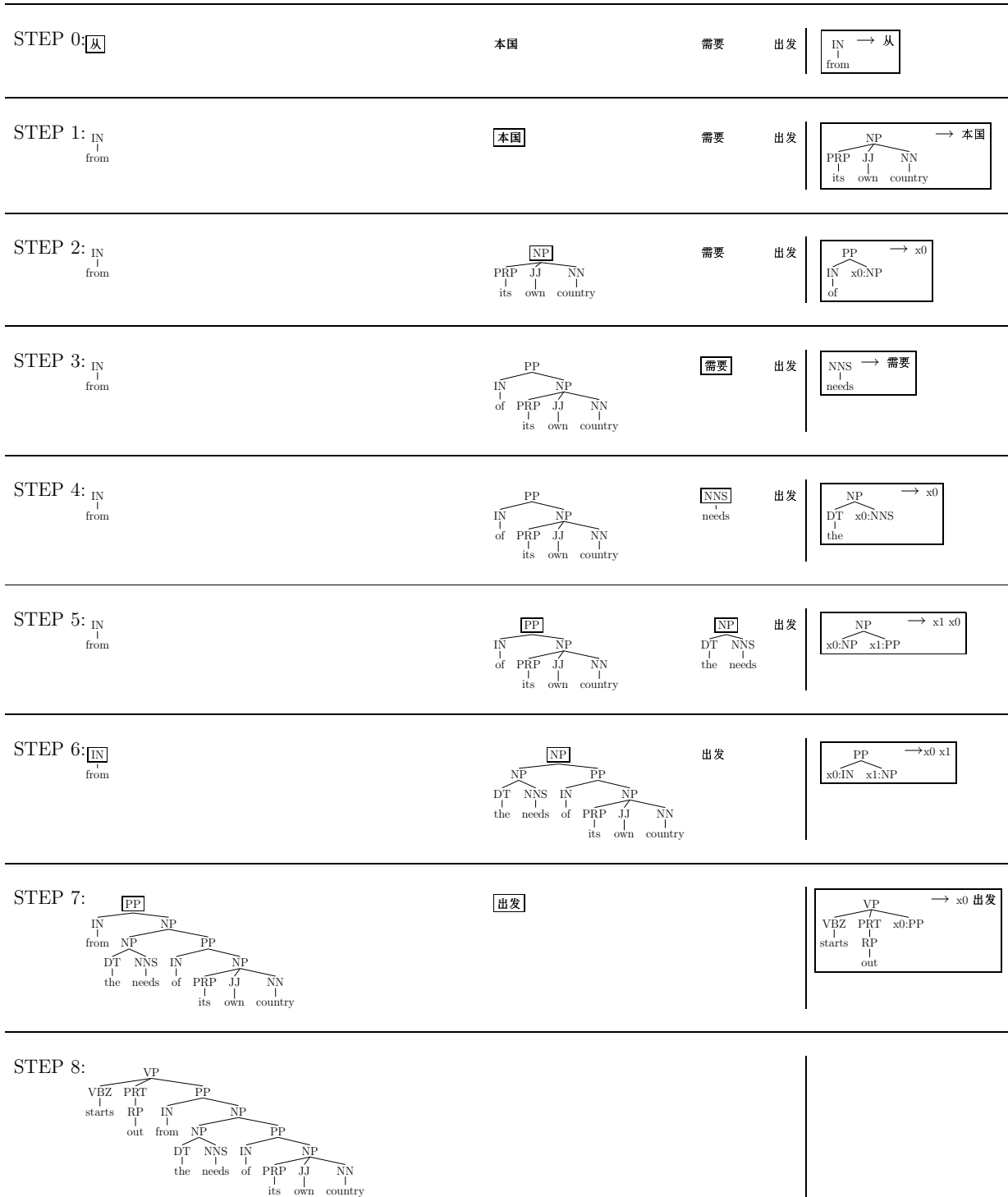


Figure 1.6: Example Decoding

algorithm, which is more efficient than the polynomial-time CKY-based decoding algorithm of the string-to-tree model described by Galley et al. [34, 33].

While decoding is more efficient in the tree-to-string models of Huang et al. [38] and Liu et al. [66] than in the string-to-tree model used in this thesis [34, 33], the string-to-tree model has an advantage over tree-to-string models in that it is typically more important to enforce syntactic constraints in the target, or output, language, than in the source, or input, language, since the source language is presumably syntactically well-formed. String-to-tree models are better able to enforce syntactic constraints in the target language, which can result in more fluent translations.

Tree-to-Tree Syntax-Based Models

The ITG (Iterative Transduction Grammar) formalism proposed by Wu [103] and the Multi-text Grammar formalism proposed by Melamed [75] cast translation as a synchronous parsing problem in the source and target languages. Chiang [13] proposes hierarchical phrase-based translation, which is a synchronous grammar formalism that extends the phrase pairs of phrase-based translation systems to hierarchical phrase pairs by substituting variables for phrase pairs that appear within other phrase pairs.

While these formalisms make use of syntactic structures in both languages, Wu [103] and Chiang [13] use a formal syntax that is not linguistically motivated, but rather induced by the structure of the parallel data. By contrast, the system of Galley et al. [34, 33] directly incorporates linguistically motivated syntactic structure in the target language.

In tree-to-tree formalisms that do make use of linguistically motivated syntax [75, 67], the task of inducing a synchronous grammar during training is significantly more expensive from a computational perspective than in the case of string-to-tree or tree-to-string systems. Furthermore, because such systems rely on parse trees in both languages, they are much less robust to parse errors than string-to-tree or tree-to-string systems.

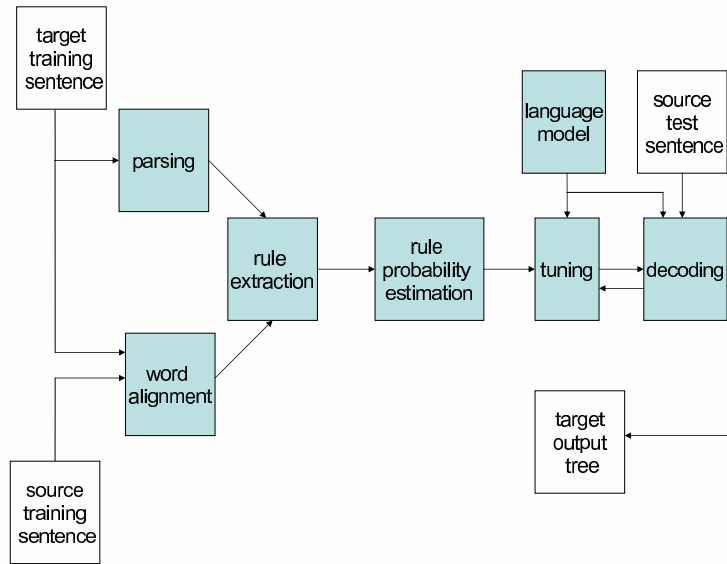


Figure 1.7: Syntax-Based Statistical Machine Translation Pipeline

In this thesis, we focus on improving parse and alignment quality for a string-to-tree syntax-based machine translation system [34, 33]. Figure 1.7 illustrates the stages of this MT pipeline.

In order to clarify the role of alignment and parsing in such a system, we first describe each component of this system in detail.

1.2.3 Word Alignment

A word alignment is a word-to-word correspondence between words in a source sentence f and the target translation of that sentence, e :

- **Word Alignment:** A word alignment A for a sentence pair (e, f) is a set of pairs (e_i, f_j) denoting indices of words $e_i \in e$ and $f_j \in f$ that are aligned.

Figure 1.8 illustrates an example word alignment.

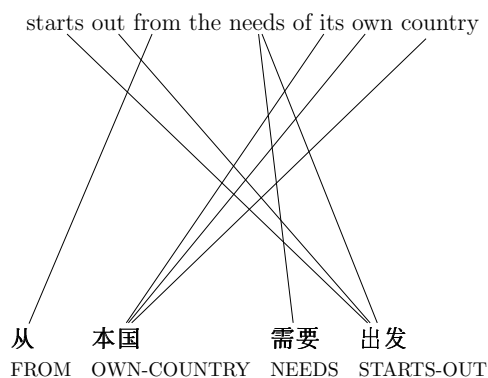


Figure 1.8: Word Alignment

IBM Models

Brown et al. [7, 8] first introduced the concept of a word alignment when they laid the foundation for statistical machine translation with the word-based IBM models for translation, and presented techniques for estimating parameters for these models. Word alignments were introduced to facilitate parameter estimation of the IBM translation models. For IBM Model 4, the generative process by which a target sentence t of length n produces a source sentence s of length m is parameterized as follows:

- **Fertility:** For each word t_i in the target sentence, select a fertility ϕ_i . This fertility is an estimate of how many words in the source sentence are likely to be “generated” by target word t_i , and is conditioned on the word type t_i .
- **Fertility of the Null Word:** For the null word t_0 , select a fertility ϕ_0 . This fertility is an estimate of the number of “spurious” words appearing in the source sentence (words that have not been generated by any particular word in the target sentence, but rather, appear as an artifact of translation), and is conditioned on the sum of the fertilities $\phi_0 \dots \phi_n$ determined in the previous step. $\sum_{i=0}^n \phi_i = m$, the length of the source determined in the previous step.

- **Translation:** For each word t_i in the target sentence, translate it into one or more words $s_j\dots s_k$ in the source language. This translation probability is conditioned on the word type t_i .
- **Distortion:** For each source word that has been generated by a target word t_i , select a position j for that source word in the source sentence. This distortion probability is conditioned on positions i and j ; sentence lengths n and m , and the positions of all other words s_k that have also been generated by the target word t_i .

Given this parameterization of $P(s|t)$ in terms of fertility probabilities, translation probabilities, and distortion probabilities, we need to estimate these probabilities from a bilingual corpus of source and target sentence pairs. In a bilingual corpus we can observe the target and source sentences, but not the intermediate steps in the generative process described above. In order to estimate the probabilities needed for Model 4, Brown et al. [8] introduce the concept of a word alignment a between s and t .

The problem of estimating the fertility, translation, and distortion probabilities in an unsupervised manner can be solved using the Expectation-Maximization (EM) algorithm [12]. During the E-step, one calculates a probability distribution over word alignments using the current model parameter values, and during the M-step, one re-estimates the model parameter values using the current probability distribution over word alignments.

In practice, computing the exact IBM Model 4 parameters from a large parallel corpus using EM is intractable, so Brown et al. [8] introduce a set of successively simpler models, Models 1-3, which are meant to be computed in series, so that the results of Model 1 are used to initialize parameter values for Model 2, and so on. Models 1 and 2 can be computed exactly, but Models 3 and 4 require the following approximations: during the M-step, the model parameter values are re-estimated

using only the most likely word alignments according to the current model. Also, the search space of alignment links is restricted to 1-to-many links in the source-to-target direction in order to render the search problem tractable. Vogel et al. [101] propose an alternative approach to the distortion model of IBM Model 4 using a Hidden Markov Model (HMM). In the HMM model, the placement of each source word s_j is conditioned not upon absolute positions, but upon the relative position of s_j with respect to the previously placed source word.

Most current state-of-the-art statistical machine translation systems rely upon word alignments to define an initial translational correspondence between the source and target sentences. IBM Model 4 (in its open-source implementation, GIZA++ [85]) remains the predominant approach to word alignment to date. We use GIZA++ word alignments as our baseline in all translation experiments in this thesis.

Symmetrization Heuristics

IBM Models 1-4 [7, 8] and the HMM model based on Vogel [101] form the basis of the open-source GIZA++ word alignment software [85] which is perhaps the most widely used statistical alignment algorithm in the current MT literature. Because of an asymmetry in the generative process of the IBM Models that was introduced to render the search over the space of alignments tractable, a single target word t_i can generate multiple source words s_j , but the converse is not true: a single source word s_j cannot be generated by multiple target words t_i . Thus, the model permits one-to-one and one-to-many alignments in the target-to-source direction, but does not permit one-to-many alignments in the source-to-target direction.

To remedy this asymmetry, allowing for one-to-many source-to-target alignments and many-to-many alignments, a standard approach in the alignments literature is to run IBM Models 1-4 twice, once in the source-to-target direction and once in the target-to-source direction, then combine the two sets of alignments using one of

several possible “symmetrization” heuristics. Och and Ney [85] present three such heuristics for combining two alignments A_1 and A_2 to form a symmetrized alignment A :

- *intersection*(A_1, A_2) : $A = A_1 \cap A_2$
- *union*(A_1, A_2) : $A = A_1 \cup A_2$
- *refined*(A_1, A_2): (See Algorithm 1)

Algorithm 1: Refined Alignment Symmetrization

input : A source-to-target alignment A_1 and a target-to-source alignment A_2
output: A symmetrized alignment A , where $(A_1 \cap A_2) \subseteq A$ and $A \subseteq (A_1 \cup A_2)$

Initialize $A = A_1 \cap A_2$;
Consider each link $(e_i, f_j) \in (A_1 \cup A_2)$;
if $(e_i \notin A) \wedge (f_j \notin A)$ **then**
| Add link (e_i, f_j) to A ;
end
else if $((e_i, f_j)$ has a horizontal neighbor (e_{i-1}, f_j) or (e_{i+1}, f_j) , or a vertical neighbor (e_i, f_{j-1}) or (e_i, f_{j+1}) , that is already in A), and $(A \cup (e_i, f_j))$ does not contain any links with both horizontal and vertical neighbors) **then**
| Add link (e_i, f_j) to A ;
end

1.2.4 Parsing

Syntactic parsing is the task of assigning a syntactic tree structure t to a sentence s . An example parse tree is shown in Figure 1.9. In statistical parsing, the parameters of the parsing model $P(t|s)$ are learned automatically from a training corpus consisting of sentences and their associated syntactic trees. The parser can then determine the most likely tree structure $\hat{t} = \arg \max_t P(t|s)$ for an unseen sentence s according to the probabilities learned during training.

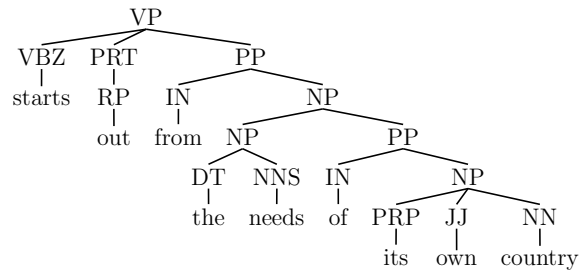


Figure 1.9: Parse Tree

In Chapters 3 and 4 of this thesis, we use constituent parsers (specifically, the Collins Model 3 parser [17] as implemented by Soricut [97]) in all parsing and string-to-tree machine translation experiments. In the Collins parsing Model 3 [17], every non-terminal label is lexicalized (augmented with the head child dominated by that node, and the part of speech tag of the head child). The generative model proceeds top-down as follows: first, the root node is generated, then the head child, then all children nodes to the left of the head child (conditioning on the root node and the head), and then all children nodes to the right of the head child (conditioning on the root node and the head). Then, each child node is processed recursively, stopping when lexicalized pre-terminals have been generated. Decoding proceeds in a bottom-up fashion using a CKY-style algorithm [16, 43, 106].

In Chapter 2, Section 2.2, we experiment with combining the output of multiple constituent parsers (specifically, the Bikel parser [4]; the Berkeley parser [88]; the Collins parser [17]; the Charniak parser [10]; and the Stanford parser [44]).

Parse Binarization

The parse trees appearing in the Penn Treebank [72], upon which most modern statistical parsers are trained, tend to have a flat structure, with each node having a

potentially very large number of children. This flatness makes it difficult for syntax-based rule extraction to extract minimal units of translational correspondence. To alleviate this problem, the parse trees are *binarized* before rule extraction according to one of two methods:

- **head-out binarization:** A deterministic method of binarizing parse trees in which each node is binarized to the left if the head is the first child, and to the right otherwise.
- **EM binarization:** A probabilistic method of binarizing parse trees in which both left and right binarizations are considered for each node, and the binarization for each node that maximizes the likelihood of the training data is selected. Word alignments are used to restrict the space of binarizations explored [102].

1.2.5 Rule Extraction

We now examine in detail the role that word alignment and parsing play in the syntax-based machine translation pipeline. Galley et al. [34] present an algorithm for extracting syntax-based translation rules from a source sentence, a target tree, and a word alignment. While familiarity with the details of the rule extraction algorithm is not a prerequisite for reading the rest of this thesis, we describe and illustrate each step of the algorithm here for completeness.

Consider the alignment shown in Figure 1.8 and parse shown in Figure 1.9. The alignment A , target parse tree T , and source sentence f together form a graph (Figure 1.10) that we refer to as an *alignment graph*. The goal of syntax-based rule extraction is to extract patterns of translational equivalence by decomposing this graph into a set of target subtrees and their corresponding source substrings. Each (target subtree, source substring) pair then constitutes a translation rule.

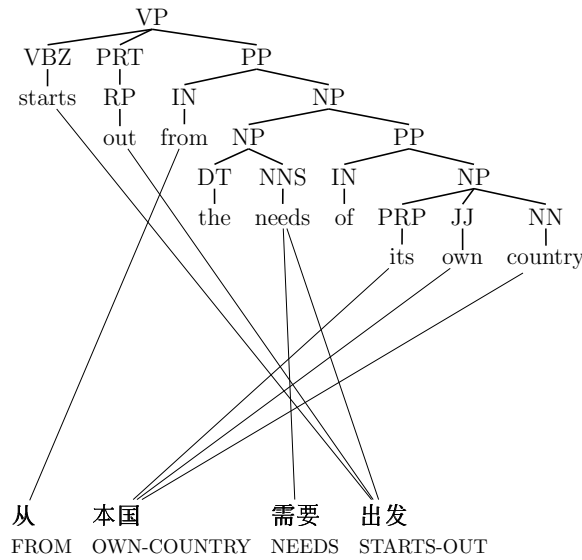


Figure 1.10: Alignment and Parse Graph

To facilitate the description of the rule extraction algorithm, Galley et al. [34] introduce the following definitions:

- **span:** The span of a node n in an alignment graph G includes the indices of all words in f that are reachable from n . In Figure 1.11, each node in the graph is annotated with its span (shown as a subscript).
- **closure:** The closure of the span of a node n in an alignment graph G is defined by the indices of the first and last words in f that are reachable from n . For example, $closure(0 - 1, 2 - 4) = (0 - 4)$.
- **complement span:** The complement span of a node n in an alignment graph G is the union of spans of all nodes n in G that are neither ancestors nor descendants of n . In Figure 1.11, each node in the graph is annotated with its complement span (shown as a superscript).
- **frontier node:** A frontier node $n \in G$ is a node for which the closure of the span of n and complement span of n do not overlap:

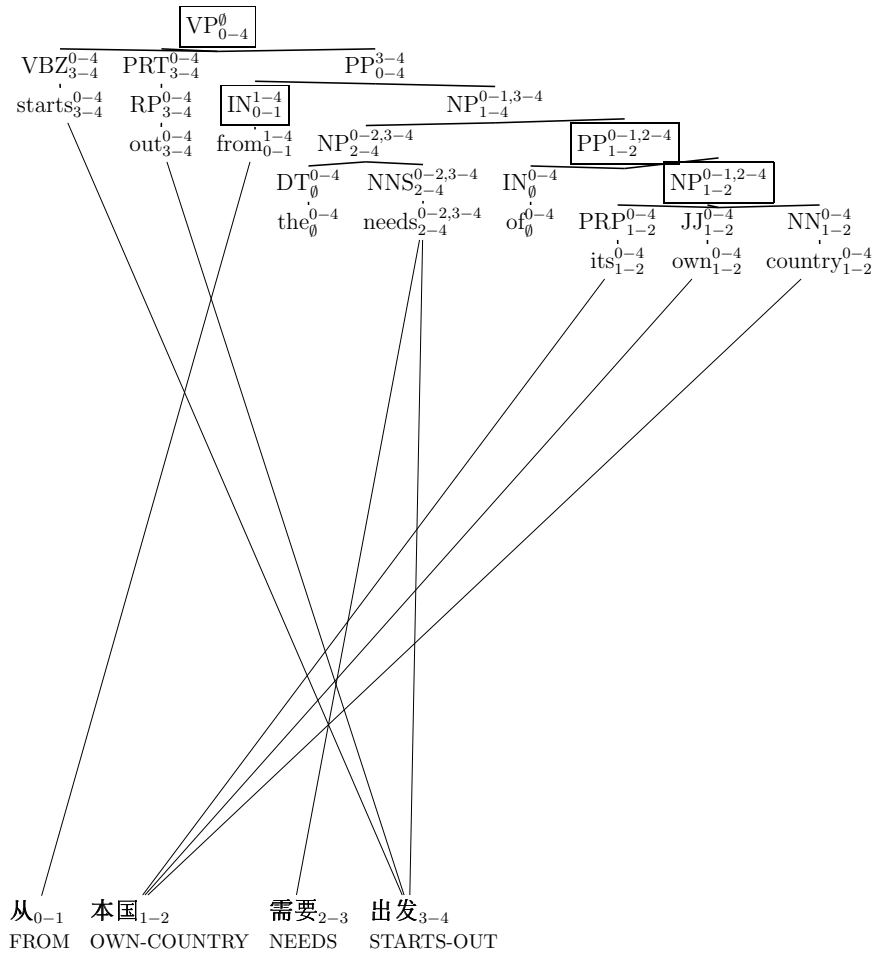


Figure 1.11: Alignment and Parse Graph with Spans (Subscript) and Complement Spans (Superscript). Each frontier node n , for which $\text{closure}(\text{span}(n)) \cap \text{complementspan}(n) = \emptyset$, is indicated by a box.

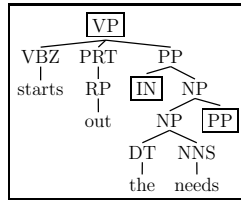
$$\text{closure}(\text{span}(n)) \cap \text{complementspan}(n) = \emptyset$$

In Figure 1.11, there are 4 frontier nodes: the root node VP, the node IN dominating “from”, the node PP dominating “of its own country”, and the node NP dominating “its own country”.

- **frontier graph fragment:** A frontier graph fragment of a graph G is a subgraph G' of G of which the root node n , and all sink nodes, are frontier nodes. In Figure 1.11, there are 4 frontier graph fragments, each one rooted at a frontier node.

The set of translation rules extracted from an alignment graph is defined by the set of frontier graph fragments, which contain target subtrees and their aligned source words. Galley et al. [34] outline a linear-time algorithm for computing this set of rules, which they refer to as the *minimal* rule set:

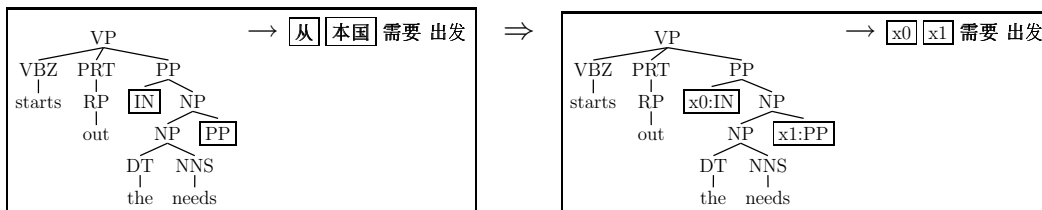
- Compute the frontier set F of the alignment graph. In Figure 1.11, the nodes in the frontier set are indicated by a box.
- For each node $n \in F$, compute the minimal frontier graph fragment rooted at that node as follows: starting from n , expand n as long as n has one or more child which is not in F . When no more expansions can be made, the frontier graph fragment rooted at node n has been identified. In Figure 1.11, the expansion of the frontier graph fragment rooted at the VP continues until the frontier nodes IN and PP are reached, producing the following frontier graph fragment:



- Each frontier graph fragment forms the left hand side of a minimal rule. The right hand side of each rule includes all foreign words within the closure of the span of the root node. Since the span of the VP in Figure 1.11 is (0-4), the entire foreign string is included in the right hand side of the rule:

从 本国 需要 出发

- For each frontier graph fragment, replace each sink node that is in F with a variable; replace the foreign words aligned to that sink node with a co-indexed variable. In the frontier graph fragment rooted at VP in Figure 1.11, the sink nodes IN and PP are in F , so each one is replaced by a variable, and the foreign words aligned to each of those nodes are replaced by the same co-indexed variable:



The extracted set of minimal translation rules (shown in Figure 1.12) corresponds exactly to the set of frontier graph fragments.

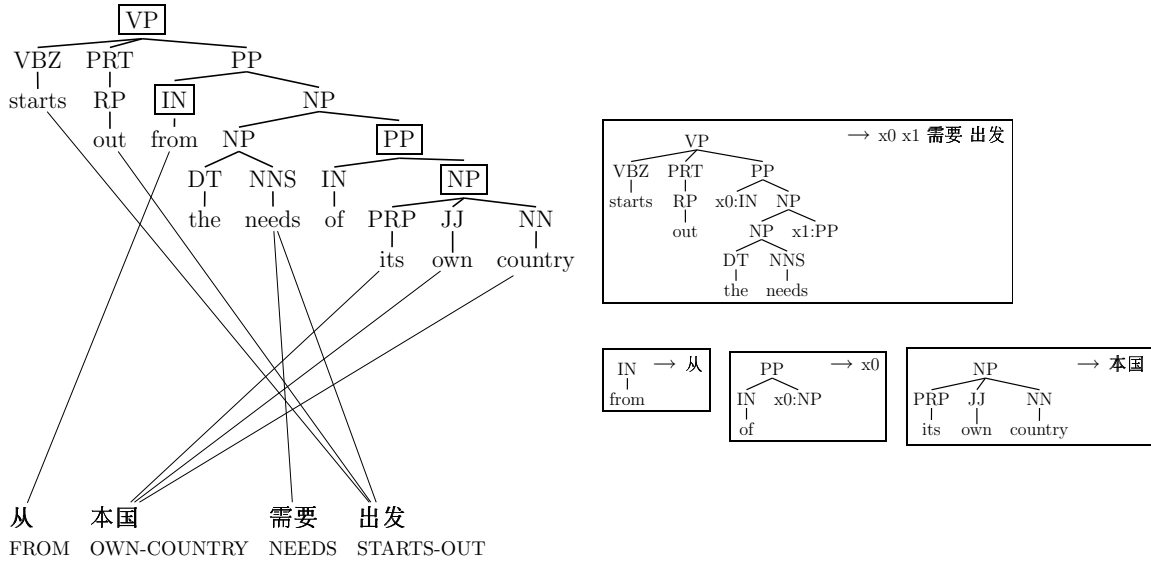


Figure 1.12: Alignment and Parse Graph with Frontier Nodes and Extracted Minimal Rules



Figure 1.13: Alignment and Parse Graph with Example of Extracted Composed Rule

In order to allow rules to include larger amounts of context, Galley et al. [33] introduce the notion of *composed* rules, which can be constructed by composing two or more minimal rules to form a single rule. For the alignment graph with minimal rules shown in Figure 1.12, an example of a composed translation rule formed by combining two minimal rules is shown in Figure 1.13.

1.2.6 Parameter Estimation

Rule probabilities are calculated based on relative frequency estimation over the training corpus, and normalized based on the root node of the left hand side (target tree) of the rule, as in a standard CFG: the normalized probability $pr(r)$ of a rule r

$$\text{is given by } \frac{\text{count}(r)}{\text{count}(\text{left-hand-side root node of } \bar{r})}.$$

1.2.7 Tuning with Minimum Error Rate Training

In addition to the translation model described above, the syntax-based machine translation pipeline also relies upon a target language model and an array of secondary features whose weights must be tuned discriminatively on a tuning set. In all experiments in this thesis, we tune all feature weights using Minimum Error Rate Training [85] in order to maximize BLEU score [87], the standard metric for translation quality.⁴

1.2.8 Decoding

The decoder used in our machine translation experiments uses the grammar of source-string-to-target-tree rules described above, and a CKY-based chart parsing algorithm to essentially parse the source language input string, producing a target language tree as output. As in monolingual CKY parsing, each item in the chart is labelled with its span in the source language, as well as the non-terminal label of the root node of its target language subtree. As in monolingual parsing, the weighted CFG must be binarized where possible to achieve optimal efficiency during decoding.

There are two important differences between monolingual CKY parsing and bilingual CKY-style translation decoding, however. The first is that, after the weights assigned to each rule by the translation model, the next most important feature used during decoding is the score assigned by an n -gram target language model to each partial translation. To facilitate the integration of the language model during decoding, each item is also labelled with the leftmost and rightmost boundary words spanned by its target language subtree. Because these additional annotations create a larger search space of possible items, cube pruning [37] is used to render the search space tractable while minimizing the number of search errors.

⁴We discuss BLEU score in more detail in Chapter 1, Section 1.2.9.

The second important difference between monolingual parsing and bilingual translation decoding is that in the bilingual case, cross-lingual reorderings may create situations in which two words whose spans are adjacent in the source language may be translated to two items which are not adjacent to each other in the target language. In this case, both items must be maintained and scoring of the partial translation by the n -gram language model must be delayed until the partial translations for the intervening target language items have been generated. Thus, the choice of binarization schemes for the weighted CFG is extremely important to the efficiency of the decoder. To minimize the number of cases where the target-language items are not contiguous, ITG-style synchronous binarization [107] is used.

1.2.9 Evaluation Metrics

We review existing evaluation metrics for alignment, parse, and translation quality, and introduce a new metric for translation rule quality.

Alignment Metrics: AER and Weighted Fully-Connected F-Measure

AER Alignment Error Rate (AER) [85] is the most widely used metric of alignment quality, but requires gold-standard alignments labelled with “sure/possible” annotations to compute. Given a hypothesized alignment A , a gold-standard set of sure alignment links S , and a gold-standard set of possible alignments P , AER is computed as follows:

$$Precision(A) = \frac{|P \cap A|}{|A|}$$

$$Recall(A) = \frac{|S \cap A|}{|S|}$$

$$AER(A) = 1 - \frac{|P \cap A| + |S \cap A|}{|S| + |A|}$$

Balanced F-Measure In many cases, there is no distinction between “sure” and “possible” alignment links in the gold-standard alignment G . In this case, the most widely used metric is balanced f-measure:

$$\begin{aligned} \textit{Precision}(A) &= \frac{|G \cap A|}{|A|} \\ \textit{Recall}(A) &= \frac{|G \cap A|}{|G|} \\ \textit{f-measure}(A) &= \frac{2}{\frac{1}{\textit{precision}(A)} + \frac{1}{\textit{recall}(A)}} \end{aligned}$$

Weighted Fully-Connected F-Measure Fraser and Marcu [32] show that improvements in AER or balanced f-measure do not necessarily correlate with improvements in BLEU score. They propose two modifications to f-measure: varying the precision/recall tradeoff, and *fully-connecting* the alignment links before computing f-measure.

To vary the precision/recall tradeoff, Fraser and Marcu [32] introduce a parameter α which controls the relative contribution of each measurement to the overall f-measure. F-measure of an alignment A for a particular value of α can be computed as follows:

$$\textit{f-measure}(A) = \frac{1}{\frac{\alpha}{\textit{precision}(A)} + \frac{1-\alpha}{\textit{recall}(A)}}$$

To more accurately reflect the impact of each alignment link upon a downstream phrase- or syntax-based MT system, Fraser and Marcu [32] introduce the concept of fully-connecting an alignment before computing its f-measure. To “fully-connect” an alignment A means to perform the following steps. First, process each word e_i in e :

- For each word e_i in e , let $f_{\textit{reachable}}(e_i)$ be the set of all words f_j in f that are reachable from e_i by traversing arbitrarily many alignment links. For example, in Figure 1.14, $f_{\textit{reachable}}(\textit{“needs”}) = \{\text{需要, 出发}\}$.

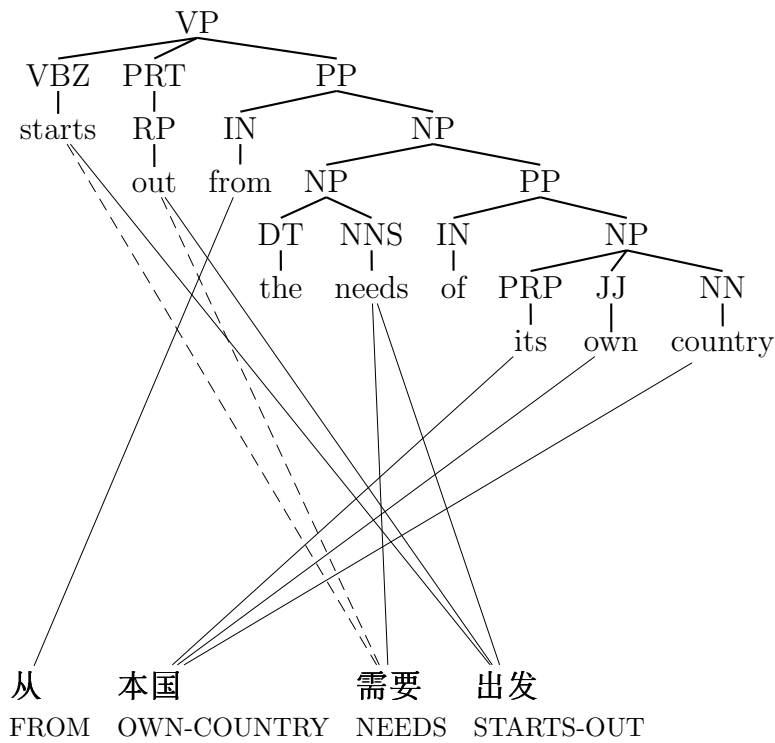
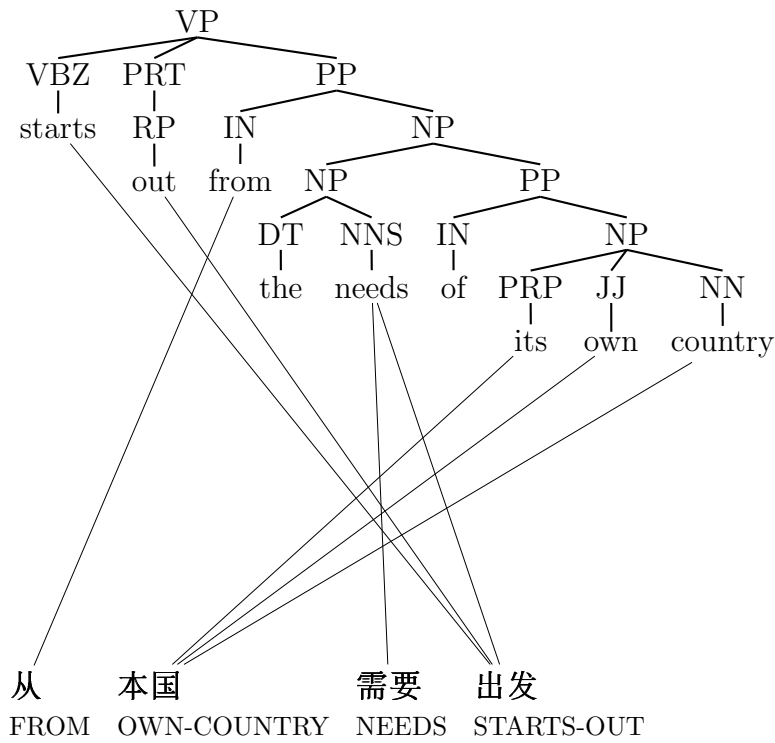


Figure 1.14: Before and After Fully-Connecting an Alignment

- Add an alignment link e_i, f'_j to A for each word $f'_j \in f_{reachable}(e_i)$. In Figure 1.14, “needs” is already aligned to every word in $f_{reachable}(\text{“needs”})$, so we do not need to add any links to A .

Then, perform the analogous procedure for each word f_j in f :

- For each word f_j in f , let $e_{reachable}(f_j)$ be the set of all words e_i in e that are reachable from f_j by traversing arbitrarily many alignment links. For example, in Figure 1.14, $e_{reachable}(\text{需要})$ contains “needs”, since 需要-“needs” $\in A$, but $e_{reachable}(\text{需要})$ also contains “starts” and “out”, since those words can be reached from 需要 by first traversing the link 需要-“needs”, then traversing the link “needs”-出发, and finally traversing the links 出发-“starts” and 出发-“out”.
- Add an alignment link e'_i, f_j to A for each word $e'_i \in e_{reachable}(f_j)$. In Figure 1.14, we must add “starts”-需要 and “out”-需要 to A , since “starts” and “out” $\in e_{reachable}(\text{需要})$ but 需要-“starts” $\notin A$ and 需要-“out” $\notin A$.

In Figure 1.14, the fully-connected version of the alignment shown includes the links 需要-starts and 需要-out. Given a hypothesized set of alignment links A and a gold-standard set of alignment links G , we define $A^+ = \text{fullyConnect}(A)$ and $G^+ = \text{fullyConnect}(G)$, and then compute:

$$Precision(A^+) = \frac{|G^+ \cap A^+|}{|A^+|}$$

$$Recall(A^+) = \frac{|G^+ \cap A^+|}{|G^+|}$$

$$f\text{-measure}(A^+) = \frac{2}{\frac{\alpha}{precision(A^+)} + \frac{1-\alpha}{recall(A^+)}}$$

For phrase-based Chinese-English and Arabic-English translation tasks, Fraser and Marcu [32] obtain the closest correlation between weighted fully-connected align-

ment f-measure and BLEU score using $\alpha=0.5$ and $\alpha=0.1$, respectively. We use weighted fully-connected alignment f-measure as the training criterion for link deletion, and to evaluate alignment quality on training and test sets.

Parse Metrics: F-Measure over Labelled Constituents

F-measure over labelled constituents as defined by the PARSEVAL guidelines is the standard metric of parse evaluation [5]. The f-measure of a parse p with respect to a reference parse g is defined as follows. Each parse is converted into a set of constituents with a syntactic label and a span indicating the minimum and maximum indices of words dominated by the subtree rooted at that constituent. Then, f-measure is computed over these labelled constituents as follows:

$$f\text{-measure}(p) = \frac{2}{\frac{1}{\text{precision}(p)} + \frac{1}{\text{recall}(p)}}$$

where $\text{precision}(p) = \frac{|p \cap g|}{|p|}$ and $\text{recall}(p) = \frac{|p \cap g|}{|g|}$

Rule Metrics: F-Measure over Extracted Translation Rules

We introduce a rule f-measure metric for translation rule quality. To compute rule f-measure in a string-to-tree syntax-based machine translation system, we require a hypothesized alignment A , target parse tree $p(e)$, and source string f . We then extract a set of minimal string-to-tree translation rules H from $(A, \text{parse}(e), f)$ as described in Section 1.2.5. We extract a gold-standard set of translation rules G from a gold-standard alignment, gold-standard target parse, and the source string; this is our reference set. We then compute precision, recall, and f-measure over the rule set H as follows:

$$f\text{-measure}(H) = \frac{2}{\frac{1}{\text{precision}(H)} + \frac{1}{\text{recall}(H)}}$$

where $precision(H) = \frac{|H \cap G|}{|H|}$ and $recall(H) = \frac{|H \cap G|}{|G|}$

Figure 1.15 illustrates an example computation of rule f-measure. The size of the hypothesized rule set, $|H|$, is 4, while the size of the gold-standard rule set, $|G|$, is 9. The number of matching rules, $|H \cap G|$, is 3: R2, R3, R4. The precision of H is $\frac{3}{4}$ and the recall of H is $\frac{3}{9}$, resulting in a rule f-measure of $\frac{6}{13}$, or approximately 0.45.

Translation Metrics: BLEU

BLEU (Bilingual Evaluation Understudy) [87] is the standard metric for translation evaluation. BLEU measures n -gram precision of the hypothesized translation against a set of multiple reference translations. Since shorter hypothesized translations contain fewer n -grams than longer ones, it is easier for shorter sentences to achieve a high n -gram precision. To combat this, BLEU also includes a brevity penalty, which penalizes hypothesized translations for being too short with respect to the reference translations. To compute BLEU, one converts the hypothesized translation into a set of n -grams, typically for values of n up to 4. The precision $precision_n$ of the hypothesis over these n -grams can then be computed against a gold-standard set of reference translations.

BLEU can then be computed from these n -gram precisions as follows:

$$BLEU_N = brevity-penalty * \sqrt[N]{\prod_{n=1}^N precision_n}$$

1.3 Thesis Outline

In this thesis, we present several methods for improving the accuracy of word alignment and syntactic parsing within the context of string-to-tree syntax-based machine translation:

Hypothesized Rules

Gold-Standard Rules

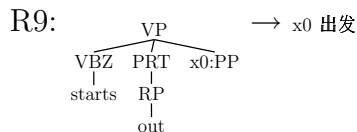
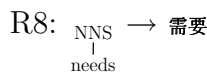
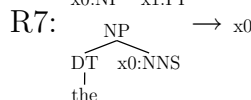
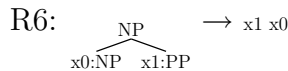
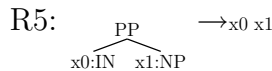
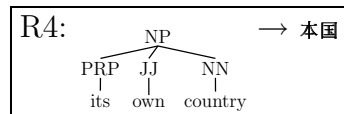
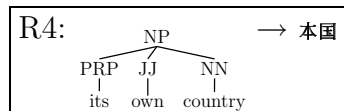
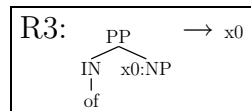
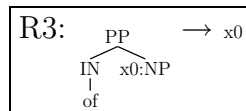
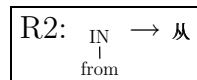
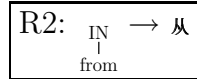
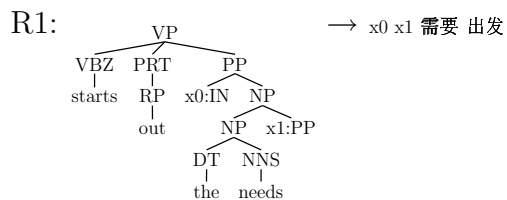


Figure 1.15: Computing Rule F-Measure

- We present a novel method for combining the output of multiple parsers by converting each parse into a set of context-free productions, then recombining those context-free productions to form a potentially new parse tree. While an existing method that recombines *constituents* achieves the highest parse accuracy, our method of recombining *context-free productions* results in better translation accuracy in a downstream string-to-tree syntax-based machine translation system [28].
- We show that automatically produced bilingual Chinese-English word alignments can be used to resolve syntactic ambiguity in English [27].
- We present a novel algorithm that uses automatically produced parses to improve the precision of word alignments by identifying and deleting alignment links that violate syntactic constraints. We demonstrate that these more highly precise alignments result in a significant gain in machine translation accuracy in syntax-based translation [29].
- We present a method for reranking (parse, alignment) *pairs* that allows parses and alignments to inform and constrain each other simultaneously.
- We investigate the relationship between rule f-measure and translation quality.

Chapter 2

Improving Parsing Accuracy

Errors in English parse trees impact the quality of syntax-based MT systems trained using those parses. Frequent sources of error for English parsers include PP-attachment ambiguity, NP-bracketing ambiguity, and coordination ambiguity. One way to improve the accuracy of a parser is to combine the output of multiple parsers; insofar as the errors made by each parser are independently distributed, the consensus combination tends to achieve a higher accuracy than the best individual parser. Another way to improve the accuracy of a parser is to incorporate an additional source of information (other than the features on which the parser was trained).

In this chapter, we present two ways to improve parse quality in English. First, in Section 2.2, we combine the output of multiple statistical parsers, achieving the highest reported parse accuracy on a standard testing section of the Wall Street Journal. We examine problems with existing methods for parse combination, proposing a) an efficient, linear-time algorithm for selecting the parse with highest expected f-measure from an n -best list, and b) a novel method for combining multiple parses by converting each parse in an n -best list into a set of context-free productions, then recombining those productions to form a combined parse. We examine the impact of each method of parse combination upon both parse accuracy and syntax-based machine translation. Second, in Section 2.3, we use automatically generated bilin-

gual Chinese-English word alignments to resolve syntactic ambiguity in English. We show that, using these word alignments, we can resolve PP-attachment ambiguity (a frequent source of parse error in English) with an accuracy that is better than that of the baseline Collins parser.

2.1 Background

Syntactic parsing is the task of assigning a syntactic tree structure t to a sentence s . In statistical parsing, the parameters of the parsing model $P(t|s)$ are learned automatically from a training corpus consisting of sentences and their associated syntactic trees. The parser can then determine the most likely tree structure $\hat{t} = \arg \max_t P(t|s)$ for an unseen sentence s according to the probabilities learned during training.

A common source of errors for parsers is *syntactic ambiguity*, or the existence of more than one possible syntactic structure for a particular sentence. Frequently occurring types of syntactic ambiguity in English include prepositional phrase attachment, noun phrase bracketing, and coordination ambiguity.

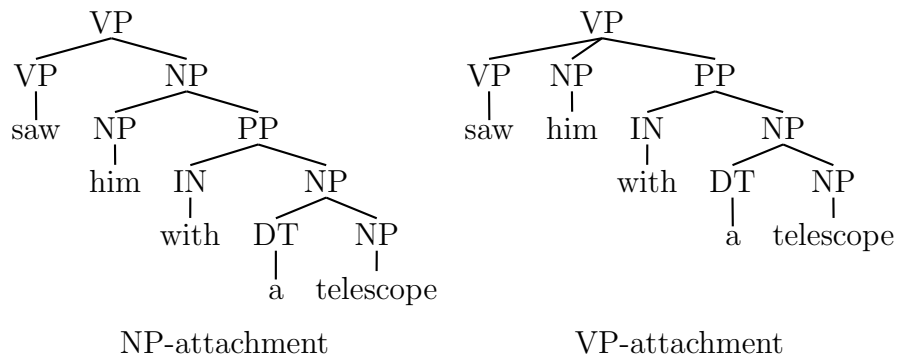


Figure 2.1: PP-attachment ambiguity in English

Figure 2.1 illustrates an example of PP-attachment ambiguity in English. Given the phrase “saw him with a telescope”, should the PP “with a telescope” modify the noun

phrase “him” (indicating that the man in question was in possession of a telescope at the time of the sighting), or the verb phrase “saw” (indicating that the man was observed through the lens of a telescope)? The leftmost syntactic structure represents attachment of the PP to the NP; the rightmost, attachment to the VP.

Errors in English parse trees negatively impact the quality of syntax-based MT systems trained using those parses. Quirk et al. [89] show that, in dependency treelet translation, BLEU scores on English-German and English-Japanese experiments degrade as the amount of training data used to train the source dependency parser decreases.

In the string-to-tree syntax-based MT system used in our experiments [34, 33], the quality of translation rules extracted from each English parse tree and bilingual word alignments deteriorates as the quality of the parses and word alignments decreases. For example, Figure 2.2 illustrates the impact of parse errors upon the translation rules extracted from an actual Chinese-English sentence pair. By contrast, Figure 2.3 illustrates the rules extracted from a correct English parse. The rules extracted from the correct English parse can be applied in more general contexts than the rules extracted from the incorrect English parse.¹

In this chapter, we explore two ways to improve the accuracy of constituent parsers, with the ultimate goal of improving the accuracy of a downstream syntax-based machine translation system. In Section 2.2, we combine the output of multiple constituent parsers. In Section 2.3, we incorporate an additional source of information into monolingual English parsing: the information contained in automatic bilingual Chinese-English word alignments. The primary contributions of this work are as follows:

¹In some instances, it may be advantageous to incorporate larger amounts of context into a translation rule, but the decision to include more context where necessary should be a principled one, not a default action forced upon the rule extraction algorithm by the training data.

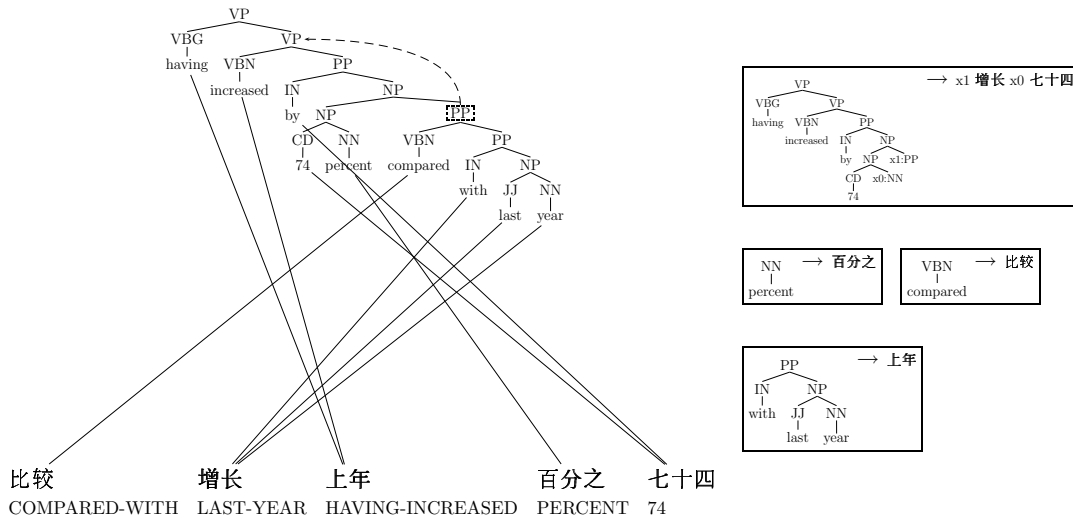


Figure 2.2: Parse with PP-Attachment Error and Extracted Translation Rules

- We present an efficient, linear-time algorithm for parse selection that maximizes expected f-measure
- We propose a novel method for parse hybridization that recombines context-free productions instead of constituents
- We show that bilingual word alignments can be used to resolve English syntactic ambiguity

2.2 Combining Constituent Parsers of English

Combining multiple classifiers has been shown to improve accuracy on a variety of tasks in NLP (such as POS tagging [99, 1], PP-attachment ambiguity [1], word-sense disambiguation [22, 21] and machine translation [93]). To the extent that errors are independently distributed, the combined classifier can achieve a higher accuracy than the best individual classifier. The output of multiple individual parsers can be combined in two ways:

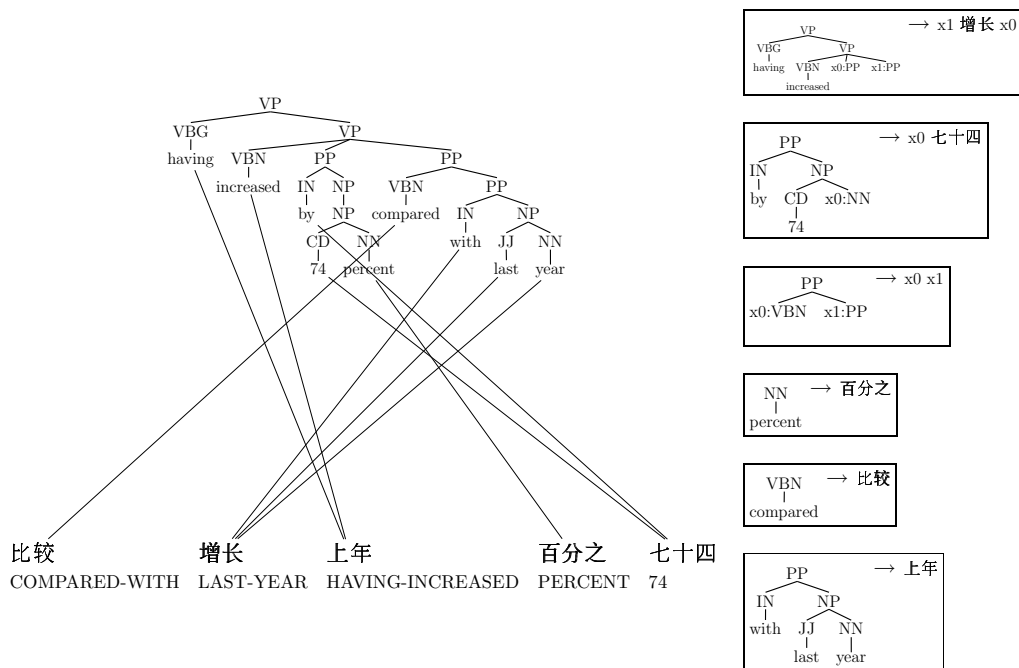


Figure 2.3: Parse with Correct PP-Attachment and Extracted Translation Rules

- **Parse Selection:** Selecting the best parse from the output of the individual parsers.
- **Parse Hybridization:** Constructing the best parse by recombining sub-sentential components from the output of the individual parsers.

Combining the 1-best output of multiple parsers via parse selection or parse hybridization has been shown to improve f-measure over the best individual parser [35, 94]. Parse selection is a more conservative way to combine the output of individual parsers, in that it can do no better than selecting the best individual parse but no worse than selecting the worst individual parse. Parse hybridization can potentially create new parses that do not appear in the list of individual parses being combined, and can create parses that are much better than the best individual parse or much worse than the worst individual parse. Because parse hybridization explores a larger space of possible combined parses than parse selection, parse hybridization achieves a higher f-measure in our oracle experiments (see Table 2.1).

System	wsj-dev	wsj-test	ce-dev	ce-test
Parse Selection: Minimum Bayes Risk				
Baseline: Charniak 1-best	90.6	91.5	84.9	84.2
Oracle n=1:	94.1	95.0	90.1	89.4
Parse Hybridization: Constituent Recombination				
Baseline: Charniak 1-best	90.6	91.5	84.9	84.2
Oracle n=1:	98.0	98.3	96.3	96.2
Parse Hybridization: Context-Free Production Recombination				
Baseline: Charniak 1-best	90.6	91.5	84.9	84.2
Oracle n=1:	95.4	95.9	92.1	92.0

Table 2.1: Oracle Results for Parse Selection and Parse Hybridization

2.2.1 Parse Selection

Parse selection is the problem of selecting the best parse from an n -best list of candidates. In order to compute a consensus selection over the list of candidate parses, it is necessary to devise an objective function that measures the extent to which the parsers “agree” with each candidate parse.

Related Work

Henderson and Brill [35] perform parse selection by maximizing the number of matched constituents between the selected parse and the set of parses being combined. Given an n -best list of parses to combine, they first convert each parse into a set of constituents labelled with their spans. For each parse p_i , they then compute the number of matched constituents $|p_i \cap p_j|$ between p_i and all parses p_j in the n -best list. The score of each parse p_i is then given by $\sum_{p_j} |p_i \cap p_j|$. They return the parse with maximum score.

The parse selection approach of Henderson and Brill [35] has two problems. First, they maximize a metric of parse similarity (the number of matched constituents) that is different from the standard metric used to evaluate parse quality (PARSEVAL, or f-measure over labelled constituents) [5]. The f-measure of parse p_i with respect

to a reference parse p_j is defined as follows. Each parse is converted into a set of constituents with a syntactic label and a span indicating the minimum and maximum indices of words dominated by the subtree rooted at that constituent. Then, f-measure is computed over these labelled constituents as follows:

$$f\text{-measure}(p_i) = \frac{2}{\frac{1}{\text{precision}(p_i)} + \frac{1}{\text{recall}(p_i)}}$$

where $\text{precision}(p_i) = \frac{|p_i \cap p_j|}{|p_i|}$ and $\text{recall}(p_i) = \frac{|p_i \cap p_j|}{|p_j|}$

Since f-measure, and not the number of matched constituents, is the metric by which parse quality is evaluated, it would be preferable to optimize f-measure directly. F-measure is equivalent to the number of matched constituents divided by the average number of constituents in p_i and p_j :

$$f\text{-measure}(p_i) = \frac{|p_i \cap p_j|}{\frac{|p_i| + |p_j|}{2}}$$

We present an efficient, linear-time algorithm for parse selection that optimizes this criterion directly.

The second problem with the parse selection approach of Henderson and Brill [35] is that computing $\sum_{p_j} |p_i \cap p_j|$ for each parse p_i requires $O(n^2)$ time, where n is the number of parses in the n -best list, since each parse p_i must be compared, pairwise, against each parse p_j . To combat this inefficiency, we present a method for computing the expected f-measure of each parse p_i in time that is *linear*, rather than *quadratic*, in the number of parses.

Our Work

We improve upon existing methods for parse selection in two ways. While the parse selection method of Henderson and Brill [35] requires $O(n^2)$ time to select the parse with maximum expected number of *matched constituents*; here, we present

an efficient $O(n)$ time algorithm for selecting the parse with maximum expected *f-measure* within the Minimum Bayes Risk (MBR) framework. We also generalize this method to combining k -best outputs from multiple parsers, rather than only 1-best outputs from multiple parsers.

An Efficient Linear-Time Algorithm for Selecting the Parse with Maximum Expected F-Measure

If we knew the true f-measure of each parse relative to the gold-standard parse, we could simply select the parse with the highest f-measure from our n -best list. In general, of course, we do not know the true f-measure of each parse in our n -best list when performing parse selection. In the absence of the *true* f-measure, we can instead use a Minimum Bayes Risk approach to minimize the expected loss in f-measure. In the Minimum Bayes Risk framework, although the true gold-standard parse is unknown, we assume that the individual parses in the n -best list form a reasonable distribution over possible gold-standard parses.

We compute the expected f-measure of each parse p_i as follows:

$$expected\ f(p_i) = \sum_{p_j} f(p_i, p_j) \cdot pr(p_j)$$

where $f(p_i, p_j)$ is the f-measure of parse p_i with respect to parse p_j and $pr(p_j)$ is the prior probability of parse p_j .

We estimate the probability $pr(p_j)$ of each parse p_j as follows: $pr(p_j) = pr(parser_k) \cdot pr(p_j|parser_k)$, where $parser_k$ is the parser generating p_j . We set $pr(parser_k)$ according to the proportion of sentences in the development set for which the 1-best output of $parser_k$ achieves the highest f-score of any individual parser, breaking ties randomly; this is a measure of the accuracy of each parser relative to the others.

When $n = 1$, $pr(p_j|parser_k) = 1$ for all p_j ; when $n > 1$ we must estimate $pr(p_j|parser_k)$, the distribution over parses in the n -best list output by any given parser. We estimate this distribution using the model score, or log probability, given by $parser_k$ to each entry p_j in its n -best list:

$$pr(p_j|parser_k) = \frac{e^{\alpha \times score_{parser_k}(p_j)}}{\sum_{j'=1}^n e^{\alpha \times score_{parser_k}(p_{j'})}}$$

We tune α on a development set to maximize f-score. A low value of α creates a uniform distribution, while a high value concentrates probability mass on the 1-best entry in the n -best list. In practice, tuning α produces a higher f-score than setting α to the value that exactly reproduces the individual parser’s probability distribution.

After selecting a value of α for each parser that maximizes f-measure on the development set, we can then apply our method to an unseen list of n -best parses at test time to select the parse p_i with highest expected f-measure relative to the n -best list.

Computing *exact* expected f-measure requires $O(n^2)$ operations per sentence, where n is the number of parses being combined. If f-measure were a linear function of the constituents appearing in each parse, we could compute expected f-measure in $O(n)$ time by indexing each parse by the constituents it contains, and then iterating over constituents rather than iterating over parses to compute f-measure for each parse [24]. Unfortunately, f-measure is not a linear function of the constituents of a parse, but precision and recall are. Thus, we can compute expected precision and expected recall for all parses in the n -best list in $O(n)$ time. Algorithm 2 illustrates how to compute expected precision in $O(n)$ time, and 3 illustrates how to compute expected recall in $O(n)$ time.

Using these exact expected precision and expected recall scores, which can each be computed in linear time, we can then compute an *approximate* expected f-measure as follows:

$$\textit{approximate expected } f(p_i) = \frac{2 \times \textit{precision}(p_i) \times \textit{recall}(p_i)}{\left(\frac{1}{\textit{precision}(p_i)} + \frac{1}{\textit{recall}(p_i)}\right)}$$

Essentially, our approximation computes the harmonic mean of expected precision and expected recall, while the exact method computes the expectation of the harmonic mean of precision and recall. This approximate expected f-score can be computed for all parses in $O(n)$ time, and in practice, this approximation is very close to the true expected f-score.

Algorithm 2: Linear-Time Expected Precision

```
input : A list  $P$  of parses  $p$  to combine, and a probability  $pr(p)$  associated  
with each parse  $p$   
output: Expected precision of each parse  $p$  in  $P$   
for  $p \in P$  do  
  |  $expected-precision(p) = 0$ ;  
  | for  $c \in p$  do  
  |   |  $occurrences(c) = \emptyset$ ;  
  |   | end  
end  
for  $p \in P$  do  
  | for  $c \in p$  do  
  |   |  $occurrences(c) = occurrences(c) \cup p$ ;  
  |   | end  
end  
for  $c \in C$  do  
  |  $sum(c) \leftarrow 0$ ;  
  | for  $p \in occurrences(c)$  do  
  |   |  $sum(c) = sum(c) + pr(p)$ ;  
  |   | end  
  | for  $p \in occurrences(c)$  do  
  |   |  $expected-precision(p) = expected-precision(p) + sum(c)$ ;  
  |   | end  
end  
for  $p \in P$  do  
  |  $expected-precision(p) = \frac{expected-precision(p)}{|p|}$ ;  
  | return  $expected-precision(p)$ ;  
end
```

Algorithm 3: Linear-Time Expected Recall

```
input : A list  $P$  of parses  $p$  to combine, and a probability  $pr(p)$  associated  
with each parse  $p$   
output: Expected recall of each parse  $p$  in  $P$   
for  $p \in P$  do  
  |  $expected-recall(p) = 0$ ;  
  | for  $c \in p$  do  
  | |  $occurrences(c) = \emptyset$ ;  
  | end  
end  
for  $p \in P$  do  
  | for  $c \in p$  do  
  | |  $occurrences(c) = occurrences(c) \cup p$ ;  
  | end  
end  
for  $c \in C$  do  
  |  $sum(c) \leftarrow 0$ ;  
  | for  $p \in occurrences(c)$  do  
  | |  $sum(c) = sum(c) + \frac{pr(p)}{|p|}$ ;  
  | end  
  | for  $p \in occurrences(c)$  do  
  | |  $expected-recall(p) = expected-recall(p) + sum(c)$ ;  
  | end  
end  
for  $p \in P$  do  
  | return  $expected-recall(p)$ ;  
end
```

Extending Parse Selection from 1-best to k -best Outputs from n Parsers

We extend this method from the 1-best output of multiple parsers to the n -best output of multiple parsers as follows: we compute the weight of each constituent by summing $pr(parser_k) \cdot pr(p_j|parser_k)$ over all parses p_j generated by $parser_k$ in which the constituent appears.

2.2.2 Parse Hybridization

Parse hybridization is the problem of recombining sub-sentential components from an n -best list of candidate parses. While parse *selection* cannot create any new parses that do not already appear among the individual parses, parse *hybridization* can recombine parts of the individual parse trees to produce an entirely new parse.

Related Work

In addition to their method for parse *selection* described above, Henderson and Brill [35] also propose a method for parse *hybridization* called *constituent voting* in which they convert each parse into constituents with syntactic labels and spans, and weight each constituent by summing $pr(parser_k)$ over all parsers k in whose output the constituent appears. They include all constituents with weight above a threshold $t = \frac{n}{2}$, where n is the number of input parses, in the combined parse.

While the constituent voting scheme of Henderson and Brill [35] performs better than their parse selection scheme in terms of f-measure of the resulting combined parse, there are two problems with their approach.

First, because constituent voting is not constrained by a context-free grammar, it is possible for the combined parse to contain context-free productions that have not been seen in any of the individual parses. Since the combined parse is evaluated on the basis of f-measure, which looks only at constituents but not at context-free

productions, this property of constituent voting does not negatively impact parse f-measure, but it can pose a problem for downstream applications that rely upon the well-formedness of the combined parses. To illustrate, Figures 2.4 and 2.5 contrast the output of the Charniak parser with the output of constituent recombination on a sentence from WSJ section 24. Henderson and Brill [35] acknowledge this shortcoming:

In general, [constituent voting] does not ensure that all the productions in the combined parse are found in the grammars of the member parsers. There is a guarantee of no crossing brackets but there is no guarantee that a constituent in the tree has the same children as it had in any of the $[n]$ original parses. One can trivially create situations in which strictly binary-branching trees are combined to create a tree with only the root node and the terminal nodes, a completely flat structure. This drastic tree manipulation is not appropriate for situations in which we want to assign particular structures to sentences. For example, we may have semantic information (e.g. database query operations) associated with the productions in a grammar. If the parse contains productions from outside our grammar the machine has no direct method for handling them (e.g. the resulting database query may be syntactically malformed).

Second, their method produces a parse tree with maximum *precision* over labelled constituents; it does not allow the user to tune the tradeoff between precision and recall of the combined parse. Since precision and recall both affect the f-measure of the combined parse, it would be preferable to be able to tune this precision-recall tradeoff.

To allow the user to tune the tradeoff between precision and recall in the combined parse, Sagae and Lavie [94] propose an extension of the constituent voting scheme called *reparsing*. In the reparsing framework, rather than simply selecting all constituents with a weight greater than threshold $t = \frac{\eta}{2}$, they tune t on a development set to maximize f-measure. (A high threshold results in high precision, while a low threshold results in high recall.) They populate a chart with all constituents whose weight meets the threshold, and use a CKY-style parsing algorithm to find the

“heaviest” tree, where the weight of a tree is the sum of its constituents’ weights. Parsing is not constrained by a grammar; any context-free production is permitted. Thus, this method still suffers from the same problem as [35] in that the combined parses may contain context-free productions not seen in the individual parsers’ outputs. While this failure to preserve the structure of individual parses does not affect parse f-measure, it may hinder downstream syntax-based applications.

To remedy the problem with constituent recombination wherein the combined parse may contain context-free productions that have not been seen in any of the individual parses, we propose a novel method for parse hybridization called *context-free production recombination*. In this framework, we convert each parse into a set of labelled context-free productions (with each constituent annotated with its syntactic label and span) instead of constituents. We then recombine context-free productions to produce a combined parse tree. This approach guarantees that no new context-free productions will be created in the combined parse, and thereby better preserves the syntactic structures of the individual parses. In the example sentence from WSJ Section 24 referenced above, our context-free production recombination method produces the output shown in Figure 2.6.

Our Work

While constituent recombination [35, 94] gives a significant improvement in f-score, it has the potential to create context-free productions that have not been seen in the output of any of the individual parses. To combat this, we recombine *context-free productions* instead of *constituents*, producing trees containing only context-free productions that have been seen in the individual parsers’ output. We extend these parser combination methods from 1-best outputs to n -best outputs.

Recombining Context-Free Productions Instead of Constituents

To ensure that all context-free productions in the combined parses have been seen in the individual parsers' outputs, we recombine context-free productions rather than constituents. We convert each parse into context-free productions, labelling each constituent in the production with its span and syntactic category and weighting each production by summing $pr(parser_k) \cdot pr(p_j|parser_k)$ over all parses p_j generated by $parser_k$ in which the production appears. We then create a parse forest containing these context-free productions, and use the Tiburon tree transducer toolkit [73] to return the heaviest tree (where the weight of a tree is the sum of its context-free productions' weights). We optimize f-score by varying the tradeoff between precision and recall using a derivation length penalty, which we tune on a development set. By subtracting higher(lower) values of this length penalty from the weight of each production, we can encourage the combination method to favor trees with shorter(longer) derivations and therefore higher precision(recall) at the constituent level.

Extending Parse Hybridization from 1-best to k -best Outputs from n Parsers

We extend this method from the 1-best output of multiple parsers to the n -best output of multiple parsers as follows: we compute the weight of each constituent by summing $pr(parser_k) \cdot pr(p_j|parser_k)$ over all parses p_j generated by $parser_k$ in which the constituent appears.

2.2.3 Methods

Parsing Experiments

Table 2.2 illustrates the data sets used in our parse combination experiments. All parsers were trained on the standard WSJ training sections (02-21). We eval-

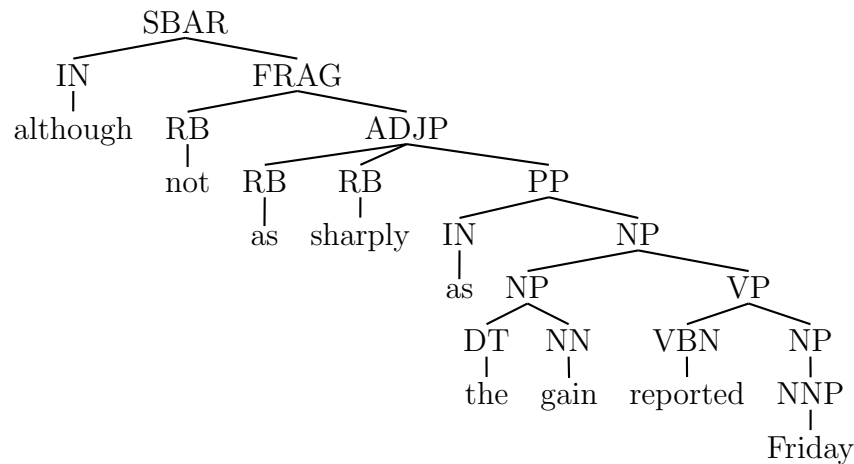


Figure 2.4: Output of Charniak Parser

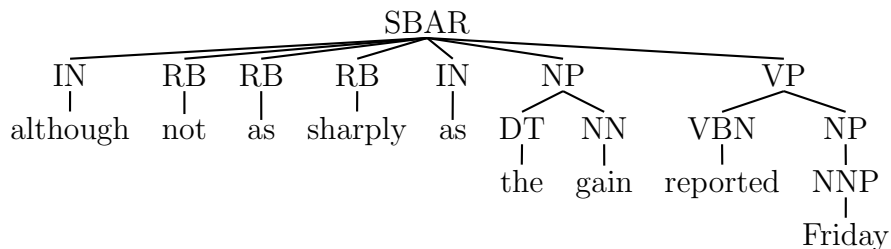


Figure 2.5: Output of Constituent Recombination

uate parse accuracy using two different corpora: the WSJ (sections 24 and 23 are the development and test sets, respectively), and an excerpt of English text from the LDC2007T02 Chinese-English parallel corpus [3] (the development and test sets contain 400 sentences each).

Table 2.3 illustrates the 5 parsers used in our combination experiments and the f-scores of their 1-best output on our data sets. We use the n -best output of the Berkeley, Charniak, and Soricut (a re-implementation of the Collins Model 3) parsers, and the 1-best output of the Bikel and Stanford parsers.

Results of each method of parse combination are shown in Tables 2.4, 2.5, and 2.6. On both test sets, constituent recombination (Table 2.5) achieves the largest gain in parse f-measure (+1.0 points on WSJ test and +2.3 points on Chinese-English test), followed by context-free production combination, then parse selection, though the differences in f-measure among the combination methods are not statistically

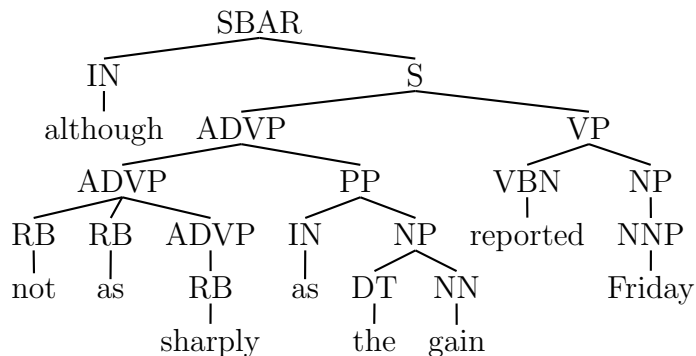


Figure 2.6: Output of Context-Free Production Recombination

Genre	Train	Tune	Test
WSJ train +	WSJ 02-21[72]	WSJ 24[72]	WSJ 23[72]
WSJ tune/test	(39,832 sent.)	(1346 sent.)	(2416 sent.)
WSJ train +	WSJ 02-21[72]	LDC2007T02[3] excerpt	LDC2007T02[3] excerpt
newswire tune/test	(39,832 sent.)	(400 sent.)	(400 sent.)

Table 2.2: Data Sets Used in Parser Combination: Parsing Experiments

significant. Increasing the n -best list size from 1 to 10 improves parse selection and context-free production recombination, though further increasing n from 10 to 25 or 50 does not, in general, help. These diminishing gains in f-score as n increases reflect the diminishing gains in f-score of the oracle parse produced by each individual parser as n increases (Table 2.1). The Chinese-English test set f-measure gets a bigger boost from combination than the WSJ test set f-score, perhaps because the best individual parser’s baseline f-score is lower on the out-of-domain data.

Parser	wsj		ce	
	dev	test	dev	test
Berkeley[88]	88.6	89.3	82.9	83.5
Bikel–Collins Model 2[4]	87.0	88.2	81.2	80.6
Charniak[11]	90.6	91.4	84.7	84.1
Soricut–Collins Model 2[97]	87.3	88.4	82.3	82.1
Stanford[44]	85.4	86.4	81.3	80.1

Table 2.3: F-Measure of 1-best Output of Individual Parsers

Parse Selection: Minimum Bayes Risk												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	91.7	90.5	91.1	92.5	91.8	92.0	87.1	84.6	85.8	86.7	83.7	85.2
n=10	92.1	90.8	91.5	92.4	91.7	92.0	87.9	85.3	86.6	87.7	84.4	86.0
n=25	92.1	90.9	91.5	92.4	91.7	92.0	88.0	85.4	86.7	87.4	84.2	85.7
n=50	92.1	91.0	91.5	92.4	91.7	92.1	88.0	85.3	86.6	87.6	84.3	85.9

Table 2.4: Precision, Recall, and F-score Results from Parse Selection

Parse Hybridization: Constituent Recombination												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	92.5	90.3	91.4	93.0	91.6	92.3	89.2	84.6	86.8	89.1	83.6	86.2
n=10	92.6	90.5	91.5	93.1	91.7	92.4	89.9	84.4	87.1	89.9	83.2	86.4
n=25	92.6	90.5	91.5	93.2	91.7	92.4	89.9	84.4	87.0	89.7	83.4	86.4
n=50	92.6	90.5	91.5	93.1	91.7	92.4	89.9	84.4	87.1	89.7	83.2	86.3

Table 2.5: Precision, Recall, and F-score Results from Constituent Recombination

Parse Hybridization: Context-Free Production Recombination												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	91.7	91.0	91.4	92.1	91.9	92.0	86.9	85.4	86.2	86.2	84.3	85.2
n=10	92.1	90.9	91.5	92.5	91.8	92.2	87.8	85.1	86.4	86.2	84.3	86.1
n=25	92.2	91.0	91.6	92.5	91.8	92.2	87.8	85.1	86.4	87.6	84.6	86.1
n=50	92.1	90.8	91.4	92.4	91.7	92.1	87.6	84.9	86.2	87.7	84.6	86.1

Table 2.6: Precision, Recall, and F-score Results from Context-Free Production Recombination

Syntax-Based Machine Translation Experiments

Train	Tune	Test 1	Test 2
ara-eng train (small)	ara-eng tune newswire	ara-eng test 1 newswire	ara-eng test 2 newswire
(170,863 sent.)	(1178 sent.)	(1298 sent.)	(765 sent.)

Table 2.7: Data Sets Used in Parser Combination: Syntax-Based MT Experiments. Contents of Each Data Set Listed in Appendix A, Tables A.1 and A.2.

To examine the impact of parse quality upon syntax-based machine translation, we first retrain the Berkeley, Charniak, and Soricut parsers on the concatenation of the Wall Street Journal Portion of the Penn Treebank [72] and 25,000 manually parsed sentences taken from the English sides of the following parallel newswire corpora: the Chinese-English corpus LDC2007T02[3], the Arabic-English corpora LDC2005E85[51], LDC2006E36[56], LDC2006E82[58], and LDC2006E95[60].

Using the retrained Berkeley, Charniak, and Soricut parsers, we parse the English side of an Arabic-English parallel corpus (ara-eng train (small))² We combine the n -best outputs of these 3 individual parsers using each of the 3 combination methods described above: MBR parse selection, constituent recombination, and context-free production recombination. We then train a syntax-based Arabic-English string-to-tree translation system using the output from each of the individual parsers and each of the combination methods in turn, tune the MT system parameters using data set ara-eng tune³, and compare the resulting BLEU scores on two held-out test sets: Test 1 (ara-eng test 1 newswire)⁴ and Test 2 (ara-eng test 2 newswire)⁵. Data sets and sizes are listed in Table 2.7.

Tables 2.8, 2.9, and 2.10 illustrate the results of performing string-to-tree syntax-based MT experiments using various configurations of the MT system. In Table

²See Appendix A for contents of data set.

³See Appendix A for contents of data set.

⁴See Appendix A for contents of data set.

⁵See Appendix A for contents of data set.

Parser	Rules	Binarization	BLEU score		
			Dev	Test 1	Test 2
Individual Parsers					
Berkeley	minimal	–	51.3	49.1	45.7
Charniak	minimal	–	50.8	48.8	46.2
Soricut–Collins model 3	minimal	–	50.3	47.8	45.1
Combined Parses					
MBR	minimal	–	51.3	49.0	45.9
Constituent Recombination	minimal	–	51.1	48.7	45.9
CFG Production Recombination	minimal	–	51.7	49.0	46.3

Table 2.8: BLEU Scores Using Parser Combination in Arabic-English Syntax-Based MT with Minimal Rules and No Binarization

Parser	Rules	Binarization	BLEU score		
			Dev	Test 1	Test 2
Individual Parsers					
Berkeley	composed	head-out	52.7	48.7	45.8
Charniak	composed	head-out	52.2	49.3	46.9
Soricut–Collins model 3	composed	head-out	51.7	48.4	45.3
Combined Parses					
MBR	composed	head-out	51.7	48.6	45.6
Constituent Recombination	composed	head-out	52.0	49.0	47.2
CFG Production Recombination	composed	head-out	53.0	49.0	47.9

Table 2.9: BLEU Scores Using Parser Combination in Arabic-English Syntax-Based MT with Composed Rules and Head-Out Binarization

Parser	Rules	Binarization	BLEU score		
			Dev	Test 1	Test 2
Individual Parsers					
Berkeley	composed	em	51.8	48.6	47.3
Charniak	composed	em	52.8	49.2	47.3
Soricut–Collins model 3	composed	em	51.1	48.1	45.1
Combined Parses					
MBR	composed	em	52.0	48.8	45.8
Constituent Recombination	composed	em	52.0	48.7	46.9
CFG Production Recombination	composed	em	52.6	49.7	47.4

Table 2.10: BLEU Scores Using Parser Combination in Arabic-English Syntax-Based MT with Composed Rules and EM Binarization

2.8, we present the results of training a syntax-based string-to-tree MT system using *minimal* rules only and no *composed* rules⁶, and without binarizing the parse trees after combination.⁷ In Table 2.9, we present the results of an MT experiment using *composed* rules and *head-out binarization*. In Table 2.10, we use *composed* rules and *em-binarization*.⁸

In all configurations tested, our method for parse hybridization by recombining context-free productions achieves the highest BLEU score of all three parse combination methods. Our method achieves a BLEU score that is equivalent to or better than that achieved by the best baseline parser (Berkeley or Charniak) in all configurations tested, though the gains in BLEU score due to parser combination are small (+0.1 points relative to the Berkeley parser using minimal rules and no binarization, +1.0 points relative to the Charniak parser using composed rules and head-out binarization, and +0.1 points relative to the Berkeley parser using composed rules

⁶For more details about minimal and composed rules, please refer to Chapter 1, Section 1.2.5.

⁷For more details about parse binarization for syntax-based machine translation, please refer to Chapter 1, Section 1.2.4.

⁸This configuration represents the one used to produced state-of-the-art MT results, though it is possible that in order to achieve the best possible results with em-binarization, it is necessary to use a larger training corpus than the one used in our experiments.

and em-binarization. From these results, we conclude that, while the constituent recombination method of Henderson and Brill [35] and Sagae and Lavie [94] achieves the highest parse f-measure, our method for parse combination by context-free production recombination outperforms constituent recombination when used to train a downstream syntax-based machine translation system.

While the results of our parse combination experiments show that both parse selection and parse hybridization improve the f-measure of the combined parse relative to the best individual parse, the results of our syntax-based machine translation experiments show an inconclusive relationship between parse f-measure and BLEU score. The best performing method for parse combination in terms of parse f-measure was constituent recombination, while the best performing method in terms of BLEU score was context-free production recombination. None of the methods for parse combination, however, led to a consistent increase in BLEU score across all the system configurations we explored. Thus, we conclude that, while previous researchers have found that degrading parse quality results in a degradation in BLEU score, the converse is not necessarily true—improving parse quality over a state-of-the-art constituent parser does not necessarily improve BLEU score, at least for the string-to-tree syntax-based system used in our experiments.

2.2.4 Summary of Contributions

In our work on parsing combination, we have presented:

- An efficient, linear-time algorithm for selecting the parse with maximum expected f-measure from an n -best list of candidates, using a Minimum Bayes Risk approach
- A novel method for parse hybridization that recombines context-free productions, instead of constituents, thereby better preserving the structure of the

individual parsers and producing better-quality translations in a downstream syntax-based MT system

In the next section, we address an alternative way to improve the accuracy of a state-of-the-art constituent parser: incorporating bilingual word alignments as an additional source of knowledge that can be used by the parser to resolve syntactic ambiguity.

2.3 Using Bilingual Word Alignments to Resolve English Syntactic Ambiguity

Frequent sources of error for English parsers include PP-attachment ambiguity, NP-bracketing ambiguity, and coordination ambiguity. Not all ambiguities are preserved across languages. We examine a common type of ambiguity in English that is not preserved in Chinese: given a sequence “VP NP PP”, should the PP be attached to the main verb, or to the object noun phrase? We present a discriminative method for exploiting bilingual Chinese-English word alignments to resolve this ambiguity in English. On a held-out test set of Chinese-English parallel sentences, our method achieves 86.3% accuracy on this PP-attachment disambiguation task, an improvement of 4% over the accuracy of the baseline Collins parser (82.3%).

An example of ambiguous PP-attachment in English is shown in Figure 2.7: the PP “from reporters” can modify the VP “answered” or the NP “questions”.

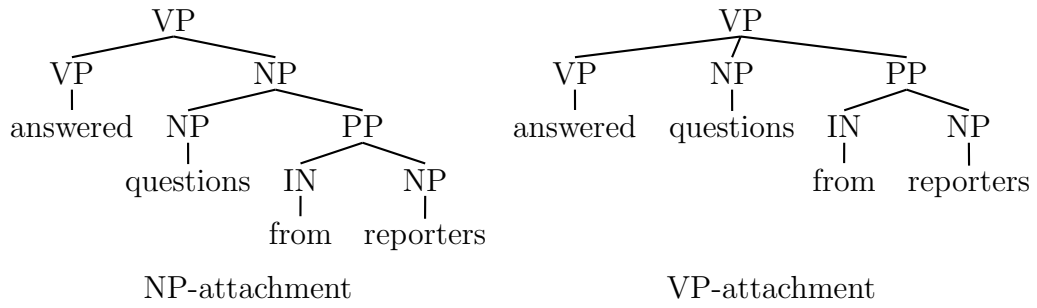


Figure 2.7: PP-attachment ambiguity in English

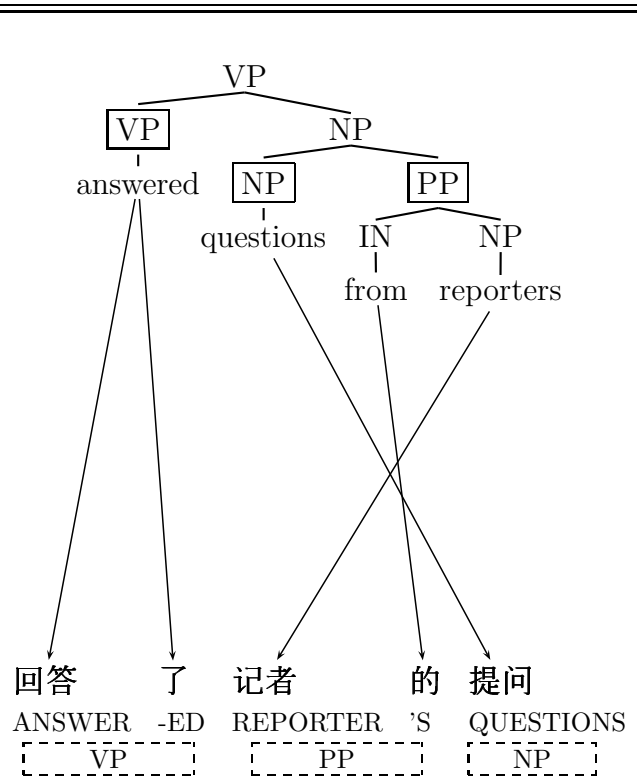


Figure 2.8: Resolving PP-attachment ambiguity using Chinese-English word alignments

As long as syntactic ambiguities are not preserved across languages, we can use bilingual word alignments to disambiguate the construction. For example, in Chinese, PP’s generally appear directly before the head that they modify. Thus, PP-attachment ambiguity is not preserved from English to Chinese. Given the bilingual word alignments shown in Figure 2.8, we can deduce that the ordering of constituents on the Chinese side is “VP PP NP”, indicating that the PP modifies the NP in Chinese, and presumably therefore in English as well.

2.3.1 Related Work

PP-Attachment Disambiguation

Most previous work in PP-attachment disambiguation for English, whether unsupervised [36, 91] or supervised [6, 20], has focused on *monolingual* information such as relationships among the lexical heads of the VP (“answered”), NP (“questions”), and PP (“from”) constituents, as well as the lexical head of the NP dominated by the PP (“reporters”). The statistical parsers of Charniak [10] and Collins [17] implement a variety of monolingual lexical and structural features to resolve syntactic ambiguities while constructing parse trees.

In contrast to these approaches, our approach uses *bilingual* word alignments to resolve PP-attachment ambiguity in English. In this respect, our approach is similar to that of Schwartz et al. [95], who leverage Japanese-English parallel bitext to improve the resolution of PP-attachment ambiguity on monolingual English text. In keeping with the approaches of Hindle and Rooth [36] and [91], Schwartz et al. [95] estimate the probability of each possible attachment decision as follows: they first identify unambiguous instances of PP-attachment in English text, then compute the relative frequency of each attachment decision using these instances, conditioned on the verbs, nouns, and prepositions (or some subset thereof) appearing in the ambiguous construction. They subsequently use these statistics, computed over unambiguous

instances, to estimate the probability of a PP attaching to an NP or VP in unseen (potentially ambiguous) English text.

Schwartz et al. [95] differs from the other unsupervised approaches in that the authors use *bilingual* information to identify unambiguous instances of PP-attachment. Specifically, they exploit the fact that PP-attachment is strictly unambiguous in Japanese by parsing both sides of a Japanese-English parallel bitext into LF, aligning nodes in the LF, and using the PP-attachment decision dictated by the Japanese side to infer the correct attachment decision on the English side. The authors evaluate their approach in two MT applications: English-Japanese and English-Spanish translation. They compare against a baseline method of PP-attachment ambiguity resolution that does *not* make use of the relative frequency statistics collected from the bilingual Japanese-English corpus. Their method improves Japanese-English translation quality but decreases Spanish-English translation quality, as measured by human evaluation.

Our work differs from that of Schwartz et al. [95] in several ways. First, because they evaluate their PP-attachment method only indirectly (by measuring its impact on English-to-Japanese and English-to-Spanish MT tasks), and not directly (by measuring the improvement in accuracy on the PP-attachment task), it is difficult to conclude from their results how effective their method is at improving PP-attachment accuracy (especially since their results in MT were mixed, with English-Japanese translation quality improving but English-Spanish translation quality worsening). In contrast, we evaluate our method directly on a PP-attachment task, and obtain a statistically significant gain of 4.0% in accuracy over the baseline Collins parser. Second, their method is unsupervised but requires a large parallel English-Japanese bitext in order to obtain reliable statistics of relative frequency for each set of lexical items; in contrast, our method is supervised but requires only a few hundred sentences of parallel English-Chinese bitext with manual parses on the English side during training.

Finally, Schwartz et al. [95] implement hard cutoffs based on lexical associations, while we use a variety of features whose weights are learned discriminatively; thus our method appears to be more easily applicable to other problems in monolingual syntactic ambiguity resolution besides PP-attachment.

Bilingual Corpora for Monolingual Analysis

Yarowsky and Ngai [105] use bilingual word alignments to project part-of-speech taggers and NP-bracketers across languages; Hwa et al. [40, 39] extend this work to project syntactic dependency analyses across languages. Our work is similar to these approaches in that we use bilingual word alignments to induce a syntactic correspondence between languages; however, our focus is not on inducing analyses in the *target* language of the projection. Instead, we induce a syntactic correspondence from the source to the target language, then use that projection to resolve ambiguities in the syntactic analysis on the *source* side.

Burkett and Klein [9] parse both sides of a parallel English-Chinese bitext to generate a k -best list of English parses and a k -best list of Chinese parses, then rerank the $k \times k$ -best list of English/Chinese parse tree pairs using the score assigned to each tree by the baseline parser; features of the word alignment; and features measuring structural correspondence between the English and Chinese trees in each pair. They obtain improvements in monolingual parse accuracy for both English and Chinese relative to state-of-the-art baseline English and Chinese parsers, and they obtain gains in translation quality when training a syntax-based MT system using the reranked trees. Our approach is different from that of Burkett and Klein [9] in that we do not *rerank* a k -best list of parses; instead, we restrict ourselves to *repairing* common sources of attachment errors in English parses (specifically, PP-attachment).

2.3.2 Our Work

The main contribution of this work is the use of Chinese-English bilingual word alignments to resolve PP-attachment ambiguity in English. Specifically, we address the following binary classification problem: given a “VP NP PP” sequence in English, should the PP be attached to the VP or the NP? To answer this question, we consider all instances of “VP NP PP” sequences in a bilingual corpus for which we have automatic Chinese-English word alignments, automatic English parses, and gold-standard English parses. In the following sections, we discuss two instances of PP-attachment ambiguity and present features of the automatic word alignment that can be used to determine whether the PP should be attached to the VP or the NP. We then describe our procedure for training a perceptron classifier using these features, and compare the accuracy of our classifier on this PP-attachment task against the accuracy of the PP-attachment decisions made by the baseline Collins parser.

2.3.3 Bilingual Alignments and PP-attachment Ambiguity

Figure 2.7 illustrates a case where the correct attachment site of the PP (“from reporters”) is the NP (“questions”). To determine the correct attachment site for the English PP using the word alignments as a guide, we proceed as follows:

- **Project tags:** Wherever there is a VP NP PP sequence in the English parse tree, each node dominates a span of English words, and each English word is aligned with zero or more Chinese words. Label each of those aligned Chinese words with the category of the associated English node: “VP”, “NP”, or “PP”. Figure 2.8 gives an example; the resulting projected tag sequence on the Chinese side is “VP VP PP PP NP”.
- **Merge identical tags:** Merge adjacent labels in the Chinese tag sequence if they are identical, and ignore any words that have not received a projected

tag. In Figure 2.8, the final merged tag sequence is “VP PP NP”. If a Chinese word receives more than one projected English tag, create a new hybrid tag combining the English tags for that word (for example, if the English VP and NP project onto the same Chinese word, that word receives a tag of “VP/NP”.)

- **Determine Chinese PP-attachment site:** As a general rule, PP’s in Chinese modify the head directly following them, so the Chinese tag sequence “VP PP NP” implies NP-attachment for the PP. We use a perceptron to learn such rules, predicting NP- versus VP- attachment.
- **Deduce English PP-attachment site:** Assuming that the PP-attachment dependency relation is preserved across languages, we can infer that the English PP should most likely be attached to the English NP.

Figure 2.9 illustrates a case where the correct attachment of the PP is to the main verb instead of to the object noun. In this case, tag projection proceeds as above; the resulting projected Chinese tag sequence is “NP PP VP”, thus indicating that the PP modifies the VP (Figure 2.10).

There are two stages involved in disambiguating PP-attachment correctly. First, a parser must correctly label and bracket the main VP, object NP, and PP. Second, the parser must correctly choose an NP or VP attachment site for the PP. Since the latter problem is the focus of this work, we limit the scope of our classification task to those instances where the base VP, NP, and PP constituents have been labelled and bracketed correctly by the automatic parser.⁹

In order to identify whether these constituents have been bracketed correctly, we refer to the gold standard parses. We then put the gold-standard parses aside, and return to the following problem: given a sequence “VP NP PP” that the automatic

⁹Note that relaxing this restriction would affect our absolute performance numbers, but it would have no effect on our performance relative to the Collins parser: the Collins parser by definition fails on any case we have excluded.

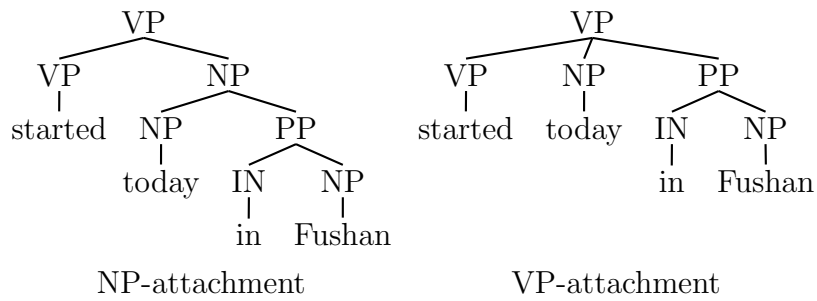


Figure 2.9: PP-attachment ambiguity in English

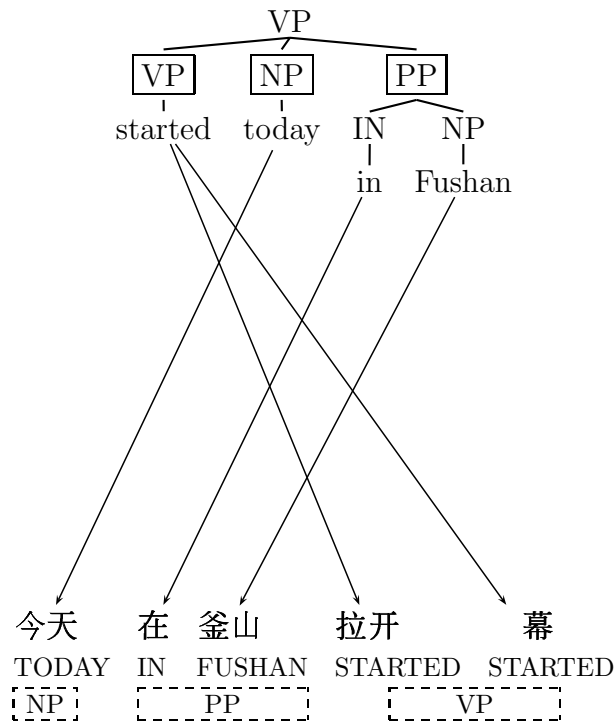


Figure 2.10: Resolving PP-attachment ambiguity using Chinese-English word alignments

parser has correctly labelled and bracketed, build a classifier that uses features of the automatic parse and the automatic bilingual word alignment to predict whether the PP should be attached to the VP or to the NP. To measure the accuracy of our classifier, we compute the percentage of correct attachment decisions, and compare this against the percentage of correct attachment decisions made by the baseline automatic parser.

2.3.4 Methods

Data Sets

Our training and test sets consist of bilingual Chinese-English sentence pairs that have been automatically parsed on the English side using the Collins parser [17], manually parsed on the English side to produce the gold-standard parses, and automatically word-aligned using GIZA++ with *refined* symmetrization [85]. GIZA++ is trained on 10M sentence pairs, but the total size of our PP-attachment data sets is 800 sentence pairs excerpted from LDC2007T02[3], from which we extract 300 instances of potentially ambiguous PP-attachment. We train and test our PP-attachment binary classifier on these 300 instances using 10-fold cross-validation.

Features

In addition to the feature **collinsParserAttachment**, which is the attachment decision made by the baseline Collins parser, our feature set includes two types of features: lexical and alignment-based.

Lexical Features

- **englishPrepositionHead**: the lexical head of the English PP (In Figure 2.10, this is “in”.)

- **projectedChinesePrepositionHead**: the Chinese word or words aligned to the lexical head of the English PP.

Alignment-Based Features

- **projectedChineseTagSequence**: the sequence of part-of-speech tags after projection from English to Chinese (In Figure 2.10, this is “NP PP VP”.)
- **projectedChineseTagSeqLength**: the number of tokens in the sequence of part-of-speech tags after projection from English to Chinese (In Figure 2.10, this is 3.)
- **initialChineseTag**: the initial tag in the projected Chinese sequence
- **projectedChineseTagAfterFirstPP**: the tag immediately following the first occurrence of a PP in the projected Chinese sequence (In Figure 2.10, this is “VP”.)
- **projectedFinalChineseTag**: the final tag in the projected Chinese sequence
- **splitNP**: whether or not the English NP tag was split into discontinuous tags on the Chinese side during projection
- **splitPP**: whether or not the English PP tag was split into discontinuous tags on the Chinese side during projection
- **splitVP**: whether or not the English VP tag was split into discontinuous tags on the Chinese side during projection

Discriminative Training

We train a perceptron for binary classification [92] to solve the PP-attachment problem using the features described in Section 2.3.4. We initialize the weights w of

all features h to 0, and the bias b to 0. We make multiple passes over the training data. For each sentence pair in the training data, we represent the sentence pair as a vector x , where x_i is the value of feature h_i for the sentence pair. Our predicted attachment y_{hyp} is NP-attachment if $w \cdot x + b > 0$ and VP-attachment otherwise. If our predicted attachment y_{hyp} matches the gold-standard attachment decision y_{gold} , then the example is correctly classified and we proceed to the next example. Otherwise, the example is incorrectly classified and we update the weights so that $w' = w + y_{gold} \times x$ and the bias so that $b' = b + y_{gold}$. We stop training when the number of incorrect classifications no longer decreases on the training set. After training, we return the average weight vector over all iterations of training, following Collins [19].

2.3.5 Results

After training a binary perceptron classifier, we apply our classifier during testing to instances where the automatic parser has correctly identified “VP NP PP” sequences, and we predict the attachment site of the PP using the features described in Section 2.3.4 and the learned weights.

Due to the limited size of our data set (we use 800 sentence pairs of parallel Chinese-English text), we train and test our classifier using 10-fold cross-validation. We extract 300 instances of “VP NP PP” sequences from 800 sentences of parallel data, and divide this set of 300 instances into 10 sets of 30 instances each. We train on 9 of the sets, and measure accuracy on the held-out set. We then average the test set accuracy over all 10 iterations of cross-validation.

We measure the accuracy of our method by classifying each instance of “VP NP PP” appearing in the test set as either attachment to the NP or attachment to the VP, and compare the accuracy of our method against the Collins parser baseline (Table 2.12). Our method achieves an average accuracy of 86.3% on held-out test sets in 10-fold cross validation, compared to 82.3% for the baseline Collins parser

Data Set	Total Instances	Correct	Incorrect	%Correct
train	152	128	24	84.2%
test1	71	55	16	77.5%
test2	59	49	10	83.1%

Table 2.11: PP-Attachment Accuracy of Baseline Collins Parser

Method	% Correct PP-Attachment
Collins parser	82.3%
Perceptron classifier	86.3%

Table 2.12: PP-attachment Accuracy of Perceptron Classifier vs. Baseline Collins Parser

(Table 2.11). This improvement in accuracy of 4% is statistically significant under a paired t -test ($p=0.015$).

To determine the relative contribution of each type of feature to the accuracy of the classifier, we perform feature ablation: we remove each type of feature from consideration in turn, and measure the impact upon classifier accuracy relative to the accuracy achieved using *all* feature types. Table 2.13 illustrates the impact of removing each feature type upon the classifier accuracy; features are listed in the order of greatest impact upon classifier accuracy.

2.3.6 Conclusion

In our work on combining constituent parsers, we have explored existing methods for parser combination and examined their impact upon parse quality and translation quality. We have presented a novel algorithm for parse hybridization by recombining context-free productions.

While an existing method (recombining constituents) results in the highest parse f-measure of the methods explored, our method (recombining context-free productions) produces trees which better preserve the syntactic structure of the individual parses,

Features	% Accuracy
All Features	86.3
-projectedChinesePrepositionHead	84.7
-splitVP	85.3
-splitNP	85.3
-splitPP	85.3
-englishPrepositionHead	85.7
-finalChineseTag	85.7
-projectedChineseTagAfterFirstPP	86.0
-initialChineseTag	86.0
-projectedChineseTagSequence	86.3
-projectedChineseTagSeqLength	86.3

Table 2.13: Feature Ablation: Accuracy of PP-Attachment Classifier with Individual Features Removed

thereby achieving higher BLEU scores in our Arabic-English syntax-based translation experiments. We have also presented an efficient linear-time algorithm for selecting the parse with maximum expected f-measure.

In our work on resolving structural ambiguities in English, we have presented a method for English PP-attachment disambiguation using automatic bilingual Chinese-English word alignments, achieving an improvement in accuracy of 4.0% over the baseline statistical parser. Our results validate our hypothesis that bilingual information can help to resolve monolingual ambiguities as long as the ambiguities are not preserved across languages.

2.4 Summary of Contributions

In this chapter, we have presented two ways to improve upon the accuracy of a state-of-the-art constituent parser: first, by combining the output of multiple parsers, and second, by incorporating bilingual word alignments as a feature in resolving syntactic ambiguity. The primary contributions of this work are the following:

- An efficient, linear-time algorithm for selecting the parse with maximum expected f-measure from an n -best list of candidates, using a Minimum Bayes Risk approach
- A novel method for parse hybridization that recombines context-free productions, instead of constituents, producing better BLEU scores in a downstream syntax-based MT application
- A result validating the hypothesis that bilingual Chinese-English word alignments can be used to resolve a common source of syntactic ambiguity in English

Chapter 3

Improving Word Alignment Accuracy Using Parsing

Word alignments that violate syntactic correspondences interfere with the extraction of string-to-tree transducer rules for syntax-based machine translation. We present an algorithm for identifying and deleting such word alignment links. We train a state-of-the-art syntax-based machine translation system on these corrected alignments, and obtain gains in both alignment quality and translation quality in Chinese-English and Arabic-English translation experiments relative to a baseline system trained on GIZA++ union alignments.

3.1 Background

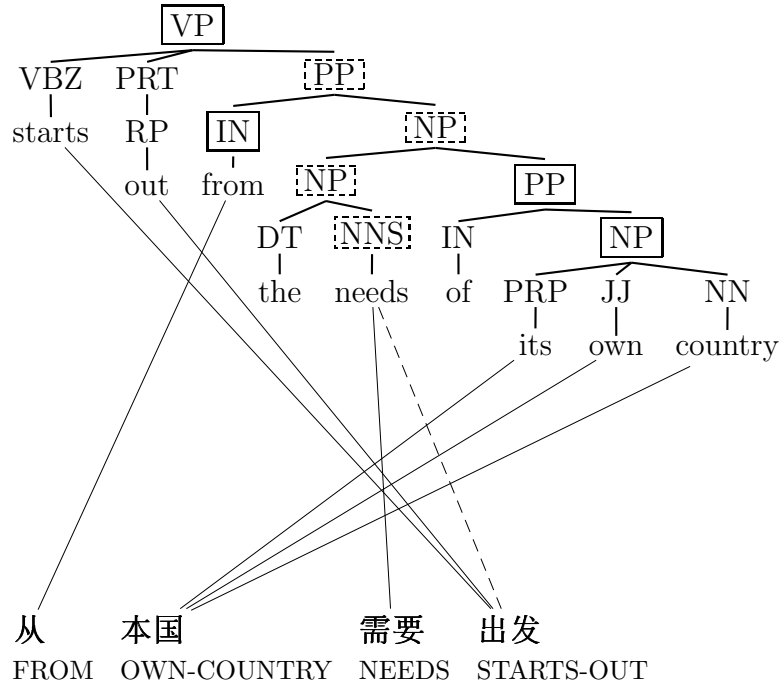
Automatic word alignment typically constitutes the first stage of the statistical machine translation pipeline. GIZA++ [85], an implementation of the IBM [8] and HMM [101] alignment models, is the most widely-used alignment system. Because of an asymmetry in the IBM alignment models, one-to-many alignments are permitted in the target-to-source direction but not in the source-to-target direction. To remedy this asymmetry, the alignment models are run twice, once in the source-to-target and

once in the target-to-source directions, and the two sets of unidirectional alignments are combined, or *symmetrized*, using heuristics. GIZA++ alignments with *union* symmetrization have been used in the state-of-the-art syntax-based statistical MT system described in Galley et al. [33] and in the hierarchical phrase-based system Hiero [14]. GIZA++ alignments with *refined* symmetrization have been used in state-of-the-art phrase-based statistical MT systems [86]; variations on the refined heuristic have been used by [46] (*diag* and *diag-and*) and by the phrase-based system Moses (*grow-diag-final*) [45].

GIZA++ union alignments have high recall but low precision, while *intersection* or *refined* alignments have high precision but low recall.¹ There are two natural approaches to improving upon existing GIZA++ alignments, then: deleting links from union alignments to improve precision, or adding links to intersection or refined alignments to improve recall. In this work, we delete links from GIZA++ union alignments to improve precision.

The low precision of GIZA++ union alignments poses a particular problem for syntax-based rule extraction algorithms [90, 33, 38, 66]: if an alignment link violate syntactic correspondences between the source and target languages, it forces the rule extraction algorithm to extract rules that are large in size, few in number, and poor in generalization ability. Figure 3.1 illustrates this problem: the dotted line represents an incorrect link in the GIZA++ union alignment. Using the rule extraction algorithm described in Galley et al. [34], we extract the rules shown in the leftmost column (R1–R4). Rule R1 is large and unlikely to generalize well. If we delete the incorrect link in Figure 3.1, we can extract the rules shown in the rightmost column (R2–R9): Rule R1, the largest rule from the initial set, disappears, and several smaller, more modular rules (R5–R9) replace it.

¹For a complete discussion of alignment symmetrization heuristics, including union, intersection, and refined, refer to Chapter 1, Section 1.2.3.



Rules Extracted Before Deleting Dotted Link	Rules Extracted After Deleting Dotted Link
<p>R1: → x0 x1 需要 出发</p> <p>R2: → 从</p> <p>R3: → x0</p> <p>R4: → 本国</p>	<p>R5: → x0 x1</p> <p>R6: → x1 x0</p> <p>R7: → x0</p> <p>R8: → 需要</p> <p>R9: → x0 出发</p>

Figure 3.1: Impact of incorrect alignment links upon extraction of tree-to-string transducer rules. Using all alignment links yields rules R1–R4. Deleting the dotted alignment link yields rules R2–R4 as before; rule R9 instead of rule R1; and additional rules R5–R8.

In this work, we present a supervised algorithm that uses these two features of the extracted syntax-based translation rules (size of largest rule and total number of rules), as well as a handful of structural and lexical features, to automatically identify and delete incorrect links from GIZA++ union alignments. We show that link deletion improves alignment quality and translation quality in Chinese-English and Arabic-English MT, relative to a strong baseline. Our link deletion algorithm is easy to implement, runs quickly, and has been used by a top-scoring MT system in the Chinese newswire track of the 2008 NIST evaluation. The primary contributions of this work are as follows:

- We present a supervised algorithm for identifying and deleting incorrect links from an existing alignment, using syntactic features
- We demonstrate that link deletion improves alignment quality and translation quality
- We show that link deletion is easy to implement, runs quickly, and requires only 100-200 sentences of training data

3.1.1 Related Work

Discriminative Methods for Word Alignment

In recent years, discriminative methods for alignment have rivalled and in some cases surpassed the accuracy of unsupervised methods [65, 42, 98, 78, 47, 2, 31]. However, except for Fraser and Marcu [31], none of these advances in alignment quality has improved the translation quality of a state-of-the-art system. We use a discriminatively trained model to identify and delete incorrect links, and demonstrate that these gains in alignment quality lead to gains in translation quality in a state-of-the-art syntax-based MT system. In contrast to the semi-supervised LEAF alignment algorithm of Fraser and Marcu [31], which requires 1,500-2,000 CPU *days* per iteration

to align 8.4M Chinese-English sentences (anonymous, p.c.), link deletion requires only 450 CPU *hours* to re-align such a corpus (after initial alignment by GIZA++, which requires 20-24 CPU days).

Syntax-Based Word Alignment

Several recent works incorporate syntactic features into alignment. May and Knight [74] use syntactic constraints to re-align a parallel corpus that has been aligned by GIZA++ as follows: they extract string-to-tree transducer rules from the corpus, the target parse trees, and the alignment; discard the initial alignment; use the extracted rules to construct a forest of possible string-to-tree derivations for each string/tree pair in the corpus; use EM (expectation-maximization) [23] to select the Viterbi derivation tree for each pair; and finally, induce a new alignment from the Viterbi derivations, using the re-aligned corpus to train a syntax-based MT system. May and Knight [74] differs from our approach in two ways: first, the set of possible re-alignments they consider for each sentence pair is limited by the initial GIZA++ alignments seen over the training corpus, while we consider all alignments that can be reached by deleting links from the initial GIZA++ alignment for that sentence pair. Second, May and Knight [74] use a time-intensive training algorithm to select the best re-alignment for each sentence pair, while we use a fast greedy search to determine which links to delete; in contrast to May and Knight [74], who require 400 CPU hours to re-align 330k Chinese-English sentence pairs (anonymous, p.c.), link deletion requires only 18 CPU hours to re-align such a corpus.

Lopez and Resnik [69] and DeNero and Klein [25] modify the distortion model of the HMM alignment model [101] to reflect tree distance rather than string distance; Cherry and Lin [12] modify an ITG aligner by introducing a penalty for induced parses that violate syntactic bracketing constraints. Similarly to these approaches,

we use syntactic bracketing to constrain alignment, but our work extends beyond improving alignment quality to improve translation quality as well.

3.2 Methods

We propose an algorithm to re-align a parallel bitext that has been aligned by GIZA++ (IBM Model 4), then symmetrized using the union heuristic. We then train a syntax-based translation system on the re-aligned bitext, and evaluate whether the re-aligned bitext yields a better translation model than a baseline system trained on the GIZA++ union aligned bitext.

3.2.1 Link Deletion Algorithm

Our algorithm for re-alignment (shown in Algorithm 4) proceeds as follows. We make a single pass over the corpus. For each sentence pair, we initialize the alignment $A = A_{initial}$ (the GIZA++ union alignment for that sentence pair). We represent the score of A as a weighted linear combination of features h_i of the alignment A , the target parse tree $parse(e)$ (a phrase-structure syntactic representation of e), and the source string f :

$$score(A) = \lambda \cdot h(A, parse(e), f)$$

We define a *branch* of links to be a *contiguous* 1-to-many alignment. In Figure 3.1, the 1-to-many alignment formed by {本国-its, 本国- own, 本国-country} constitutes a branch, but the 1-to-many alignment formed by {出发-starts, 出发-out, 出发-needs} does not, since the English words “starts”, “out”, and “needs” to which the foreign word 出发 is aligned are not contiguous in the English sentence; instead, the words “from the”, which are *not* aligned to 出发, intervene between “out” and “needs”, creating a discontinuity on the English side.

We define two alignments, A and A' , to be *neighbors* if they differ only by the deletion of a link or *branch* of links. We consider all alignments A' in the *neighborhood* of A , greedily deleting the link l or branch of links b that maximizes the score of the resulting alignment $A' = A \setminus l$ or $A' = A \setminus b$. We delete links until no further increase in the score of A by deleting another link or branch of links is possible. While using a dynamic programming algorithm would likely improve search efficiency and allow link deletion to find an optimal solution, in practice, the greedy search runs quickly, and improves alignment quality.

3.2.2 Features

In this section, we describe the features $h_i(A, parse(e), f)$ used to score each alignment A .

Syntactic Features

We use two features of the string-to-tree transducer rules extracted from A , $parse(e)$, and f according to the rule extraction algorithm described in Galley et al. [34]:

- **ruleCount:** Total number of rules extracted from A , $parse(e)$, and f . As Figure 3.1 illustrates, incorrect links violating syntactic brackets tend to decrease **ruleCount**; **ruleCount** increases from 4 to 8 after deleting the incorrect link.
- **sizeOfLargestRule:** We define an *internal* node to be any node in the target parse tree that is not a terminal node. We measure the size of a target parse tree by counting the total number of internal nodes it contains. The feature **sizeOfLargestRule** measure the size of the single largest rule extracted from A , $parse(e)$, and f . In Figure 3.1, the largest rule in the leftmost column is R1, which has a total of 9 internal nodes: VP, PRT, PP, NP, NP, VBZ, RP,

Algorithm 4: Link Deletion Algorithm

input : An initial alignment $A_{initial}$ from which to delete links; an alignment $A_{protected} \subseteq A_{initial}$ specifying which links in $A_{initial}$ to protect from deletion

output: An alignment A_{final} , where $A_{final} \subseteq A_{initial}$ and $A_{final} \supseteq A_{protected}$; a target parse $parse(e)$; and a source string f

```
A = Ainitial;
score(A) = λ · h(A, parse(e), f);
neighbors(A) = FindNeighbors(A);
while neighbors(A) ≠ ∅ do
  for A' ∈ neighbors(A) do
    | score(A') = λ · h(A', parse(e), f);
  end
  Â = arg maxA' ∈ neighbors(A) score(A')
  if score(Â) > score(A) then
    | A = Â;
    | neighbors(A) = FindNeighbors(A);
  end
  else
    | return A;
  end
end
FindNeighbors(A)
  neighbors(A) = ∅;
  for l ∈ A do
    | A' ← A \ l;
    | neighbors(A) = neighbors(A) ∪ A';
  end
  foreach branch of links b ∈ A do
    | A' ← A \ b;
    | neighbors(A) = neighbors(A) ∪ A';
  endfch
  return neighbors(A);
end
```

DT, and NNS. The largest rule in the rightmost column is R9, which contains 4 internal nodes: VP, VBZ, PRT, and RP. Thus, the value of **sizeOfLargestRule** decreases from 9 to 4 after the deletion of the incorrect link 出发- needs.

Structural Features

- **wordsUnaligned:** Total number of unaligned words. In Figure 3.1, 2 English words (“the”, “of”) and 0 Chinese words are unaligned, so the value of this feature is 2.
- **1-to-many Links:** Total number of links for which one word is aligned to multiple words, in either direction. In Figure 3.1, the links {出发-starts, 出发-out, 出发-needs} represent a 1-to-many alignment. 1-to-many links appear more frequently in GIZA++ union alignments than in gold alignments, and are therefore good candidates for deletion. The category of 1-to-many links is further subdivided, depending on the degree of *contiguity* that each link exhibits with the other links participating in the 1-to-many link. This feature is motivated by the observation that, in a manually aligned Chinese-English corpus, 82% of the Chinese words that are aligned to multiple English words are aligned to a *contiguous* block of English words; similarly, 88% of the English words that are aligned to multiple Chinese words are aligned to a *contiguous* block of Chinese words [26]. Thus, if a Chinese word is correctly aligned to multiple English words, those English words are likely to be adjacent to each other, and if an English word is correctly aligned to multiple Chinese words, those Chinese words are likely to be adjacent to each other. Each link in a 1-to-many alignment can have 0, 1, or 2 adjacent links. For example, consider the 1-to-many alignment in Figure 3.1 between the Chinese word 出发 and the English words “starts”, “out”, and “needs”:

- **zeroAdjacentLinks:** In Figure 3.1, the link 出发-needs has 0 adjacent links.
- **oneAdjacentLink:** In Figure 3.1, the links 出发-starts and 出发-out each have 1 adjacent link—namely, each other.
- **twoAdjacentLinks:** In Figure 3.1, the link 本国-own has 2 adjacent links, namely 本国-it and 本国-country.

Lexical Features

- **highestLexProbRank:** A link e_i - f_j is “max-probable from e_i to f_j ” if $p(f_j|e_i) > p(f_{j'}|e_i)$ for all alternative words $f_{j'}$ with which e_i is aligned in $A_{initial}$. In Figure 3.1, it happens that $p(\text{需要} | needs) > p(\text{出发} | needs)$, so 需要-needs is max-probable for “needs”. The definition of “max-probable from f_j to e_i ” is analogous, and a link is max-probable (nondirectionally) if it is max-probable in either direction. The value of **highestLexProbRank** is the total number of max-probable links. The conditional lexical probabilities $p(e_i|f_j)$ and $p(f_j|e_i)$ are estimated using frequencies of aligned word pairs in the high-precision GIZA++ *intersection* alignments for the training corpus.

History Features

In addition to the above syntactic, structural, and lexical features of A , we also incorporate two features of the link deletion history itself into $Score(A)$:

- **linksDeleted:** Total number of links deleted from $A_{initial}$ thus far. At each iteration, either a link or a branch of links is deleted. This feature serves as a constant cost function per link deleted.

- **stepsTaken:** Total number of iterations thus far in the search; at each iteration, either a link or a branch is deleted. This feature serves as a constant cost function per step taken during link deletion.

3.2.3 Constraints

In addition to the features described in Section 3.2.2, we also introduce two fixed constraints upon each alignment.

Protecting High-Precision Links from Deletion

GIZA++ refined links have higher precision than union links. For example, on a 400-sentence-pair Chinese-English data set, GIZA++ union alignments have a precision of 77.3 while GIZA++ refined alignments have a precision of 85.3. Because refined alignment links have higher precision and are therefore more likely to be correct than union links which are not in the refined set, we do not consider any refined links for deletion.

Given the higher precision of refined alignment links as compared to union links, a natural question to ask is: why not train a syntax-based MT system on the refined links instead of the union links? The problem is that, while refined links have higher precision than union links, they also suffer from lower recall. Because the refined links are a subset of the union links, they are, by definition, less dense. This sparsity allows rule extraction to extract many more rules from refined links than from union links. Unfortunately, these additional rules tend to be incorrect rules (i.e. rules that do not appear in the gold set) more often than not, resulting in a higher rule recall but lower rule precision when using refined alignment links. To quantify the difference in rule precision, recall, and f-measure between the rules extracted from GIZA++ union and refined alignments, we compare the rules extracted from each alignment on a set of 400 Chinese-English sentences. We find that GIZA++ union

alignments have higher rule precision (50.5) than GIZA++ refined alignments (44.2), while GIZA++ refined alignments have higher rule recall (54.1) than GIZA++ union alignments (44.2) (Table 3.4). To see how GIZA++ refined alignments compare to GIZA++ union alignments for syntax-based translation, we compare systems trained on each set of alignments for a Chinese-English translation task.² Union alignments result in a test set BLEU score of 41.2, as compared to only 37.0 for refined. Thus, in this work, we seek a set of alignment links that is a subset of the union set but a superset of the refined set, hoping to improve upon the precision of union links while avoiding the drop in recall suffered by refined links.

Deleting Links Between Common English-Chinese Word Pairs

In our Chinese-English corpora, the 10 most common English words (excluding punctuation marks) include {a,in,to,of,and,the}, while the 10 most common Chinese words include {了,是,在,和,的}. Of these, {a,and,the} and {了,的} have no explicit translational equivalent in the other language. These words are aligned with each other frequently (and erroneously) by GIZA++ union, but rarely in the gold standard alignment. We delete all links in the set {a, an, the} \times {的, 了} from $A_{initial}$ as a preprocessing step. The direct impact upon alignment f-measure of deleting these links is small; on Chinese-English Data Set *A*, the f-measure of the baseline GIZA++ union alignments on the test set increases from 63.4 to 63.8 after deleting these links, while the remaining increase in f-measure from 63.8 to 75.1 (shown in Table 3.3) is due to the link deletion algorithm itself.

3.2.4 Discriminative Training Using Averaged Perceptron

We set the feature weights λ using a modified version of averaged perceptron learning with structured outputs [19]. The training algorithm is described in Algorithm 5.

²Chinese-English task *A*, described in Section 3.3.

Following Moore [77], we initialize the value of our expected most informative feature (**ruleCount**) to 1.0, and initialize all other feature weights to 0. During each training epoch, or pass over the discriminative training set, we “decode” each sentence pair by greedily deleting links from $A_{initial}$ in order to maximize the score of the resulting alignment using the current settings of λ .

We construct a set of candidate alignments $A_{candidates}$ for use in reranking as follows. Starting with $A = A_{initial}$, we iteratively explore all alignments A' in the *neighborhood* of A , adding each *neighbor* to $A_{candidates}$, then selecting the *neighbor* that maximizes $Score(A')$. When it is no longer possible to increase $Score(A)$ by deleting any links, link deletion concludes and returns the highest-scoring alignment, $A_{predicted}$.

In general, $A_{gold} \notin A_{candidates}$; following Collins [18] and Charniak [11] for parse reranking and Liang et al. [64] for translation reranking, we define A_{oracle} as alignment in $A_{candidates}$ that is most *similar* to A_{gold} according to weighted fully-connected alignment f-measure.³ We update each feature weight λ_i towards the oracle alignment as follows: $\lambda_i = \lambda_i + h_i^{A_{oracle}} - h_i^{A_{predicted}}$. Updating towards the oracle alignment is referred to as *local* updating, since the changes made to each feature weight are likely to be small; updating towards the gold alignment is referred to as *bold* updating, since the changes made to each feature weight are likely to be large. Liang et al. [64] report that, for translation reranking, local updates outperform bold updates [64].

Following Moore [77], after each training pass, we average all feature weight vectors seen during that pass, and decode the training set using the vector of averaged feature weights. When alignment quality stops increasing on the training set, perceptron training ends. The weight vector returned by perceptron training is the average over the training set of all weight vectors seen during all iterations; averaging reduces overfitting on the training set [19].

³We discuss alignment similarity metrics in detail in Chapter 1, Section 1.2.9.

Algorithm 5: Discriminative Training with Averaged Perceptron

input : A set s of training sentences; an initial alignment $A_{initial}$ for each sentence in s ; a gold-standard alignment A_{gold} for each sentence in s ; a set of feature functions h ; a learning rate η
output: A vector of feature weights λ for each feature function in h

Initialize feature weights λ ;

foreach *training epoch* **do**

foreach *sentence* $s_i \in s$ **do**

$A = A_{initial}$;

$score(A) = \lambda \cdot h(A, parse(e), f)$;

$push(PriorityQueue, (A, score(A)))$;

$Candidates = \emptyset$;

while $PriorityQueue \neq \emptyset$ **do**

$A = pop(PriorityQueue)$;

$Candidates = Candidates \cup A$;

$neighbors(A) = FindNeighbors(A)$;

for $A' \in neighbors(A)$ **do**

$score(A') = \lambda \cdot h(A', parse(e), f)$;

$push(PriorityQueue, (A', score(A')))$;

$Candidates = Candidates \cup A'$;

end

end

$A_{predicted} = \arg \max_{A' \in Candidates} score(A')$;

$A_{oracle} = \arg \max_{A' \in Candidates} Similarity(A', A_{gold})$;

if $A_{predicted} \neq A_{oracle}$ **then**

foreach *feature* $h_i \in h$ **do**

$\lambda_i = \lambda_i + \eta * (h_i(A_{oracle}) - h_i(A_{predicted}))$;

endfch

end

endfch

endfch

return λ ;

3.3 Experimental Setup

3.3.1 Data Sets

We evaluate the effect of link deletion upon alignment quality and translation quality for two Chinese-English data sets, and one Arabic-English data set. Each data set consists of newswire text, and contains a small subset of manually aligned sentence pairs. We divide the manually aligned subset into a training set (used to discriminatively set the feature weights for link deletion) and a test set (used to evaluate the impact of link deletion upon alignment quality). Table 3.1 lists the source and the size of the manually aligned training and test sets used for each alignment task.

Using the feature weights learned on the manually aligned training set, we then apply link deletion to the remainder (non-manually aligned) of each bilingual data set, and train a full syntax-based statistical MT system on these sentence pairs. After minimum error rate training to set feature weights [84] on a held-out tuning set, we evaluate translation quality on a held-out test set. Table 3.2 lists the source and the size of the training, tuning, and test sets used for each translation task.

3.3.2 Evaluation Metrics

While Alignment Error Rate (AER) and balanced f-measure are standard metrics in the alignments literature, Fraser and Marcu [32] show that improvements in AER or balanced f-measure do not necessarily correlate with improvements in BLEU score. They propose two modifications to balanced f-measure: varying the precision/recall tradeoff using a parameter α which can be tuned according to the language pair, and *fully-connecting* the alignment links before computing f-measure.⁴

⁴We discuss the procedure for fully-connecting an alignment in Chapter 1, Section 1.2.9.

Weighted Fully-Connected Alignment F-Measure

Given a hypothesized set of alignment links H and a gold-standard set of alignment links G , we define $H^+ = \text{fullyConnect}(H)$ and $G^+ = \text{fullyConnect}(G)$, and then compute:

$$f\text{-measure}(H^+) = \frac{1}{\frac{\alpha}{\text{precision}(H^+)} + \frac{1-\alpha}{\text{recall}(H^+)}}$$

After full connection, the alignment shown in Figure 3.1 would also include the links 需要-starts and 需要-out. Fully connected alignments are the alignments that are effectively perceived by both phrase-based and syntax-based machine translation systems, and therefore evaluating over fully-connected alignments more accurately reflects the quality of alignments for machine translation purposes.⁵

For phrase-based Chinese-English and Arabic-English translation tasks, Fraser and Marcu [32] obtain the closest correlation between weighted fully-connected alignment f-measure and BLEU score using $\alpha=0.5$ and $\alpha=0.1$, respectively. We use weighted fully-connected alignment f-measure as the training criterion for link deletion, and to evaluate alignment quality on training and test sets.

Rule F-Measure

To evaluate the impact of link deletion upon rule quality, we compare the rule precision, recall, and f-measure of the rule set extracted from our hypothesized alignments and a Collins-style parser against the rule set extracted from gold alignments and gold parses.⁶

⁵For a detailed discussion of various alignment evaluation metrics, including Alignment Error Rate and balanced f-measure, please refer to Chapter 1, Section 1.2.9.

⁶For a detailed explanation of rule f-measure, please refer to Chapter 1, Section 1.2.9.

Language	Train	Test
Chinese-English <i>A</i>	LDC2007T02[3]	LDC2007T02[3]
	(400 sent.)	(400 sent.)
Chinese-English <i>B</i>	LDC2006E86 newswire[59], LDC2006E93 web[62]	LDC2006E86 newswire[59], LDC2006E93 web[62]
	(1500 sent.)	(1500 sent.)
Arabic-English	LDC2006E86 newswire[59], LDC2006E93 web[62]	LDC2006E86 newswire[59], LDC2006E93 web[62]
	(1500 sent.)	(1500 sent.)

Table 3.1: Data Sets Used in Alignment Link Deletion: Alignment Experiments.

Language	Train	Tune	Test 1	Test 2
Chinese-English <i>A</i>	chi-eng A train (small)	chi-eng A tune newswire	chi-eng A test newswire	–
	(329,031 sent.)	(878 sent.)	(919 sent.)	
Chinese-English <i>B</i>	chi-eng B train (large)	chi-eng B tune newswire	chi-eng B test newswire	–
	(395,055 sent.)	(1,482 sent.)	(1,463 sent.)	
Arabic-English	ara-eng train (large)	ara-eng tune newswire	ara-eng test 1 newswire	ara-eng test 2 test newswire
	(6,561,091 sent.)	(1,178 sent.)	(1,298 sent.)	(765 sent.)

Table 3.2: Data Sets Used in Alignment Link Deletion: Translation Experiments. Contents of Each Data Set Listed in Appendix A, Tables A.1 and A.2.

BLEU

For all translation tasks, we report case-insensitive NIST BLEU scores [87] using 4 references per sentence.

3.3.3 Experiments

Starting with GIZA++ union alignments, we use perceptron training to set the weights of each feature used in link deletion in order to optimize weighted fully-connected alignment f-measure ($\alpha=0.5$ for Chinese-English and $\alpha=0.1$ for Arabic-English) on a manually aligned discriminative training set. We report the (fully-connected) precision, recall, and weighted alignment f-measure on a held-out test set

after running perceptron training, relative to the baseline GIZA++ union alignments. Using the learned feature weights, we then perform link deletion over the GIZA++ union alignments for the entire training corpus for each translation task, as described in Section 3.2.1. Using the resulting alignments, which we refer to as “GIZA++ union + link deletion”, we train a syntax-based translation system similar to that described in Galley et al. [33]. After extracting string-to-tree translation rules from the aligned, parsed training corpus, the system assigns weights to each rule via relative frequency estimation with smoothing. The rule probabilities, as well as trigram language model probabilities and a handful of additional features of each rule, are used as features during decoding. The feature weights are tuned using minimum error rate training [85] to optimize BLEU score on a held-out development set. We then compare the BLEU score of this system against a baseline system trained using GIZA++ union alignments.

In order to compute weighted fully-connected alignment f-measure, we need to choose a value for α , the parameter determining the relative importance of precision and recall when computing f-measure. Choosing a high value for α favors precision; choosing a low value, recall. To determine which value of α is most effective as a training criterion for link deletion, we set $\alpha=0.4$ (favoring recall), 0.5, and 0.6 (favoring precision), and compare the effect on translation quality for Chinese-English data set *A*.

3.4 Results

For each translation task, link deletion improves translation quality relative to a GIZA++ union baseline. For each alignment task, link deletion tends to improve fully-connected alignment precision more than it decreases fully-connected alignment recall, increasing weighted fully-connected alignment f-measure overall.

3.4.1 Chinese-English

On Chinese-English translation task *A*, link deletion increases BLEU score by 1.3 points on tuning and 0.8 points on test (Table 3.3); on Chinese-English translation task *B*, link deletion increases BLEU score by 1.4 points on tuning and 0.5 points on test (Table 3.3), relative to the baseline system.

3.4.2 Arabic-English

On the Arabic-English translation task, link deletion improves BLEU score by 0.8 points on tuning, 0.2 points on test set 1, and 0.6 points on test set 2 (Table 3.3) relative to the baseline system. Note that the training criterion for Arabic-English link deletion uses $\alpha=0.1$; because this penalizes a loss in recall more heavily than it rewards an increase in precision, it is more difficult to increase weighted fully-connected alignment f-measure using link deletion for Arabic-English than for Chinese-English. This difference is reflected in the average number of links deleted per sentence: 4.2 for Chinese-English *B* (Table 3.3), but only 1.4 for Arabic-English (Table 3.3). Despite this difference, link deletion improves translation results for Arabic-English as well.

3.4.3 Varying the Relative Contribution of Precision and Recall to Alignment F-Measure

On Chinese-English data set *A*, we explore the effect of varying α in the weighted fully-connected alignment f-measure used as the training criterion for link deletion. Using $\alpha=0.5$ leads to a higher gain in BLEU score on the test set relative to the baseline (+0.8 points) than either $\alpha=0.4$ (+0.7 points) or $\alpha=0.6$ (+0.7 points), confirming that, for our Chinese-English translation experiments, precision and recall should be assigned equal importance when optimizing alignment f-measure during training.

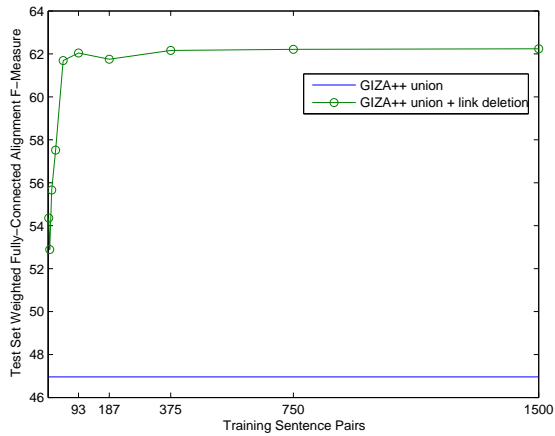
Lang.	Alignment	P	R	α	F	Links Del./ Sent	Grammar Size	BLEU		
								Dev	Test1	Test2
Chi-Eng <i>A</i>	GIZA++ union	54.8	75.4	0.5	63.4	–	23.4M	41.8	41.2	–
Chi-Eng <i>A</i>	GIZA++ union + link deletion	79.6	71.2	0.5	75.1	4.8	59.7M	43.1	41.9	–
Chi-Eng <i>B</i>	GIZA++ union	36.6	66.3	0.5	47.2	–	28.9M	39.6	41.4	–
Chi-Eng <i>B</i>	GIZA++ union + link deletion	65.5	59.3	0.5	62.2	4.2	73.0M	41.0	41.9	–
Ara-Eng	GIZA++ union	35.3	84.1	0.1	73.9	–	52.4M	54.7	50.9	38.2
Ara-Eng	GIZA++ union + link deletion	52.7	79.8	0.1	75.9	1.4	64.9M	55.6	51.1	38.7

Table 3.3: Results of link deletion. Weighted fully-connected alignment f-measure is computed on alignment test sets (Table 3.1); BLEU score is computed on translation test sets (Table 3.2). The average number of links deleted per sentence and the resulting grammar size is shown in each case.

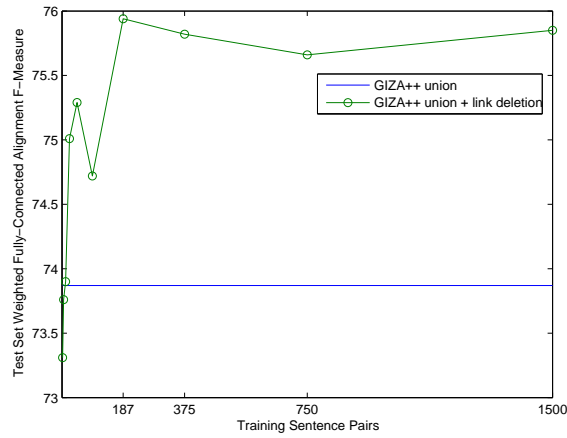
3.5 Discussion

3.5.1 Size of Discriminative Training Set

To examine how many manually aligned sentence pairs are required to set the feature weights reliably, we vary the size of the discriminative training set from 200-1500 sentence pairs while holding test set size constant at 1500 sentence pairs; run perceptron training; perform link deletion on a held-out test set; and record the resulting weighted fully-connected alignment f-measure on the test set. Figure 3.5.1 illustrates that using 100-200 manually aligned sentence pairs of training data is sufficient for Chinese-English; a similarly-sized training set is also sufficient for Arabic-English. While link deletion requires manually aligned data for discriminative training, the amount of such data required to achieve optimal performance with link deletion is minimal (100-200 sentences).



(a) Effect of discriminative training set size on link deletion accuracy for Chinese-English B , $\alpha=0.5$



(b) Effect of discriminative training set size on link deletion accuracy for Arabic-English, $\alpha=0.1$

Alignment	Parse	Rule			
		Precision	Recall	F-measure	Total Non-Unique
gold	gold	100.00	100.00	100.00	12,809
GIZA++ union	Collins	50.5	44.2	47.2	11,021
GIZA++ union + link deletion, $\alpha=0.5$	Collins	47.5	53.2	50.2	13,987
GIZA++ refined	Collins	44.2	54.1	48.6	15,182

Table 3.4: Rule Precision, Recall, and F-Measure of Rules Extracted from 400 Sentence Pairs of Chinese-English data

3.5.2 Effect of Link Deletion on Extracted Rules

Link deletion increases the *size* (as measured by the number of rules) of the extracted grammar. To determine how the *quality* of the extracted grammar changes, we compute the rule precision, recall, and f-measure of the GIZA++ union alignments and various link deletion alignments on a held-out Chinese-English test set of 400 sentence pairs. Table 3.4 indicates the total (non-unique) number of rules extracted for each alignment/parse pairing, as well as the rule precision, recall, and f-measure of each pair. As more links are deleted, more rules are extracted—but of those, some are of good quality and others are of bad quality. Link-deleted alignments produce rule sets with higher rule f-measure than either GIZA++ union or GIZA++ refined.⁷

3.6 Summary of Contributions

In this chapter, we have shown how syntactic features of a parse tree can be used to identify and delete incorrect links from a word alignment. We have presented a link deletion algorithm that improves the precision of GIZA++ union alignments without notably decreasing recall. In addition to lexical and structural features, we use syntax-based features of the rules extracted from an alignment and a parse to identify and delete incorrect links. While the majority of existing previous work in alignments has not led to an improvement in translation quality, our method results in consistent improvements in translation quality over a state-of-the-art syntax-based MT system. Our algorithm runs quickly, and is easily applicable to other language pairs with limited amounts (100-200 sentence pairs) of manually aligned data available.

The primary contributions of this work are the following:

⁷For details on how to compute rule f-measure, please refer to Chapter 1, Section 1.2.9.

- An algorithm for identifying and deleting incorrect alignment links from an existing source-target word alignment, using syntactic features of the target parse tree
- Results demonstrating that link deletion requires only 100-200 sentences of manually aligned data for discriminative training
- Improvements in both alignment quality and translation quality relative to a GIZA++ union baseline in a state-of-the-art syntax-based machine translation system on both Chinese-English and Arabic-English translation tasks

Chapter 4

Using Parsing and Word Alignment to Improve Accuracy of Both Processes Simultaneously

English parses and bilingual word alignments each play a crucial role in the string-to-tree syntax-based machine translation pipeline. By improving the accuracy of automatic parsing and automatic word alignment, we can improve the quality of string-to-tree translation rules extracted from those parses and alignments, and therefore potentially improve the quality of translations output by a system trained on those parses and word alignments. In this chapter, we explore the optimization of parses and alignments simultaneously, allowing each to constrain the other. Specifically, we present an approach to discriminatively reranking n -best lists of parses and m -best lists of alignments to select the (parse, alignment) pair that leads to the optimal set of extracted translation rules.

4.1 Background

Parsing and word alignment jointly determine the quality of translation rules extracted from the training data by a syntax-based machine translation system. Typically, these two processes are performed independently of one another. In Chapter

2, we discussed ways to improve English parse quality using constraints imposed by fixed word alignments. In Chapter 3, we discussed ways to improve bilingual word alignment quality using constraints imposed by fixed parses.

In this chapter, we address the following question: how can we improve upon parse and alignment accuracy simultaneously (i.e. holding neither one fixed, but allowing both to vary)? Ideally, we could search the space of parses and alignments jointly—this would allow each process to completely inform the other. In practice, jointly conducting a search over possible parses and possible word alignments would create a prohibitively large search space. To restrict the search space, we perform parsing and word alignment independently, then use the n -best or m -best output of each process as an approximation to the space of possible candidates explored by each process. We then consider all $n \times m$ combinations of parses and alignments from these two lists, and rerank these (parse, alignment) *pairs* for each sentence.

Since our ultimate goal is to improve translation quality, we would prefer to select the (parse, alignment) pair for each sentence that results in the highest BLEU score; in practice, the length of the experimental cycle required to train a syntax-based machine translation system on a large amount of data prevents us from optimizing BLEU score as an objective function during discriminative training. In lieu of BLEU score, we must rely upon intrinsic metrics of parse and alignment quality. The standard metric for parse quality is f-measure over labelled constituents; the standard metric for alignment quality is f-measure over alignment links. Neither parse f-measure nor alignment f-measure, however, directly measures the suitability of a (parse, alignment) pair for syntax-based machine translation.

To optimize the quality of the (parse, alignment) pair with respect to its use in a syntax-based translation, we maximize rule f-measure, which we compute by extracting a set of string-to-tree translation rules from the hypothesized (parse, alignment)

pair, and computing f-measure of that rule set against the rule set extracted from the gold parse and gold alignment.¹

4.1.1 Related Work

While the 1-best automatic parse or alignment for a particular source-target sentence pair may be sub-optimal for translation, a better parse or alignment often appears within the n -best space of parses or alignments explored by the automatic parser or aligner. By restricting a machine translation system to operating on the 1-best parse and alignment only, we risk excluding potentially superior parse and alignment candidates from consideration.

There has been considerable recent work in looking beyond the 1-best parses, word alignments, and even strings (for languages which require some pre-processing, such as Chinese, which must be segmented, or Arabic, which must be morphologically analyzed) when training a machine translation system. Recent work in this area is categorized in Table 4.1 according to whether the authors explore a 1-best, an n -best list, or a forest of candidates for parses, alignments, or strings in the source or target languages of the training data.

Venugopal et al. [100] widen a synchronous context-free grammar based machine translation pipeline by extracting string-to-tree translation rules in two different ways: first, from an m -best list of alignments and a 1-best parse; and second, from a 1-best alignment and an n -best list of parses. In each case, they define a probability distribution over the list of multiple parses or alignments, and weight each extracted rule by its fractional count according to this distribution. With these improved estimates of rule probabilities, they obtain an increase in BLEU score on a Chinese-English speech corpus translation task. They achieve greater gains in BLEU score from deepening

¹For more details on computing rule f-measure, please refer to Chapter 1, Section 1.2.9.

the size of the alignments m -best list than from deepening the size of the parse n -best list.

Mi and Huang [76] extract tree-to-string translation rules over a parse *forest* and 1-best alignments. Extracting rules from a forest is preferable to extracting them from an n -best list because a forest contains exponentially many parses, while an n -best list contains only those n parses which have been explicitly enumerated. Mi and Huang [76] achieve a significant improvement in BLEU score on a Chinese-English translation task.

Dyer et al. [15] extract phrase-based translation pairs not over a single source string f , but rather, over a *lattice* of n -best possible analyses of the source string. This approach is applicable to translation tasks where there is some ambiguity in the source string, either because it was produced by a speech recognition system or because it is written in a language that requires some pre-processing step such as segmentation (e.g. Chinese) or morphology (e.g. Arabic). Dyer et al. [15] obtain an improvement in BLEU score in both phrase-based and hierarchical phrase-based translation systems for both Chinese and Arabic.

Burkett and Klein [9] rerank an $n \times n$ list of *pairs* of (English parses, Chinese parses) by inducing a tree-node-to-tree-node alignment between the English and Chinese parse trees, and selecting the pair of parses that exhibits the highest degree of tree isomorphism. They obtain an increase in English parse f-measure, Chinese parse f-measure, and BLEU score on a Chinese-English translation task using a string-to-tree syntax-based system.

Liu et al. [68] extract phrase pairs from a *matrix*, rather than n -best list, of alignments. An alignment matrix is somewhat analogous to a parse forest in that a matrix represents exponentially many possible alignments in polynomial space, and therefore provides the potential to extract new alignments that have not been seen in any of the entries in the n -best list from which the matrix was constructed. By

Authors	Alignments	Parses		Strings	
		Source	Target	Source	Target
Venugopal et al. [100]	n -best	–	n -best	1-best	–
Mi and Huang [76]	1-best	forest	–	–	1-best
Dyer et al. [15]	1-best	–	–	n -best	–
Liu et al. [68]	matrix	–	–	1-best	1-best
Burkett and Klein [9]	1-best	n -best	n -best	–	–

Table 4.1: Survey of Recent Work in Exploring Multiple Alignments, Parses, or Strings in Training a Machine Translation System

taking advantage of shared substructures and local interactions among alignment links in this matrix, the authors can better estimate probabilities over alignments. They obtain an increase in BLEU score on several English-to-foreign and foreign-to-English phrase-based translation tasks.

With the exception of Venugopal et al. [100], none of the above methods considers *both* n -best parses and m -best alignments. Even Venugopal et al. [100] do not consider multiple alignments and multiple parses at the same time, due to prohibitive time and memory constraints; instead, they fix the size of the alignment m -best list to 1 while optimizing over n -best parses, then fix the size of the parse n -best list to 1 while optimizing over m -best alignments. In this chapter, we explore the space of n -best alignments and m -best parses *simultaneously*. We avoid the prohibitively large grammar problem that Venugopal et al. [100] face by propagating multiple alignments and multiple parses only long enough to determine which (parse, alignment) pair leads to the extraction of an optimal set of rules. After that point, we discard all other alignments and parses, resulting in a grammar size that is comparable to the baseline strategy of using the 1-best alignment and 1-best parse.

Another shortcoming of the above methods is that none of them directly optimizes the quality of the translation rules extracted from alignments or parses. In this chapter, we use a method for discriminatively reranking (parse, alignment) pairs that optimizes rule f-measure directly.

4.1.2 Our Work

We present an algorithm for selecting the best (parse, alignment) pair from an $n \times m$ -best list of (parse, alignment) pairs. We define the best (parse, alignment) pair to be the one whose extracted translation rules have the highest translation rule f-measure with respect to the gold standard translation rule set. For each sentence in a bilingual corpus, we discriminatively rerank a list of (parse, alignment) pairs using a handful of features designed to measure the quality of the parse, the quality of the alignment, and the degree to which the parse and the alignment are compatible with each other.

The primary contributions of this work are twofold: first, the use of joint features of the parse and alignment to rerank multiple parses and multiple alignments *simultaneously*; and second, the use of rule f-measure as an evaluation metric during discriminative training.

4.2 Methods

We create an n -best list of parses by running an implementation of the Collins parser (Model 3) [97] in n -best mode. We create an m -best list of alignments for each sentence pair in our data set as follows. Following a similar procedure to that of Venugopal et al. [100] [100], we first run GIZA++ in 50-best mode in the source-to-target and target-to-source directions; symmetrize the alignments using the *union* heuristic; and assign each alignment a probability

$$pr(a_{union}) = pr(a_{source-to-target}) \times pr(a_{target-to-source})$$

where $pr(a_{source-to-target})$ and $pr(a_{target-to-source})$ are given by GIZA++. Then, we select the top m -scoring symmetrized alignments from the list, after filtering out du-

plicate alignments. We score each (parse, alignment) pair using a linear combination of features h and their weights λ :

$$\text{score}(\text{parse}(e), A) = \lambda \cdot h(\text{parse}(e), A)$$

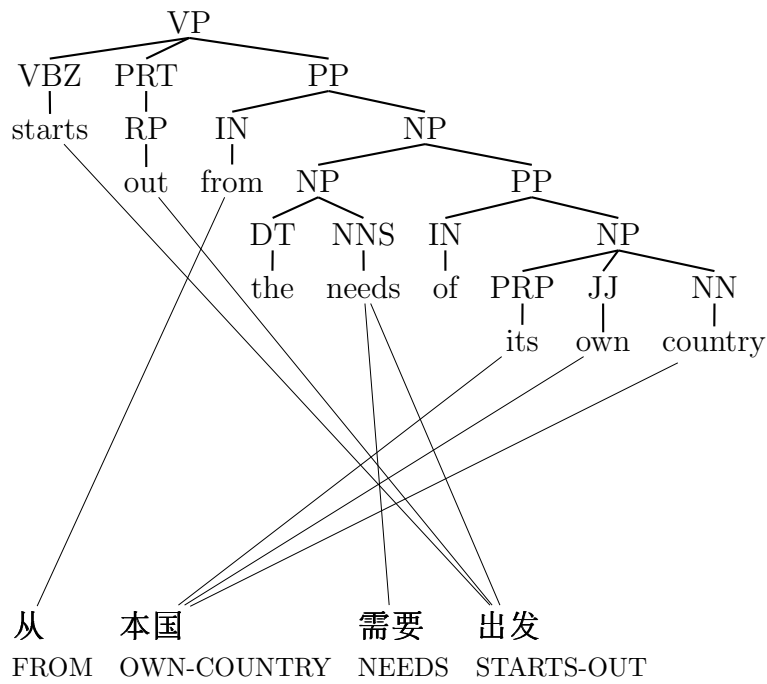
During training, we set the feature weights to maximize rule f-measure using a discriminative training set for which we have gold-standard parses and gold-standard alignments. During testing, we rerank an $n \times m$ - best list of (parse, alignment) pairs for each sentence.

4.2.1 Features

In this section, we describe the features h_i used to score each (parse, alignment) pair (p_i, a_i) .

Parse-Based Features

- **parserProb:** The probability assigned to a parse p_i by the original parser. The probabilities of all n parses in the n -best list are normalized to sum to one.
- **parseInverseRank:** The rank of a parse p_i in the n -best list is its position, i , in the n -best list. The inverse rank is $\frac{1}{i}$.
- **constituentProb:** This feature measures agreement at the constituent level among parses in the n -best list. The probability of each labelled constituent in the parse tree is computed by summing, for each constituent c_i , the probability $pr(p_j)$ assigned by the parser to parse p_j for all parses p_j in which constituent c_i occurs.



Extracted Rules:

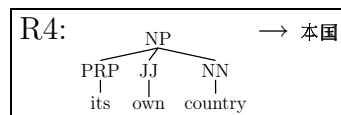
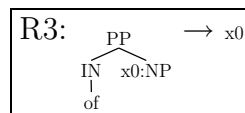
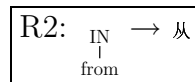
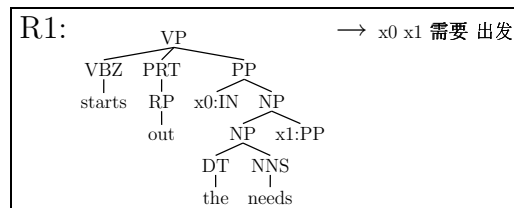


Figure 4.1: Example (Parse, Alignment) Pair and Extracted Rules

Alignment-Based Features

- **alignerProb:** The probability assigned to an alignment a_i by the original word aligner. The probabilities of all m alignments in the m -best list are normalized to sum to one.
- **alignmentInverseRank:** The rank of an alignment a_i in the m -best list is its position, i , in the m -best list. The inverse rank is $\frac{1}{i}$.
- **linkProb:** This feature measures agreement at the link level among alignments in the n -best list. The probability of each link l_i in the parse tree is computed by summing, for each constituent l_i , the probability $pr(a_j)$ assigned by the aligner to alignment a_j for all alignments a_j in which link l_i occurs.
- **unalignedForeignWords:** Total number of unaligned foreign words.
- **unalignedEnglishWords:** Total number of unaligned English words.
- **oneToMany Links:** We define a one-to-many alignment link to be a link between words e_i and f_j such that e_i , f_j , or both e_i and f_j are aligned to at least one other word in the sentence. In Figure 4.1, the links {本国-its, 本国-own, 本国-country} form a one-to-many alignment. We count the total number of links participating in such a one-to-many alignment.
- **nonMonotonicityJumpCount:** To measure non-monotonicity in an alignment, we sort all links i, j in ascending order first by their English positions, and second by their foreign positions. For each English word e_i aligned to foreign word f_j , if English word e_{i+1} is aligned to a foreign word f'_j where $j' \leq j$, then we define this to be an instance of non-monotonicity. In Figure 4.1, the English word “out” is aligned to a Chinese word 出发 at position 3 in the Chinese sentence, while the adjacent English word “from” is aligned to the Chinese word 从 at position 0 in the Chinese sentence. This represents a non-monotonic jump.

The feature **nonMonotonicityJumpCount** counts the number of times such a instance occurs in an alignment.

- **nonMonotonicityJumpSize:** With non-monotonicity defined as above, the feature **nonMonotonicityJumpSize** counts the magnitude of each non-monotonic jump in the alignment. In Figure 4.1, the non-monotonicity occurring from “out” to “from” has a jump size of 3, since there is a distance of 3 between 从 and 出发 in the Chinese sentence.
- **eOneToManyDiscontinuityCount:** A frequently occurring problem with GIZA++ Model 4 alignments is that a single English word is aligned to one or more foreign words that are not adjacent to each other. This type of discontinuity occurs rarely in the gold data and is often an indication of a spurious alignment link. In Figure 4.1, the one-to-many alignment 出发-”starts”, 出发-”out”, and 出发-”needs” exhibits a discontinuity between “out” and “needs”. The feature **eOneToManyDiscontinuityCount** counts the number of times that such a discontinuity occurs.
- **eOneToManyDiscontinuitySize:** With one-to-many discontinuities defined as above, the feature **eOneToManyDiscontinuitySize** counts the magnitude of such discontinuities (e.g. the magnitude of the gap between each non-adjacent foreign word aligned to the same English word). In Figure 4.1, the discontinuity in the one-to-many alignment 出发-”starts”, 出发-”out”, and 出发-”needs” has a size of 2, since there are 2 intervening words between “out” and “needs”.
- **fOneToManyDiscontinuityCount:** This feature is analogous to **eOneToManyDiscontinuityCount**, except computed in the reverse direction.
- **fOneToManyDiscontinuitySize:** This feature is analogous to **eOneToManyDiscontinuitySize**, except computed in the reverse direction.

- **conditionalLinkProbability:** We estimate the conditional probability that English word e_i is aligned to foreign word f_j , given that the words co-occur in the same sentence pair, from a large, automatically aligned bilingual corpus. We compute this conditional link probability for each link appearing in an alignment, and the value of the feature is then the average conditional link probability over all links in the alignment.
- **logLikelihoodRatio:** The log likelihood ratio of a pair of aligned words e_i, f_j measures the degree of association between those words than can be expected based on the probabilities of their co-occurrence in a bilingual sentence corpus. The log likelihood ratio of words e_i, f_j is defined as follows:

$$LLR(e_i, f_j) = c(e_i, f_j) \times \log \frac{pr(f_j|e_i)}{pr(f_j)} +$$

$$c(\neg e_i, f_j) \times \log \frac{pr(f_j|\neg e_i)}{pr(f_j)} +$$

$$c(e_i, \neg f_j) \times \log \frac{pr(\neg f_j|e_i)}{pr(\neg f_j)} +$$

$$c(\neg e_i, \neg f_j) \times \log \frac{pr(\neg f_j|\neg e_i)}{pr(\neg f_j)}$$

To compute the value of this feature for a given alignment, we sum the log likelihood ratio for all aligned word pairs (e_i, f_j) in that sentence pair.

Joint Parse- and Alignment-Based Features

- **ruleCount:** Total number of rules extracted from A , $parse(e)$, and f . In Figure 4.1, the value of **ruleCount** is 4.

- **maxRuleSize:** The size, measured in terms of total non-terminal nodes in the target parse tree, of the single largest rule extracted from A , $parse(e)$, and f . In Figure 4.1, the value of **maxRuleSize** is 9.
- **crossingForeignBrackets:** In monolingual parsing, the span of a node in a parse tree specifies the indices of words in the yield of the parse tree that are dominated by the subtree rooted at that node. A *crossing bracket* occurs when the span of a constituent in the hypothesized tree overlaps with the span of a constituent in the reference tree, where neither constituent dominates the other. Where bilingual word alignments are present, we can define the *foreign* span of a node in an English parse tree by first adjoining all foreign words as children of the English words to which they are aligned, and then computing the span in the *foreign* sentence for all nodes in the English parse tree. The number of cases where the foreign span of a node in the *English* parse tree overlaps with the foreign span of a sibling node in the English parse tree is the number of crossing foreign brackets. In Figure 4.1, there is a crossing bracket between the VBZ node dominating “starts” and the PRT node dominating “out”, since the foreign span of “starts” is 3-4 and the foreign span of “out” is also 3-4, so these two nodes are siblings with overlapping foreign spans. There is another crossing bracket between the PP headed by “from” and the PRT node, since the foreign span of the PP is 0-4, which overlaps with the foreign span of the PRT node, 3-4.
- **reorderings:** After adjoining each foreign word to the English leaf node(s) to which the foreign word is aligned and annotating each node in the English tree with its span in the foreign string, we observe that there may be nodes in the English tree whose foreign spans do not *overlap*, but rather, are *reordered* with respect to the English ordering. Certain reordering patterns appear regularly

in the gold standard data; other reordering patterns may be irregular and may indicate that the parse and alignment are not compatible. In Figure 4.1, there is a reordering between the children of the NP dominating “the needs of its own country”: the foreign span of the NP dominating “the needs” is 2-4, while the foreign span of the PP dominating “of its own country” is 1-2.

4.2.2 Discriminative Training

Averaged Perceptron

We set the feature weights λ using a modified version of averaged perceptron learning with structured outputs [19]. We initialize all feature weights uniformly to 1. During each pass over the discriminative training set, we score each (parse, alignment) pair in the $n \times m$ -best list of candidates for that sentence according to the current weight settings. We select the highest-scoring (parse, alignment) pair as our model hypothesis. In general, the gold (parse, alignment) pair is not a member of the $n \times m$ -best list of candidates; following Collins [18] and Charniak [11] for parse reranking and Liang et al. [64] for translation reranking, we define the oracle (parse, alignment) pair as the element in the list of candidates that is most *similar* to the gold (parse, alignment) pair. We use rule f-measure as a similarity metric. We update each feature weight λ_i towards the oracle as follows: $\lambda_i = \lambda_i + h_i^{(\text{parse}(e), A)_{\text{oracle}}} - h_i^{(\text{parse}(e), A)_{\text{predicted}}}$.

Following Moore [77], after each training pass, we average all feature weight vectors seen during that pass, and decode the training set using the vector of averaged feature weights. When alignment quality stops increasing on the training set, perceptron training ends. The weight vector returned by perceptron training is the average over the training set of all weight vectors seen during all iterations [19].

4.2.3 Linear Regression

As an alternative to discriminative training, we also use linear regression to set our feature weights.

4.3 Experimental Setup

We evaluate the effect of (parse, alignment) pair reranking upon alignment quality; parse quality; and rule quality for an Arabic-English newswire data set. The data set (Table 4.2) contains a small subset of 2248 manually parsed and manually word-aligned sentence pairs. We divide the manually parsed and word-aligned subset into a training set (used to discriminatively set the feature weights for (parse, alignment) pair reranking) and a test set (used to evaluate the impact of reranking upon rule f-measure). Using the feature weights learned on the manually parsed, manually aligned training set, we then rerank (parse, alignment) pairs for each sentence in the remainder (non-manually-annotated) of the bilingual data set.

Train	Test
LDC2009E82[63] excerpt (2008 sent.)	LDC2009E82[63] excerpt (240 sent.)

Table 4.2: Data Sets Used in (Parse, Alignment) Pair Reranking

To evaluate the impact of link deletion upon rule quality, we compare the rule f-measure of the rule set extracted from our hypothesized alignments and a Collins-style parser against the rule set extracted from gold alignments and gold parses.

4.4 Results

Table 4.3 illustrates the results of oracle reranking experiments, in which we select the (parse, alignment) pair with the highest rule f-measure from our $n \times m$ -best list

for varying values of n and m . The baseline strategy is to select the 1-best parse and 1-best alignment, which achieves a rule f-measure of 39.0 on our 240-sentence test set. Increasing parse n -best size to 10 and alignment m -best size to 100 results in an oracle rule f-measure of 44.9 (or +5.9 points relative to the baseline), but increasing n to 50 and m to 100 produces an oracle rule f-measure of 47.0 (an additional 2.1 points). Further increasing parse n -best size to 50 and alignment m -best size to 1000 results improves oracle rule f-measure only slightly, to 47.7 (an additional +0.7 points). Because little gain in rule f-measure is obtained by increasing n x m -best list sizes beyond 10 parses and 100 alignments, we use an n -best size of 10 and an m -best size of 100 in our experiments.

Method	Parse n -best size	Alignment m -best size	Rule F-Measure	
			Training	Test
Baseline	1	1	37.1	39.0
Oracle Reranking	10	10	42.2	44.9
Oracle Reranking	10	100	45.0	47.0
Oracle Reranking	50	1000	45.9	47.7

Table 4.3: Results of Oracle (Parse, Alignment) Pair Reranking Experiments on 2248 Sentences of Arabic-English Parallel Data. Baseline strategy is to select the 1-best alignment produced by the automatic aligner (GIZA++ union) and the 1-best parse produced by the automatic parser. Oracle criterion is rule f-measure.

We divide our 2248-sentence data set into a training set (2008 sentences) and a held-out test set (240 sentences), and report rule f-measure on each data set after reranking. Table 4.4 illustrates our results. Setting feature weights via discriminative training, we observe an improvement in rule f-measure of +1.8 points on the training set and +1.4 points on the held-out test set. Setting feature weights via linear regression, we observe an improvement in rule f-measure of +1.4 points on the training set and +1.2 points on the held-out test set.

Method	Parse n -best size	Alignment m -best size	Rule F-Measure	
			Training	Test
Baseline	1	1	37.1	39.0
Perceptron	10	100	38.9	40.4
Linear Regression	10	100	38.5	40.2
Oracle Reranking	10	100	45.0	47.0

Table 4.4: Results of (Parse, Alignment) Pair Reranking with 100-best Alignments and 10-best Parses. Objective Function is Rule F-Measure. Training set is 2009 sentences and test set is a held-out set of 239 sentences of Arabic-English parallel data. Baseline strategy is to select the 1-best alignment produced by the automatic aligner (GIZA++ union) and the 1-best parse produced by the automatic parser. Oracle criterion is rule f-measure.

4.5 Summary of Contributions

In this chapter, we have presented a method for discriminatively reranking a list of n -best alignments and m -best parses in order to select the best (parse, alignment) pair. We achieve a modest gain in rule f-measure of +1.4 points relative to the baseline on a held-out test set. The primary contributions of this work are the following:

- The use of joint features of the parse and the alignment to rerank (parse, alignment) pairs
- The use of rule f-measure as an objective function to optimize during discriminative training

Chapter 5

Conclusion

Parsing and word alignment are crucial stages of the syntax-based machine translation pipeline. There are two problems with the way that parsing and word alignment are typically performed in state-of-the-art syntax-based MT systems.

The first problem is that automatic word alignments and automatic parse trees are prone to errors; these errors then cause the system to extract potentially incorrect rules of translational correspondence. In some cases, these errors may cause the system to infer a translational equivalence that simply does not exist between the two languages. In other cases, the effect is more subtle: these errors may force the system to extract rules that are not incorrect, but which fail to identify *minimal* units of translational correspondence. These rules incorporate unnecessarily large amounts of context, and thus have limited applicability to unseen sentences at test time.

The second problem with current approaches to word alignment and parsing for syntax-based statistical machine translation systems is that word alignment and parsing are performed in isolation of each other. Since each process produces constraints that can potentially be used to guide the other, we can expect to improve the accuracy of each process by integrating them more closely.

In this thesis, we have examined ways to improve the accuracy of both parsing and word alignment for syntax-based MT by integrating these processes where possible. The primary contributions of this thesis are as follows:

- We have presented a novel method for combining the output of multiple parsers by converting each parse into a set of context-free productions, then recombining those context-free productions to form a potentially new parse tree. While an existing method that recombines *constituents* achieves the highest parse accuracy, our method of recombining *context-free productions* results in better translation accuracy in a downstream string-to-tree syntax-based machine translation system [28].
- We have shown that automatically produced bilingual Chinese-English word alignments can be used to resolve syntactic ambiguity in English [27].
- We have presented a novel algorithm that uses automatically produced parses to improve the precision of word alignments by identifying and deleting alignment links that violate syntactic constraints. We have demonstrated that these more highly precise alignments result in a significant gain in machine translation accuracy in syntax-based translation [29].
- We have presented a method for reranking parse, alignment *pairs* that allows parses and alignments to inform and constrain each other simultaneously.

5.1 Future Work

5.1.1 Using Bilingual Word Alignments to Improve Parsing Accuracy

In our work on using bilingual Chinese-English word alignments to resolve syntactic ambiguity in English, we limit our investigations to a particular type of common syntactic ambiguity in English (namely, PP-attachment). In order to make a sufficiently large impact upon parse quality to affect the performance of a downstream syntax-based machine translation system, we would need to extend this approach beyond PP-attachment to handle other types of syntactic ambiguity as well.

5.1.2 Using Parsing to Improve Word Alignment Accuracy

In our work on using features of the extracted syntax-based translation rules to identify and delete incorrect links from an existing word alignment, we find that such features are effective in both increasing alignment quality and increasing translation quality of a downstream syntax-based translation system. Our results demonstrate that these constraints are useful when trying to improve upon an existing alignment with high recall and low precision (such as GIZA++ union) [85]. However, we suspect that our features are not as useful when trying to improve upon an existing alignment with balanced recall and precision (such as GIZA++ refined) [85], and therefore the ability of our algorithm to improve upon an existing alignment may depend upon the precision/recall balance of the existing alignment.

5.1.3 Using Parsing and Word Alignment to Improve Accuracy of Both Processes Simultaneously

When trying to improve upon an existing alignment with balanced recall and precision (such as GIZA++ refined), it turns out that a set of syntactic constraints based on features of the extracted rules that is similar to that used in Chapter 3 is *not* very effective.

In general, we find that it is easier to improve upon the quality of the baseline 1-bests alignments or parses by generating new, potentially better alignments or parses (such as in parse combination or alignment link deletion) than by reranking from an n -best list of alignments or parses.

We also find that it is easier to improve upon alignment quality by reranking from an n -best list than it is to improve upon parsing quality. State-of-the-art parsing accuracy is higher, when measured in parse f-measure, than state-of-the-art alignment accuracy, when measured in alignment f-measure. In addition, the quality of each entry in an n -best list seems to drop off far more quickly as n increase for a state-of-the-art parser than for a state-of-the-art aligner.

In light of these observations, we believe that in lieu of *reranking* an n -best list, a more fruitful approach would be *recombining* parses and alignments to allow new, potentially unseen combinations of parses and alignments to be explored.

5.1.4 A New Evaluation Metric: Rule F-Measure

While rule f-measure seems to be a step in the right direction in the sense that it is a more direct measure of the impact of an alignment or parse upon translation quality than either alignment f-measure or parse f-measure, we did not observe a clear correlation between rule f-measure and BLEU score. Moreover, in its current, most basic formulation, rule f-measure is difficult to use as an objective criterion in discriminative

training because it increases in a step-like manner as the quality of alignments and parses increases. In comparison, alignment f-measure and parse f-measure increase more smoothly as the quality of alignments or parses increases. Perhaps a less strict version of rule f-measure (such as one that rewards partial matching on rules, or one that collapses fine-grained syntactic labels into a more coarse-grained set of labels to increase the number of matches) would prove both easier to optimize during discriminative training because of its greater sensitivity to small differences in alignments and parses, and might potentially correlate more closely with BLEU score.

Appendix

Appendix A

Data Sets

Language	Name	Sentence Pairs	Source
ara-eng	train (small)	170,863	LDC2005E46[50] LDC2004T17[71] LDC2006E93[62] LDC2006E92[61] LDC2006E34[57] LDC2006E24[54] LDC2006E86[59] LDC2006E85[55] LDC2006E25[53]
ara-eng	train (large)	6,561,091	UN LDC2007E07[41] LDC2006E25[53] LDC2004T18[96] LDC2005E46[50] LDC2004T17[71] LDC2006E93[62] LDC2006E92[61] LDC2006E34[57] LDC2006E24[54] LDC2006E86[59] LDC2006E85[55]
ara-eng	tune newswire	1,178	NIST 04[81] NIST 05[82] NIST 06[83]
ara-eng	test 1 newswire	1,298	NIST 04[81] NIST 05[82] NIST 06[83]
ara-eng	test 2 newswire	765	NIST 06[83] newswire

Table A.1: Arabic-English Data Sets

Language	Name	Sentence Pairs	Source
chi-eng	A train (small)	329,031	LDC2003E07[48] LDC2003E14[49] LDC2005T06[70] LDC2006G05[30]
chi-eng	A tune newswire	878	NIST 02[79]
chi-eng	A test newswire	919	NIST 03[80]
chi-eng	B train (small)	395,055	LDC2003E07[48] LDC2003E14[49] LDC2005E83[52] LDC2005T06[70] LDC2006E24[54] LDC2006E34[57] LDC2006E85[55] LDC2006E86[59] LDC2006E92[61] LDC2006E93[62]
chi-eng	B tune newswire	1,482	NIST 04[81] NIST 05[82] NIST 06[83]
chi-eng	B test newswire	1,463	NIST 04[81] NIST 05[82] NIST 06[83]

Table A.2: Chinese-English Data Sets

Bibliography

Bibliography

- [1] Steven Abney, Robert E. Schapire, and Yoram Singer. Boosting applied to tagging and PP attachment. In *Proceedings of EMNLP*, 1999.
- [2] Necip Fazil Ayan and Bonnie J. Dorr. Going beyond AER: an extensive analysis of word alignments and their impact on MT. In *Proceedings of ACL*, 2006.
- [3] Ann Bies, Martha Palmer, Justin Mott, and Colin Warner. English Chinese translation treebank v 1.0. *Linguistic Data Consortium, Catalog Number LDC2007T02*, 2007.
- [4] Daniel M. Bikel. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of HLT*, 2002.
- [5] E. Black, S. Abney, D. Flickinger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. A procedure for quantitatively comparing the syntactic coverage of english grammars. In *Proceedings of the DARPA Speech and Natural Language Workshop*, 1991.
- [6] Eric Brill and Philip Resnik. A rule-based approach to prepositional phrase attachment disambiguation. In *Proceedings of COLING*, 1994.
- [7] Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2), 1990.
- [8] Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2), 1993.
- [9] David Burkett and Dan Klein. Two languages are better than one (for syntactic parsing). In *Proceedings of EMNLP*, 2008.
- [10] Eugene Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of AAAI*, 1997.

- [11] Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, 2005.
- [12] Colin Cherry and Dekang Lin. Soft syntactic constraints for word alignment through discriminative training. In *Proceedings of ACL (Poster)*, 2006.
- [13] David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*, 2005.
- [14] David Chiang. Hierarchical phrase-based translation. In *Computational Linguistics*, 2007.
- [15] Smaranda Muresan Christopher Dyer and Philip Resnik. Generalizing word lattice translation. In *Proceedings of ACL*, 2008.
- [16] John Cocke and Jacob T. Schwartz. Programming languages and their compilers: Preliminary notes. In *Technical report, Courant Institute of Mathematical Sciences, New York University*, 1970.
- [17] Michael Collins. Three generative, lexicalised models for statistical parsing. In *Proceedings of ACL*, 1997.
- [18] Michael Collins. Discriminative reranking for natural language parsing. In *Proceedings of ICML*, 2000.
- [19] Michael Collins. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*, 2002.
- [20] Michael Collins and James Brooks. Prepositional phrase attachment through a backed-off model. In *Proceedings of the Workshop on Very Large Corpora*, 1995.
- [21] Ido Dagan and Alon Itai. Word sense disambiguation using a second language monolingual corpus. *Computational Linguistics*, 20(4), 1994.
- [22] Ido Dagan, Alon Itai, and Ulrike Schwall. Two languages are more informative than one. In *Proceedings of ACL*, 1991.
- [23] AP Dempster, NM Laird, and DB Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1), 1977.
- [24] John DeNero, David Chiang, and Kevin Knight. Fast consensus decoding over translation forests. In *Proceedings of ACL*, 2009.
- [25] John DeNero and Dan Klein. Tailoring word alignments to syntactic machine translation. In *Proceedings of ACL*, 2007.

- [26] Yonggang Deng and William Byrne. HMM word and phrase alignment for statistical machine translation. In *Proceedings of HLT/EMNLP*, 2005.
- [27] Victoria Fossum and Kevin Knight. Using bilingual Chinese-English word alignments to resolve PP-attachment ambiguity in English. In *Proceedings of the AMTA Student Research Workshop*, 2008.
- [28] Victoria Fossum and Kevin Knight. Combining constituent parsers. In *Proceedings of HLT-NAACL*, 2009.
- [29] Victoria Fossum, Kevin Knight, and Steven Abney. Using syntax to improve word alignment for syntax-based statistical machine translation. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, 2008.
- [30] FOUO. Gale Y1 Q2 release–LDC/FBIS/NVTC parallel text v2.0. *Linguistic Data Consortium, Catalog Number LDC2006G05*, 2006.
- [31] Alexander Fraser and Daniel Marcu. Getting the structure right for word alignment: LEAF. In *Proceedings of EMNLP*, 2007.
- [32] Alexander Fraser and Daniel Marcu. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3), 2007.
- [33] Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of ACL*, 2006.
- [34] Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What’s in a translation rule? In *Proceedings of HLT/NAACL-04*, 2004.
- [35] John C. Henderson and Eric Brill. Exploiting diversity in natural language processing: Combining parsers. In *Proceedings of EMNLP*, 2000.
- [36] Donald Hindle and Mats Rooth. Structural ambiguity and lexical relations. In *Proceedings of ACL*, 1993.
- [37] Liang Huang and David Chiang. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of ACL*, 2007.
- [38] Liang Huang, Kevin Knight, and Aravind Joshi. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, 2006.
- [39] Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11(3), 2005.
- [40] Rebecca Hwa, Philip Resnik, Amy Weinberg, and Okan Kolak. Evaluating translational correspondence using annotation projection. In *Proceedings of ACL*, 2001.

- [41] ISI. ISI Arabic-English automatically extracted parallel text. *Linguistic Data Consortium, Catalog Number LDC2007E07*, 2007.
- [42] Abraham Ittycheriah and Salim Roukos. A maximum entropy word aligner for Arabic-English machine translation. In *Proceedings of HLT/EMNLP*, 2005.
- [43] T. Kasami. An efficient recognition and syntax-analysis algorithm for context-free languages. In *Scientific report AFCRL-65-758, Air Force Cambridge Research Lab, Bedford, MA*, 1965.
- [44] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of ACL*, 2003.
- [45] 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 *Proceedings of ACL (demo)*, 2007.
- [46] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of HLT/NAACL*, 2003.
- [47] Simon Lacoste-Julien, Ben Taskar, Dan Klein, and Michael I. Jordan. Word alignment via quadratic assignment. In *Proceedings of HLT-NAACL*, 2006.
- [48] LDC. Chinese treebank English parallel corpus. *Linguistic Data Consortium, Catalog Number LDC2003E07*, 2003.
- [49] LDC. FBIS multilanguage texts. *Linguistic Data Consortium, Catalog Number LDC2003E14*, 2003.
- [50] LDC. Arabic treebank English translation. *Linguistic Data Consortium, Catalog Number LDC2005E46*, 2005.
- [51] LDC. Gale Y1 Q1 release–English translation treebank. *Linguistic Data Consortium, Catalog Number LDC2005E85*, 2005.
- [52] LDC. Gale Y1 Q1 release–translations. *Linguistic Data Consortium, Catalog Number LDC2005E83*, 2005.
- [53] LDC. Gale Y1–Arabic English parallel news text. *Linguistic Data Consortium, Catalog Number LDC2006E25*, 2006.
- [54] LDC. Gale Y1–interim release: Translations. *Linguistic Data Consortium, Catalog Number LDC2006E24*, 2006.
- [55] LDC. Gale Y1 Q1 release–English translation treebank. *Linguistic Data Consortium, Catalog Number LDC2006E85*, 2006.

- [56] LDC. Gale Y1 Q2 release–English translation treebank, phase 1. *Linguistic Data Consortium, Catalog Number LDC2006E36*, 2006.
- [57] LDC. Gale Y1 Q2 release–translations v2.0. *Linguistic Data Consortium, Catalog Number LDC2006E34*, 2006.
- [58] LDC. Gale Y1 Q3 release–English translation treebank, phase 1. *Linguistic Data Consortium, Catalog Number LDC2006E82*, 2006.
- [59] LDC. Gale Y1 Q3 release–word alignment. *Linguistic Data Consortium, Catalog Number LDC2006E86*, 2006.
- [60] LDC. Gale Y1 Q4 release–English translation treebank, phase 1. *Linguistic Data Consortium, Catalog Number LDC2006E95*, 2006.
- [61] LDC. Gale Y1 Q4 release–translations. *Linguistic Data Consortium, Catalog Number LDC2006E92*, 2006.
- [62] LDC. Gale Y1 Q4 release–word alignment. *Linguistic Data Consortium, Catalog Number LDC2006E93*, 2006.
- [63] LDC. Gale phase 4 Arabic parallel aligned treebank part 1. *Linguistic Data Consortium, Catalog Number LDC2009E82*, 2009.
- [64] Percy Liang, Alexandre Bouchard-Cote, Dan Klein, and Ben Taskar. An end-to-end discriminative approach to machine translation. In *Proceedings of COLING/ACL*, 2006.
- [65] Yang Liu, Qun Liu, and Shouxun Lin. Log-linear models for word alignment. In *Proceedings of ACL*, 2005.
- [66] Yang Liu, Qun Liu, and Shouxun Lin. Tree-to-string alignment template for statistical machine translation. In *Proceedings of ACL*, 2006.
- [67] Yang Liu, Yajuan Lü, and Qun Liu. Improving tree-to-tree translation with packed forests. In *Proceedings of ACL*, 2009.
- [68] Yang Liu, Tian Xia, Xinyan Xiao, and Qun Liu. Weighted alignment matrices for statistical machine translation. In *Proceedings of EMNLP*, 2009.
- [69] Adam Lopez and Philip Resnik. Improved hmm alignment models for languages with scarce resources. In *Proceedings of the ACL Workshop on Parallel Text*, 2005.
- [70] Xiaoyi Ma. Chinese news translation text part 1. *Linguistic Data Consortium, Catalog Number LDC2005T06*, 2005.
- [71] Xiaoyi Ma, Dalal Zakhary, and Moussa Bamba. Arabic news translation text part 1. *Linguistic Data Consortium, Catalog Number LDC2004T17*, 2004.

- [72] Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*, 19(2), 1993.
- [73] Jonathan May and Kevin Knight. Tiburon: A weighted tree automata toolkit. In *Proceedings of CIAA 2006*, volume 4094 of *Lecture Notes in Computer Science*, pages 102–113, 2006.
- [74] Jonathan May and Kevin Knight. Syntactic re-alignment models for machine translation. In *Proceedings of EMNLP-CoNLL*, 2007.
- [75] I. Dan Melamed. Statistical machine translation by parsing. In *Proceedings of ACL*, 2004.
- [76] Haitao Mi and Liang Huang. Forest-based translation rule extraction. In *Proceedings of EMNLP*, 2008.
- [77] Robert C. Moore. A discriminative framework for bilingual word alignment. In *Proceedings of HLT/EMNLP*, 2005.
- [78] Robert C. Moore, Wen tau Yih, and Andreas Bode. Improved discriminative bilingual word alignment. In *Proceedings of ACL*, 2006.
- [79] NIST. MT evaluation data 2002. <http://www.itl.nist.gov/iad/mig/tests/mt/2002/>, 2002.
- [80] NIST. MT evaluation data 2003. <http://www.itl.nist.gov/iad/mig/tests/mt/2003/>, 2003.
- [81] NIST. MT evaluation data 2004. <http://www.itl.nist.gov/iad/mig/tests/mt/2004/>, 2004.
- [82] NIST. MT evaluation data 2005. <http://www.itl.nist.gov/iad/mig/tests/mt/2005/>, 2005.
- [83] NIST. MT evaluation data 2006. <http://www.itl.nist.gov/iad/mig/tests/mt/2006/>, 2006.
- [84] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, 2003.
- [85] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1), 2003.
- [86] Franz Josef Och and Hermann Ney. The alignment template approach to statistical machine translation. In *Computational Linguistics*, 2004.
- [87] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, 2002.

- [88] Slav Petrov and Dan Klein. Improved inference for unlexicalized parsing. In *Proceedings of HLT-NAACL*, 2007.
- [89] Chris Quirk and Simon Corston-Oliver. The impact of parse quality on syntactically-informed statistical machine translation. In *Proceedings of EMNLP*, 2006.
- [90] Chris Quirk, Arul Menezes, and Colin Cherry. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of ACL*, 2005.
- [91] Adwait Ratnaparkhi. Statistical models for unsupervised prepositional phrase attachment. In *Proceedings of ACL*, 1998.
- [92] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(386), 1958.
- [93] Antti-Veikko I. Rosti, Necip Fazil Ayan, Bing Xiang, Spyridon Matsoukas, Richard M. Schwartz, and Bonnie J. Dorr. Combining outputs from multiple machine translation systems. In *Proceedings of HLT-NAACL*, 2007.
- [94] Kenji Sagae and Alon Lavie. Parser combination by reparsing. In *Proceedings of HLT-NAACL*, 2006.
- [95] Lee Schwartz, Takako Aikawa, and Chris Quirk. Disambiguation of English PP attachment using multilingual data. In *Proceedings of MT Summit IX*, 2003.
- [96] Several. Arabic English parallel news part 1. *Linguistic Data Consortium, Catalog Number LDC2004T18*, 2004.
- [97] Radu Soricut. A reimplementation of collins' parsing models. In *Technical report, Information Sciences Institute, Department of Computer Science, University of Southern California*, 2004.
- [98] Ben Taskar, Simon Lacoste-Julien, and Dan Klein. A discriminative matching approach to word alignment. In *Proceedings of HTL/EMNLP*, 2005.
- [99] Hans van Halteren, Jakub Zavrel, and Walter Daelemans. Improving data driven wordclass tagging by system combination. In *Proceedings of ACL-ICCL*, 1998.
- [100] Ashish Venugopal, Andreas Zollmann, Noah A. Smith, and Stephan Vogel. Wider pipelines: N-best alignments and parses in mt training. In *Proceedings of AMTA*, 2008.
- [101] Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based word alignment in statistical translation. In *Proceedings of COLING*, 1996.
- [102] Wei Wang, Kevin Knight, and Daniel Marcu. Binarizing syntax trees to improve syntax-based machine translation accuracy. In *Proceedings of ACL*, 2007.

- [103] Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403, 1997.
- [104] Kenji Yamada and Kevin Knight. A decoder for syntax-based statistical mt. In *Proceedings of ACL*, 2001.
- [105] David Yarowsky and Grace Ngai. Inducing multilingual POS taggers and np bracketers via robust projection across aligned corpora. In *Proceedings of NAACL*, 2001.
- [106] Daniel H. Younger. Recognition and parsing of context-free languages in time n^3 . *Information and Control*, 10(2), 1967.
- [107] Hao Zhang, Liang Huang, Daniel Gildea, and Kevin Knight. Synchronous binarization for machine translation. In *Proceedings HLT-NAACL*, 2006.