

# **Improving Statistical Machine Translation with Linguistic Information**

*Hieu Hoang*

Doctor of Philosophy  
Institute for Communicating and Collaborative Systems  
School of Informatics  
University of Edinburgh  
2013



# Abstract

Statistical machine translation (SMT) should benefit from linguistic information to improve performance but current state-of-the-art models rely purely on data-driven models.

There are several reasons why prior efforts to build linguistically annotated models have failed or not even been attempted. Firstly, the practical implementation often requires too much work to be cost effective. Where ad-hoc implementations have been created, they impose too strict constraints to be of general use. Lastly, many linguistically-motivated approaches are language dependent, tackling peculiarities in certain languages that do not apply to other languages.

This thesis successfully integrates linguistic information about part-of-speech tags, lemmas and phrase structure to improve MT quality.

The major contributions of this thesis are:

1. We enhance the phrase-based model to incorporate linguistic information as additional factors in the word representation. The *factored phrase-based model* allows us to make use of different types of linguistic information in a systematic way within the predefined framework. We show how this model improves translation by as much as 0.9 BLEU for small German-English training corpora, and 0.2 BLEU for larger corpora.
2. We extend the factored model to the *factored template model* to focus on improving reordering. We show that by generalising translation with part-of-speech tags, we can improve performance by as much as 1.1 BLEU on a small French-English system.
3. Finally, we switch from the phrase-based model to a syntax-based model with the *mixed syntax model*. This allows us to transition from the word-level approaches using factors to multiword linguistic information such as syntactic labels and shallow tags. The mixed syntax model uses source language syntactic information to inform translation. We show that the model is able to explain translation better, leading to a 0.8 BLEU improvement over the baseline hierarchical phrase-based model for a small German-English task. Also, the model requires only labels on continuous source spans, it is not dependent on a tree structure, therefore, other types of syntactic information can be integrated into the model. We experimented with a shallow parser and see a gain of 0.5 BLEU for the same dataset. Training with more training data, we improve translation by 0.6 BLEU (1.3 BLEU out-of-domain) over the hierarchical baseline.

During the development of these three models, we discover that attempting to rigidly model translation as linguistic transfer process results in degraded performance. However, by combining the advantages of standard SMT models with linguistically-motivated models, we are able to achieve better translation performance. Our work shows the importance of balancing the specificity of linguistic information with the robustness of simpler models.

# Acknowledgements

Biggest thanks must go to Philipp Koehn for unwavering support throughout the course of the Phd. This thesis wouldn't have started or finished without you.

Having a large group of fellow SMTers at the University of Edinburgh has certainly helped me understand the issues in machine translation and NLP. So thank you to Miles Osborne for the insights and curve balls, Barry Haddow and Adam Lopez for piling through drafts of this thesis. Abhishek Arun, Loïc Dugast, Abby Levenberg, Lexi Birch for fun times in the office and beyond.

The Moses project was initially just my thesis code but it has now taken on a life of its own. Credit must go to the CLSP workshop participants who turned a buggy, underperforming decoder into something useful. These people made it happen: Chris Callison-Burch, Nicola Bertoldi, Ondrej Bojar, Alexandra Constantin, Brooke Cowan, Chris Dyer, Marcello Federico, Evan Herbst, Christine Moran, Wade Shen, Philipp Koehn, and Richard Zens.

Thanks to all the people how have contributed code and time to the Moses project, answered questions on the mailing list or asked picky questions. You have made the project one of the most open, inclusive projects in NLP. Being involved has been an honour and a privilege.

I managed to sneak off to Valencia once when the Scottish winter became too much. Thank you to Francisco Casacuberta for hosting me during my stay. Also Daniel Ortíz, Alicia Pérez, Jesús, Raúl, and others for sun and beers.

A Phd is a long, arduous road to take. These are some of the people who helped me stay awake and sane during the journey. Ivan, Sebastian, Martin, Michael, Gregor, Tim, Neil, Carsten, Amittai, Markus, Volker, Paul for hanging out. Phil and Erich for breaking your stuff playing football in the office! Vera for the skiing and introducing me to the Canoe Club. The Canoe Club for showing me wet, wild, cold, beautiful Scotland. My big bad van for getting me there. Housemates Hugh, Diane and Ondrej for being homely & housematey. My mates in London for the drinks (Jonno, Jolene, Liam, Daniel, Erika, Mitch, Carl, Amit, Tom, Natasha). My mum, my dad, brothers Uy and Andrew, Di Mi, Cau Tau, Bà.

Last but not least, I would like to thank the people who generously funded my studies: EPSRC, the European Union, DARPA.

# Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

*(Hieu Hoang)*

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Machine Translation Approaches . . . . .	1
1.2	Statistical Machine Translation . . . . .	2
1.3	Why use Linguistic Information in SMT? . . . . .	4
1.4	Thesis Outline and Contribution . . . . .	5
1.5	Experimental Setup . . . . .	6
1.6	Related publications . . . . .	8
<b>2</b>	<b>Factored Translation</b>	<b>9</b>
2.1	Model . . . . .	10
2.1.1	Phrase-Based Model . . . . .	10
2.1.2	Factored Model . . . . .	12
2.1.3	Factor Sequence Models . . . . .	15
2.1.4	Examples . . . . .	16
2.2	Decoding . . . . .	19
2.2.1	Phrase-Based Decoding . . . . .	19
2.2.2	Translation Option . . . . .	19
2.2.3	Hypotheses . . . . .	20
2.2.4	Beam Search . . . . .	20
2.2.5	Hypothesis Recombination . . . . .	22
2.2.6	Pruning . . . . .	22
2.3	Factored Model Decoding . . . . .	24
2.3.1	Construction of Translation Options . . . . .	24
2.3.2	Over-Creation of Factored Translation Options . . . . .	28
2.4	Experiments . . . . .	28
2.4.1	Source Factors . . . . .	29
2.4.2	Generating Target Factors . . . . .	33

2.4.3	Using Source and Target Factors . . . . .	35
2.4.4	N-Gram Sequence Models on Factors . . . . .	36
2.4.5	Reducing Sparsity . . . . .	36
2.4.6	Intermediate Translation Option Pruning . . . . .	40
2.4.7	Out-of-Vocabulary Words . . . . .	43
2.5	Large Training Corpora . . . . .	45
2.6	Conclusion . . . . .	46
<b>3</b>	<b>Factored Template Model</b>	<b>49</b>
3.1	Reordering in Phrase-Based Models . . . . .	51
3.1.1	POS-Based Reordering . . . . .	52
3.2	Translation Using Templates of Factors . . . . .	54
3.2.1	Factored Template Model . . . . .	56
3.2.2	Constraints . . . . .	57
3.2.3	Example of Constraints . . . . .	59
3.3	Decoding . . . . .	61
3.3.1	Creating Translation Options . . . . .	62
3.4	Training . . . . .	63
3.5	Experiments . . . . .	63
3.5.1	German-English Results . . . . .	64
3.5.2	French-English Results . . . . .	65
3.5.3	Maximum Size of Templates . . . . .	66
3.5.4	Lexicalized Reordering Models . . . . .	66
3.6	Analysis . . . . .	67
3.6.1	Use of Factored Template Model during Decoding . . . . .	67
3.6.2	POS Sequences Used . . . . .	69
3.6.3	Manual Evaluation . . . . .	72
3.6.4	Larger training corpora . . . . .	73
3.7	Conclusion . . . . .	73
<b>4</b>	<b>Mixed-Syntax Translation</b>	<b>75</b>
4.1	Past Work . . . . .	76
4.1.1	Overview . . . . .	76
4.1.2	Synchronous Tree-Substitution Grammar . . . . .	78
4.1.3	Syntax Models . . . . .	82
4.1.4	String-to-Tree Models . . . . .	83



4.1.5	Tree-to-String Models . . . . .	85
4.1.6	Tree-to-Tree Models . . . . .	86
4.1.7	Summary of Past Work . . . . .	86
4.2	Model . . . . .	88
4.2.1	Preamble . . . . .	88
4.2.2	CKY for Parsing . . . . .	89
4.2.3	CKY+ for Parsing . . . . .	89
4.2.4	CKY+ for Hierarchical Phrase-Based Model . . . . .	90
4.2.5	Syntax Decoding using a SCFG . . . . .	91
4.2.6	Mixed-Syntax Model . . . . .	93
4.2.7	Formal Definition of the Mixed-Syntax Model . . . . .	93
4.2.8	Decoding with the Mixed-Syntax Model . . . . .	94
4.2.9	Feature Functions . . . . .	96
4.2.10	Example Decoding with the Mixed-Syntax Model . . . . .	97
4.2.11	Tree-to-String Model . . . . .	98
4.3	Rule Extraction . . . . .	100
4.3.1	Hierarchical Phrase-Based Model Extraction . . . . .	100
4.3.2	Tree-to-string Model Extraction . . . . .	101
4.3.3	Mixed-Syntax Model Extraction . . . . .	102
4.3.4	Examples of Extracted Rules . . . . .	104
4.4	Experiments . . . . .	107
4.4.1	Results . . . . .	109
4.4.2	Rule Span Width During Decoding . . . . .	110
4.4.3	Example Decoding Derivation . . . . .	111
4.5	Using Shallow Parsers . . . . .	113
4.5.1	Experiments with Shallow Parsers . . . . .	114
4.6	English to German . . . . .	116
4.7	Large Training Corpora . . . . .	117
4.8	Improved Mixed-Syntax Extraction . . . . .	118
4.8.1	Issues with Mixed-Syntax Extraction . . . . .	119
4.8.2	New Mixed-Syntax Extraction Algorithm . . . . .	121
4.8.3	Results . . . . .	125
4.9	Conclusion . . . . .	125

<b>5</b>	<b>The Discriminative Power of the Translation Model</b>	<b>127</b>
5.1	Training and Using the classifier . . . . .	128
5.1.1	Example . . . . .	129
5.2	Definition . . . . .	130
5.3	Previous Work . . . . .	131
<b>6</b>	<b>Conclusion and Future Work</b>	<b>133</b>
6.1	Future Work . . . . .	135
	<b>Bibliography</b>	<b>137</b>

# List of Figures

2.1	Basic Translation Models . . . . .	17
2.2	Factorizing into 1 translation model and 1 generation model . . . . .	18
2.3	Analysis and Generation Translation Model . . . . .	18
2.4	Stack Decoding algorithm . . . . .	21
2.5	Creating output string for a generation step . . . . .	26
2.6	Creating translation options . . . . .	27
2.7	Joint translation model of surface and POS tags . . . . .	35
2.8	Creating translation options 2 . . . . .	42
3.1	Example alignment matrix for 3-word source, 2-word target sentence .	58
3.2	Combined phrase-pair . . . . .	61
3.3	Creating Template Translation Options . . . . .	64
3.4	Average distortion cost per sentence . . . . .	68
3.5	Zipfian distribution of usage of factored template model . . . . .	71
3.6	Zipfian distribution: # examples against rank of phrase-pair (log-log scale) . . . . .	71
3.7	Percentage of correctly ordered NOUN ADJ phrases (100 samples) . .	72
3.8	Percentage of correctly ordered NOUN ADJ CONJ ADJ phrases (69 sam- ples) . . . . .	73
4.1	Definition of hierarchical phrase-based rules extraction from bilingual phrases . . . . .	101
4.2	Definition of tree-to-string rules extraction from bilingual phrases . .	102
4.3	Definition of mixed-syntax rules extraction from bilingual phrases . .	103
4.4	Aligned parsed sentence . . . . .	105
4.5	Source span lengths . . . . .	110
4.6	Length and count of glue rules used decoding test set . . . . .	111
4.7	Example input parse tree . . . . .	112

4.8	Derivation with Hierarchical model . . . . .	112
4.9	Derivation with mixed-syntax model . . . . .	113
4.10	Rules used by the mixed-syntax model to translate sentence . . . . .	113
4.11	Chunk - Length and count of glue rules used decoding test set . . . . .	115
4.12	Chunked sentence . . . . .	115
4.13	Translated chunked sentence . . . . .	116
4.14	Number of unique target string per sentence in 100-best list during tuning	118
4.15	Example partial sentence, alignment information, and parse structure	120
4.16	Identifying and Creating Lattice for Rule Extraction . . . . .	122
4.17	Creating Mixed-Syntax Translation Rules . . . . .	123
4.18	Example sentence, alignment information, and parse structure . . . . .	124
5.1	Search space learned from 3 parallel sentences . . . . .	129

# List of Tables

1.1	Experimental variables of each experiment in the thesis . . . . .	7
2.1	Training, tuning, and test conditions . . . . .	29
2.2	Source factors helps to disambiguate source words . . . . .	30
2.3	Statistics for source language of (in-domain) test set . . . . .	30
2.4	Disambiguation of source test set . . . . .	31
2.5	Percentage of correct translation of <i>das</i> . . . . .	32
2.6	Manual evaluation of translation of <i>sein</i> . . . . .	32
2.7	Generating target factors . . . . .	34
2.8	Translating source and target POS factors . . . . .	36
2.9	POS sequence model . . . . .	37
2.10	Number of occurrences of 1-word conjugations of <i>sein</i> . . . . .	38
2.11	Number of unique tokens . . . . .	39
2.12	Decoding with decomposed model . . . . .	39
2.13	Number of partial translation options for 100 input sentences . . . . .	41
2.14	Decoding with decomposed model and phrase-based model . . . . .	43
2.15	Number and rate of OOV words (Out-of-domain) . . . . .	44
2.16	# and rate of OOV words (In-domain) . . . . .	44
2.17	Manual Evaluation of 100 OOV words . . . . .	44
2.18	Training, tuning, and test conditions . . . . .	45
2.19	Results when trained with Europarl corpus . . . . .	46
3.1	French–English translation of <i>nom adj</i> . . . . .	50
3.2	French–English translation of <i>nom adj kon adj</i> . . . . .	50
3.3	Example phrase-pairs with alignment information . . . . .	60
3.4	German–English results, in %BLEU . . . . .	65
3.5	French–English results, in %BLEU . . . . .	66
3.6	Varying the maximum template lengths and effect on translation quality	66

3.7	French–English results for lexicalized reordering model . . . . .	67
3.8	Number of translation options to decode out-of-domain test set, French– English . . . . .	68
3.9	Size of source segments decoding out-of-domain French–English . . .	69
3.10	Top 10 POS tag sequence translated by factored template phrases . . .	70
3.11	French–English results, trained on Europarl corpus . . . . .	73
4.1	Syntax Formalisms . . . . .	83
4.2	Training, tuning, and test conditions . . . . .	108
4.3	German–English results for hierarchical and syntactic models, in %BLEU	109
4.4	Effect on %BLEU of varying maximum number of non-terminals . . .	110
4.5	Example input and output decoding with each model . . . . .	111
4.6	German–English results for hierarchical and mixed-syntax models us- ing chunk tags, in %BLEU . . . . .	114
4.7	Example best output by mixed-syntax model with chunk tags . . . . .	115
4.8	English–German results in %BLEU . . . . .	116
4.9	Training, tuning, and test conditions . . . . .	117
4.10	German–English results, trained on Europarl corpus, in %BLEU . . .	117
4.11	Number of translation rules extracted for each model . . . . .	125
4.12	German–English results, using new extraction algorithm, in %BLEU .	125
4.13	Effect of varying max. span limit on translation quality, in %BLEU . .	126

# Chapter 1

## Introduction

Machine translation (MT) is the application of computers to automatically translate human languages.

There are a variety of reasons to pursue research into MT. It has obvious practical usage in enabling people to communicate with others who do not share a common language. There is simply too much information that we would like to disseminate and understand in other languages to be translated by hand. As the world becomes more globalized, this problem can only become more acute. Human translators are a limited and expensive resource. Machine translation can increase the efficiency of human translators, replace them entirely, or perform the tasks which would have otherwise have been left undone.

Furthermore, the modality of communication has become increasingly varied and instantaneous. Email, mobile texting, instant messaging, online social media and video conferencing are an integral part of today's information society. Machine translation offers the direct and immediate response that would be difficult to achieve with human translators.

The study of machine translation gives us an insight into language and linguistics, and to test the utility of our own understanding of these subjects. Lastly, the idealistic goal of promoting accord and understanding by removing the language barrier has been the noble aim of many researchers.

### 1.1 Machine Translation Approaches

With such strong motivation, it is not surprising that research into machine translation has a long history stretching back as far as the invention of modern computers in the

1940s and 1950s (Hutchins, 2000).

Early MT systems relied on a collection of translation rules which were manually created. These so-called *rule-based* systems are based on linguistically-informed foundations requiring extensive morphological, syntactic and semantic knowledge. The input is transferred to the target using a large set of sophisticated linguistic translation rules. Translation rules are created manually, demanding significant multilingual and linguistic expertise. Therefore, rule-based systems require large initial investment and maintenance for every language pair.

Within the rule-based paradigm, different approaches exist. *Transfer-based* machine translation seeks to analyze the grammatical structure of the input. The output is generated using translation rules which act on this structure. This is the most common rule-based approach, used by systems such as Systran <sup>1</sup> and OpenLogos <sup>2</sup> and has been shown to achieve high quality translation for limited domain (Hutchins and Somers, 1992). However, the analysis, transfer and generation stages follow each other in sequence, therefore, the multiplication of errors in each stage depresses translation quality.

*Interlingua* machine translation strives to understand the semantics of the input and to capture it in a language-independent representation. The interlingua representation can then be used to generate the output for any given language, given the appropriate generation model for that language. While this may be an attractive concept, in practice it is extremely difficult to create a representation for human language, parse the source sentence into such a representation, and from it generate the target sentence.

## 1.2 Statistical Machine Translation

The last two decades have seen a new direction of research where translation rules are automatically created from parallel corpora. A key part of this *statistical machine translation* (SMT) approach is much more reliant on data intensive methods and statistical techniques, coupled with the dramatic increase and availability of computing power. Translation rules in SMT systems do not have the human insight of the hand-written rules and are often noisy but they can be created within hours or days rather than months or years.

As with the rule-based paradigm, statistical machine translation has also advanced

---

<sup>1</sup> <http://www.systransoft.com/>

<sup>2</sup> <http://logos-os.dfki.de/>



in a sequence of approaches. The earliest approach starting with (Brown et al., 1993) was the *word-based* model which, as the name suggests, translates word-for-word. These models have largely been superceded by more complex models but they survive in specific areas such as word alignment (Al-Onaizan et al., 1999) where they serve as the basis for other models.

Phrase-based models proposed by Zens et al. (2002); Koehn (2004); Koehn et al. (2007) translate contiguous sequences of words in the source sentence to contiguous words in the target. The term *phrase* in this case just means contiguous words, rather than any syntactic phrasal category,

The phrase-based models significantly improve on the word-based models and remain the state-of-the-art for many language pairs, especially between closely related languages. A reason for their success is their modelling of local reordering and the implicit assumption that most orderings are monotonic, which is usually the case when translating between closely related languages.

A notable variation on the march from word-based to phrase-based is the *alignment template system* (ATS) (Och and Ney, 2004) which uses a word-based translation model for lexical translation but models reordering using templates of word-classes.

The *hierarchical phrase-based model* (Chiang, 2005) extends the phrase-based notion of a phrase from a sequence of words to a sequence of words and *subphrases*. Translation rules in hierarchical phrase-based models leverage the strengths of phrase-based models and model reordering of subphrases too. For example, the French translation of ‘John misses Mary’ reorders the object and subject, ‘Marie manque à Jean’. The reordering can be expressed in a hierarchical rule, lexicalized with the translation of ‘misses’:

$$\langle X_1 \text{ misses } X_2 , X_2 \text{ manque à } X_1 \rangle \quad (1.1)$$

where  $X_1$  and  $X_2$  are placeholders for subphrases. Hierarchical translation rules can also model discontinuous phrasal phenomena such as the translation of the French *ne...pas* phrase.

$$\langle \text{ne } X_1 \text{ pas} , \text{do not } X_1 \rangle \quad (1.2)$$

Formally, translation rules in the hierarchical phrase-based model are production rules of a synchronous context free grammar (SCFG). Translation with a SCFG parses the input while simultaneously generating the output. The model is said to be a formally syntax-based MT model without linguistic commitment as its rules are induced from a parallel corpus without reference to any linguistic annotations or assumptions.

The hierarchical phrase-based model is particularly suited for language pairs, such as Chinese-English, which require long-range reordering.

### 1.3 Why use Linguistic Information in SMT?

Even at the inception of SMT, the utility of linguistic information to translation was acknowledged. To quote from Brown et al. (1993):

*...it is not our intention to ignore linguistics, neither to replace it... we hope to enfold it in the embrace of a secure probabilistic framework ... and guide us to better natural language processing systems in general and to better machine translation systems in particular.*

Attempts to form this embrace have been the subject of research ever since.

Statistical machine translation often makes no use of linguistic information, relying purely on corpus data and statistical modelling to train and decode. This is a convenient, language independent approach which allows translation systems for any language pairs to be created quickly, given the corpora. However, parallel corpus data is an expensive resource and not always available in the quantity required to build models which can perform to acceptable standards. This problem is exacerbated for highly inflected languages and translation of specialized domains.

It is conjectured that by combining linguistic insight and the strength of the corpora-based SMT approach, we may be able to create better translation systems. But what are the shortcomings with machine translation and how in particular can linguistic information help?

Firstly, natural language is ambiguous. Phenomena such as homographs mean that we cannot be certain of the use or meaning of some words and how those words are to be translated. Augmenting such words with linguistic information may help with their disambiguation and improve translation.

Secondly, language is sparse. Many words and phrases are infrequently seen in the training corpora, or not seen at all. Linguistic information can be used to generalize translation, helping to overcome this sparsity.

Thirdly, machine translation is a practical application which must operate within an environment of finite training corpora, memory and time resources. Arbitrary limits are often placed on the translation process to ensure that the training and decoding remain tractable, at the cost of reduced performance. Better translation may result if linguistically-motivated constraints replace these arbitrary limits.

The integration of linguistic information into SMT has not been smooth or guaranteed to improve translation, and in many cases, adding linguistic information decreases performance. There are some broad reasons why this is the case.

First, it would be surprising if the linguistic information obtained from tools such as taggers, parsers and morphological analyzers were to be perfectly suited for machine translation out-of-the-box. Their objectives are not necessarily aligned with those of machine translation. Also, integrating such information may make existing linguistic-free SMT methods more complicated and error prone. Lastly, the linguistic tools themselves are subject to errors and ambiguity, therefore, they cannot always be relied upon.

Therefore, to successfully exploit linguistic information it is necessary to use those parts which are useful, and alter or overcome those which are not.

## 1.4 Thesis Outline and Contribution

The major contributions of this thesis are as follows:

In Chapter 2, we present the *factored phrase-based model* which extends the phrase-based model of Koehn et al. (2003) to incorporate linguistic information as additional factors in the word representation. When augmenting the surface string with part-of-speech tags, lemma and morphological information, we show that this aids in the disambiguation of source words. Output grammaticality is also improved with sequence models on target factors.

We also studied how the factored translation model can be decomposed into multiple steps where each step outputs a subset of the target factors, conditioned on a subset of the available source or target factors. We show how this can be constructed to improve coverage of previously out-of-vocabulary words. This leads to overall better translation when the decomposed model is combined with a standard factored phrase-based model.

In Chapter 3, the issue of weak reordering models in current phrase-based systems is discussed. Introducing reordering into SMT models also dramatically expands the search space, therefore, arbitrary limits are placed on reordering for efficient decoding. However, this negatively affects translation quality. We describe the *factored template model* which uses translations of linguistic properties (such as POS tags) to act as templates for reordering of surface phrases. When the factored template model is combined with a standard factored phrase-based model, we see improved short and

mid-range reordering, leading to overall better translation.

Both the factored model and factored template model, above, are wedded to word-level information which limits its usefulness for improving long-range dependency.

In Chapter 4, we extend the word-level information to information on continuous spans of the source sentence. We also transition from the phrase-based model to a model based on synchronous context-free grammar (SCFG). SCFG-based models offer a natural fit with multi-word linguistic annotation that we are interested in studying. Chapter 4 presents a novel tree-to-string model, the *mixed-syntax model*, which combines the specificity of syntactic models and the generality of the hierarchical phrase-based model. We show how this model outperforms both the tree-to-string and hierarchical phrase-based models.

We conclude in Chapter 6 with some discussion of future work.

## 1.5 Experimental Setup

We used the Moses SMT toolkit (Koehn et al., 2007) throughout the thesis for training and decoding. Table 1.1 list the pertinent experimental variables used in each chapter. Due to these differences and the drift of the Moses codebase over time, the models are comparable only with their baseline but not strictly comparable between chapters.

	Chapter 2		Chapter 3		Chapter 4		
	Small-scale	Large-scale	Small-scale(De-En)	Small-scale(Fr-En)	Large-scale(Fr-En)	Small-scale (chunks tags)	Large-scale
<b>Data</b>							
Training corpus	proj-syndicate2 dev2006	europarl-v5 dev2006	proj-syndicate2 dev2006	news-commentary dev2006	europarl-v3 dev2006	news-commentary09 news-dev2009a	europarl-v5 dev2006
Tuning corpus	nc-devtest2007 devtest2006	devtest2006 newstest2007	nc-devtest2007 devtest2006	nc-devtest2007 devtest2006	test2007 devtest2006	news-dev2009b news-dev2009b	devtest2006 nc-dev2007
Test corpus (in-domain)							
Test corpus (out-of-domain)							
<b>Training</b>							
Translation model feature functions	$p(f s), p(s f), lex(f s), lex(s f),$ phrase count <sub>1</sub>	as (1) Truecase	as (1) Lowercase	as (1) Lowercase	as (1) Lowercase	$p(f s),$ phrase count <sub>2</sub> Truecase	as (2) Truecase
Casing methodology	Lowercase	as (1)	as (1)	as (1)	as (1)	as (1)	as (1)
LM smoothing	interpolate, K-N discounting <sub>1</sub>	as (1)	as (1)	as (1)	as (1)	as (1)	as (1)
Phrase table smoothing	Unsmoothed	Good-Turing	Unsmoothed	Unsmoothed	Unsmoothed	Good-Turing	Good-Turing
Maximum training sentence length	40	80	40	80	80	80	80
Exclude non-parse sentences	No	No	No	No	No	Yes	Yes
<b>Tuning and Evaluation</b>							
Automate evaluation metric	nist-bleu	nist-bleu	nist-bleu	nist-bleu	nist-bleu	nist-bleu	multi-bleu
Case-sensitive evaluation	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Exclude non-parse sentences	No	No	No	No	No	No	Yes
No translation for non-parse sentences	No	No	No	No	No	No	No

Table 1.1: Experimental variables of each experiment in the thesis

## 1.6 Related publications

This thesis is based on the following publications:

1. Parts of Chapter 2 were presented in our published papers ‘Open Source Toolkit for Statistical Machine Translation’ (Koehn et al., 2006) and ‘Factored Translation Models’ (Koehn and Hoang, 2007), presented at the Johns Hopkins University Summer Workshop 2006 and in the proceedings of Empirical Methods in Natural Language Processing, respectively. This is joint work with Philipp Koehn et al. The software developed for this was also described in Koehn et al. (2007); Hoang and Koehn (2008).
2. Chapter 3 expands on ‘Improving Mid-Range Re-Ordering using Templates of Factors’ (Hoang and Koehn, 2009), published in the proceedings of the European chapter of the Association of Computational Linguistics.
3. Chapter 4 is based on ‘Improved Translation with Source Syntax Labels’ (Hoang and Koehn, 2010) published in the proceedings of the Association of Computational Linguistics workshop on machine translation. The accompanying software is described in Hoang et al. (2009).

# Chapter 2

## Factored Translation

Many language phenomena are strongly correlated to the linguistic properties of words, for example, adjective-noun agreements in Romance languages are dependent on the word classes, number and gender agreements are dependent on the word morphology. These linguistic properties are latent in standard phrase-based models and must be inferred from context and the surface form. We study the advantages of making these properties explicit. Also, the linguistic properties are usually less sparse than the surface form, making it easier to collect reliable statistics. We study how linguistic properties can be used to generalize existing translation and language models.

This chapter begins our study of how linguistic information can help statistical machine translation, specifically phrase-based SMT, what kind of linguistic information, and the issues that using it may bring up. We describe the ideas and components behind the factored translation model and present results on German-English experiments on the small news commentary. This chapter also lays the foundation for the next chapter on factored templates where we focus on using linguistic information to improve reordering.

In the process, we also create a framework for integrating word-level information into phrase-based SMT. Parts of this chapter is also based on (Hoang and Koehn, 2008; Hoang et al., 2009), which described the publicly available Moses toolkit which was created for this thesis. Other papers on this toolkit include (Koehn and Hoang, 2007; Koehn et al., 2006).

## 2.1 Model

### 2.1.1 Phrase-Based Model

The *factored translation* model is based on the phrase-based model (Koehn et al., 2003) and extends it to model variables representing linguistic information. We therefore take time to recap the phrase-based model before giving the formal definition of factored translation model.

SMT models the probability of a target translation, the objective of SMT is to find the target translation with the maximum probability, given a source sentence. That is, for a source sentence  $s$ , the objective is to find a target translation  $\hat{t}$  which has the highest conditional probability  $p(t|s)$ . Mathematically, this is written as:

$$\hat{t} = \arg \max_t p(t|s) \quad (2.1)$$

where the *arg max* function is the search. The model probability,  $p(t|s)$ , can be factorized into three parts using Bayes' rule.

$$p(t|s) = \frac{p(t)}{p(s)} p(s|t) \quad (2.2)$$

The noisy channel model of SMT uses this reformulation to express its objective:

$$\hat{t} = \arg \max_t p(t)p(s|t) \quad (2.3)$$

The prior,  $p(s)$ , can be omitted as it is constant for each source sentence. The probability of the output,  $p(t)$ , is approximated by an n-gram language model (Stolcke, 2002).

The translation probability  $p(s|t)$  is calculated by decomposing the translation into multiword phrases. The phrase-based model translates sentences by segmenting the source into contiguous segments and translating each segment independently. The translated segments can be in a different order on the target side.

We define a variable  $b = (start_s, end_s, start_t, end_t)$  as two sets of contiguous spans on the source and target sentence, and  $p(s^b|t^b)$  as the probability that the target phrase in span  $[start_t, end_t]$  is translated to the source phrase in span  $[start_s, end_s]$ .

$$\begin{aligned} p(s|d,t) &= \prod_{b \in d} p(s^b|t^b) \\ p(s,d|t) &= p(s|d,t)p(d|t) \end{aligned} \quad (2.4)$$



where  $d = \{b^*\}$  is a set of  $b$  which covers all words in the source and target sentence once and only once.  $p(s^b|t^b)$  is the phrasal translation probability learnt from a parallel corpus, using the relative values of the source and target spans in  $b$ .

$p(d|t)$  combines the reordering and segmentation probabilities which is usually termed the *derivation probability*. In phrase-based models, this is weakly modelled by a relative distance-based distortion model and phrase count features. The translation probability  $p(s|t)$  is calculated by summing over all possible derivations of the source sentence.

$$p(s|t) = \sum_d p(s, d|t) \quad (2.5)$$

Inserting this back into the noisy channel model results in the following formula:

$$\begin{aligned} \hat{t} &= \arg \max p(t)p(s|t) \\ &= \arg \max p(t) \sum_d p(s, d|t) \end{aligned} \quad (2.6)$$

This approach is known as maximum translation (Blunsom and Osborne, 2008). However, the summation over all segmentations is intractable for all but the smallest models and short sentences. Therefore, the maximum derivation approach approximates  $p(t|s)$  by using the segmentation with the highest probability. This is commonly known as the Viterbi approximation (Jelinek, 1998).

$$\begin{aligned} \hat{t} &= \arg \max_t p(t)p(s|t) \\ &\approx \arg \max_{t,d} p(t)p(s, d|t) \\ &\approx \arg \max_{t,d} p(t)p(d|t) \prod_{b \in d} p(s^b|t^b) \end{aligned} \quad (2.7)$$

The log-linear model generalizes the noisy channel model to include more component models and weighting each model according to the contribution of each model to the total probability.

$$p(t|s) = \frac{1}{Z} \exp\left(\sum_m \lambda_m h_m(t, s)\right) \quad (2.8)$$

where  $\lambda_m$  is the weight, and  $h_m$  is the feature function, or ‘score’, for model  $m$ .  $Z$  is the partition function which can be ignored for optimization. The log-linear formulation in phrase-based SMT uses log probabilities as feature functions, in addition to features which do not have a probabilistic interpretation. Typical feature functions include the log transforms of the translation model probabilities,  $p_{TM}(t|s)$  and  $p_{TM}(s|t)$ , which we have suffixed with  $_{TM}$  to avoid confusion with the overall model probability  $p(t|s)$  and

$p(s|t)$ . The normalization factor  $Z$  does not affect optimization, and the  $\exp()$  function is monotonic, therefore the log-linear model can be expressed as a linear model.

$$\begin{aligned}\arg \max_t p(t|s) &= \arg \max_t h(t, s) \\ \arg \max_t h(t, s) &= \arg \max_t \sum_m \lambda_m h_m(t, s)\end{aligned}\quad (2.9)$$

## 2.1.2 Factored Model

The *factored model* extends the standard phrase-based model by redefining a word from a single symbol to a vector of factors. The surface string is a factor for each word but additional factors can be added as required, in source and target words. A phrase,  $s$  and  $t$ , remains a sequence of words:

$$\begin{aligned}s &= [s^1, s^2, \dots, s^{|s|}] \\ t &= [t^1, t^2, \dots, t^{|t|}]\end{aligned}$$

where  $s^i$  is the  $i^{\text{th}}$  source word, and similarly for  $t^i$ . Each word is composed of a vector of factors:

$$\begin{aligned}s^i &= [s_1^i, s_2^i, \dots, s_{|S|}^i]' \\ t^i &= [t_1^i, t_2^i, \dots, t_{|T|}^i]'\end{aligned}$$

where word  $s^i$  contains factors  $s_1^i, s_2^i, \dots, s_{|S|}^i$ .  $S$  is an ordered set of source factor types.  $t_1^i$  and  $T$  are similarly defined for the target language. For convenience, we can refer to a factor by name or by index, and we can refer to a sequence of factors within a phrase. For example, the sequence of (1) surface factors in the target string  $t$ , and (2), the part-of-speech and lemma in the  $i^{\text{th}}$  source word is shown below:

$$\begin{aligned}t_{\text{surface}} &= [t_{\text{surface}}^1, t_{\text{surface}}^2, \dots, t_{\text{surface}}^{|t|}] \\ s_{\text{POS,lemma}}^i &= [s_{\text{POS}}^i, s_{\text{lemma}}^i]'\end{aligned}$$

All source factors are given as input. The target surface factors are the output of the model, while the other target factors are latent variables. The factored model reformulates the basic SMT formula of Equation 2.1 to take factors into account.

$$\hat{t}_{\text{surface}} = \arg \max_{t_{\text{surface}}} p(t_{\text{surface}}|s) \quad (2.10)$$

Translation is modelled as a process which jointly translates all target factors, conditioned on all source factors,  $p(t|s)$ . However, we are only concerned with the surface

string output, therefore, the conditional probability of the target surface string can be arrived at by marginalizing the other target factors. Assuming the surface factor is  $t_1$ :

$$p(t_1|s) = \sum_{t_2} \sum_{t_3} \dots \sum_{t_{|T|}} p(t_1, \dots, t_{|T|}|s) \quad (2.11)$$

However, this summation is often intractable so the maximum derivation principle recalling Equation 2.7, is applied to find  $\hat{t}_{surface}$ :

$$\hat{t}_{surface} \approx \arg \max_t p(t|s) \quad (2.12)$$

This allows us to focus on modelling translation as the joint probability of all target factors, given all source factors,  $p(t|s)$ . Without loss of generality, we can use the chain rule to rewrite this probability as

$$\begin{aligned} p(t|s) &= p(t_1, \dots, t_{|T|}|s) \\ &= p(t_1|s)p(t_2|t_1, s)p(t_3|t_2, t_1, s) \dots p(t_{|T|}|t_{|T|-1:1}, s) \end{aligned} \quad (2.13)$$

where  $f$  is an index into the ordered set of target factor types  $T$ .

The factored model limits the factorization of the conditional probability to two types of *mapping steps*:

1.  $p(t_{out}|s_{in})$ , the conditional probability of a non-empty set target factors, given a non-empty set of *source* factors. We call this a *translation step*.
2.  $p(t_{out}|t_{in})$ , the conditional probability of a non-empty set target factors, given a non-empty set of *target* factors. We call this a *generation step*.

Factorizing the probability with these limitations requires conditional independence assumptions; factorizing with a translation step implies the independence of the set of target factors, *out*, from other source and target factors, given the source factors *in*. Likewise, using a generation step implies the independence of set *out* of target factors, from other target and all source factors, given the target factors *in*.

The phrase-based model in Section 2.1.1 relies on the log-linear model to provide the foundation for incorporating the translation model,  $p_{TM}(t|s)$ , in addition to  $p_{TM}(s|t)$  as features in the model. The factored model seeks to factorize the translation model probabilities, rather than the overall probability. Therefore, the factored model decomposes the translation model,  $p_{TM}$ , into a series of mapping steps in Equation 2.14.

$$p_{TM}(t|s) = \prod_{r \in \tau} p(t_{out(r)}|s_{in(r)}) \times \prod_{g \in \gamma} p(t_{out(g)}|t_{in(g)}) \quad (2.14)$$

where

1.  $\tau$  is the set of translation steps
2.  $\gamma$  is the set of generation steps
3.  $out(i)$  is the set of factor types for step  $i$  that the step is estimating
4.  $in(i)$  is the set of factor types for step  $i$  on which it is conditioned.

For example, when  $|T| = 3$  and  $|S| = 3$

$$p(t|s) = p(t_1, t_2, t_3 | s_1, s_2, s_3) \quad (2.15)$$

If we want to factorize this into three mapping steps:

1.  $t_1$  is created by a generation step from  $t_2$  and  $t_3$
2.  $t_2$  is created by a translation step from  $s_2$
3.  $t_3$  is created by a translation step from  $s_3$

We can use the chain rule without making any assumptions:

$$p(t|s) = p(t_1, t_2, t_3 | s_1, s_2, s_3) \quad (2.16)$$

$$= p(\mathbf{t}_1 | \mathbf{t}_2, \mathbf{t}_3, s_1, s_2, s_3) p(\mathbf{t}_2 | t_3, s_1, \mathbf{s}_2, s_3) p(\mathbf{t}_3 | s_1, s_2, \mathbf{s}_3) \quad (2.17)$$

However, the factorization as outlined above makes independence assumptions:

$$t_1 \perp s_1, s_2, s_3 | t_2, t_3 \Rightarrow p(\mathbf{t}_1 | \mathbf{t}_2, \mathbf{t}_3, s_1, s_2, s_3) = p(t_1 | t_2, t_3)$$

$$t_2 \perp t_1, t_3, s_1, s_3 | s_2 \Rightarrow p(\mathbf{t}_2 | t_3, s_1, \mathbf{s}_2, s_3) = p(t_2 | s_2)$$

$$t_3 \perp t_1, t_2, s_1, s_2 | s_3 \Rightarrow p(\mathbf{t}_3 | s_1, s_2, \mathbf{s}_3) = p(t_3 | s_3)$$

$$\Rightarrow p(t|s) = p(t_1 | t_2, t_3) p(t_2 | s_2) p(t_3 | s_3)$$

The probability of each translation step,  $p(t_{out(r)} | s_{in(r)})$ , is calculated as the product of the probability of a series of multiword phrases in the same way as standard phrase-based models. However, the segmentation of all factorized translation steps are identical. This simplifies decoding so that the combined translation model,  $p_{TM}(t|s)$ , can be constructed during the preprocessing phase before decoding, making the decoding algorithm identical to the standard, non-factored algorithm. Each generation step is processed word-by-word.

$$\prod_{r \in \tau} p(t_{out(r)} | s_{in(r)}) = \prod_{r \in \tau} \prod_{b \in d} p_{TM}(t_{out(r)}^b | s_{in(r)}^b) \quad (2.18)$$

$$\prod_{g \in \gamma} p(t_{out(g)} | t_{in(g)}) = \prod_{g \in \gamma} \prod_{i=1..|t|} p(t_{out(g)}^i | t_{in(g)}^i) \quad (2.19)$$

where  $d$  is the set of coupled source and target spans that covers every word position once and only once in  $s$  and  $t$ . The log probability of each step is simply used as a feature function within the log-linear model.

$$\{h_r = \log p(t_{out(r)}|s_{in(r)})\}_{r \in \tau} \quad (2.20)$$

$$\{h_g = \log p(t_{out(e)}|t_{in(g)})\}_{g \in \gamma} \quad (2.21)$$

### 2.1.3 Factor Sequence Models

The probability of the target surface string  $p(t_{surface})$  is often estimated with a Markov model of the target language (Stolcke, 2002), called an n-gram language model (LM), which operates on the sequence of surface strings. From empirical evidence, the language model is an important component model in phrase-based SMT that improves output fluency and grammaticality, assisting the MT decoder to make better lexical choices and re-ordering phrases.

However, there are limitations with using language models on surface forms. Firstly, data sparsity hinders the accuracy of probability estimates, especially for high-order n-gram models. Backoff strategies and interpolation may enhance the accuracy of high-order models a little but a realistic maximum language model n-gram is still very short, typically 3-5 words.

Secondly, language models do not attempt to use the linguistic relationship of words to improve accuracy. Factored language models (FLM) (Bilmes and Kirchhoff, 2003) overcome this by using linguistic information such as POS tags and morphology. FLM view words as a vector of factors, just as in the factored translation, using the surface forms but backing off to other factors where necessary. FLM models predict the sequence of factors,  $p(t_{1:K}^i | t_{1:K}^{i-1:i-n})$ , where  $n$  is the n-gram order, rather than just the sequence of surface factors.  $K$  is the number of factors.

A *sequence model* only models the sequence of a specific factor,  $t_f$ . This contrasts with FLMs, which models the sequence of all factors,  $t_{1:K}$ . Similarly, a language model models the sequence of just the surface form,  $t_{surface}$ .

By targeting specific factors, the n-gram order of the sequence model can be adjusted to the data and sparsity of that factor. For instance, a higher order sequence model over POS tags may be possible due to the size of the tag set. This could improve grammaticality and long range ordering by guiding the translation to target strings with more plausible POS sequences.

We model the probability of the target sequence of factors  $p(t)$  as the weighted product of sequence models on each factor  $i$ . Each factor  $i$  may have multiple sequence models.

$$p(t) = \prod_{i=1}^{|T|} \prod_{j \in J_i} p_j(t_i)^{\lambda_{i,j}} \quad (2.22)$$

$$\log p(t) = \sum_{i=1}^{|T|} \sum_{j \in J_i} \lambda_{i,j} \log p_j(t_i) \quad (2.23)$$

where  $J_i$  is the set of sequence models for factor  $i$ ,  $p_j(t_i)$  is the probability of the sequence of factors  $t_i$  predicted by model  $j$ , and  $\lambda_{i,j}$  is the weight for factors  $i$ , model  $j$ . This formulation conveniently fits into the log-linear model used to evaluate the machine translation output by using the log transform of each sequence model probability as a feature function in the log-linear framework.

$$\{h_{i,j} = \log p_j(t_i)\}_{i,j} \quad (2.24)$$

## 2.1.4 Examples

The source and target factors can be viewed as vertices in a directed, acyclic hypergraph where each edge represents a mapping step. We illustrate the factored model with a series of hypergraphs, below. Figure 2.1(a) illustrates the traditional non-factored translation model which simply maps surface forms of the source to target. In the factored framework, the translation model parameter is expressed as a probability of factors.

$$p_{TM}(t|s) = p(t_{surface}|s_{surface}) \quad (2.25)$$

where  $t_{surface}$  and  $s_{surface}$  are a sequence of surface factors in the target and source language, respectively.

Figure 2.1(b) shows surface and part-of-speech factors used jointly in one translation model. Equation 2.26 is the translation probability for this model.

$$p_{TM}(t|s) = p(t_{surface,pos}|s_{surface,pos}) \quad (2.26)$$

The joint translation model, using all available factors in the source and target, serves as a baseline for later experiments when the translation is decomposed.

Figure 2.2(a) illustrates a factorization of the translation model  $p(t_{surface,pos}|s_{surface})$  into a translation and a generation step. The motivation in this model is to create POS

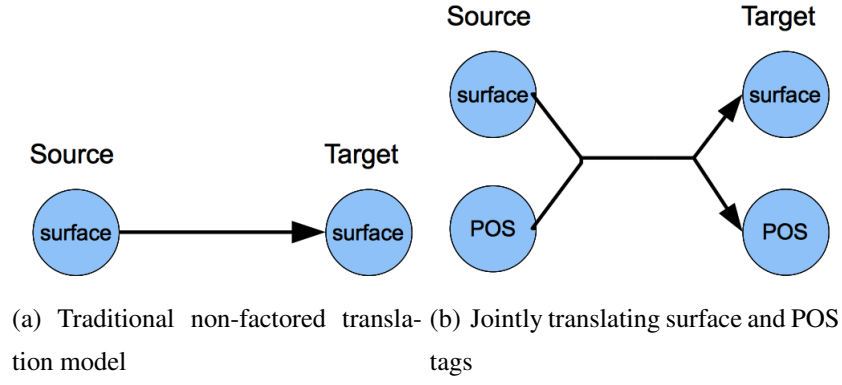


Figure 2.1: Basic Translation Models

tags for modelling with POS sequence models. By factorization, we hope to reduce data sparsity of the joint factor translation model and achieve better parameterization.

$$\begin{aligned}
 p_{TM}(t|s) &= p(t_{surface,pos} | s_{surface}) & (2.27) \\
 &= p(t_{surface} | s_{surface}) p(t_{pos} | t_{surface}, s_{surface}) \\
 &\approx p(t_{surface} | s_{surface}) p(t_{pos} | t_{surface}) \\
 t_{pos} &\perp s_{surface} | t_{surface}
 \end{aligned}$$

Figure 2.2(b) extends the previous model by simply creating two factors on the target side using the same generation model.

$$\begin{aligned}
 p_{TM}(t|s) &= p(t_{surface,pos,lemma} | s_{surface}) & (2.28) \\
 &= p(t_{surface} | s_{surface}) p(t_{pos}, t_{lemma} | t_{surface}, s_{surface}) \\
 &\approx p(t_{surface} | s_{surface}) p(t_{pos,lemma} | t_{surface}) \\
 t_{pos}, t_{lemma} &\perp s_{surface} | t_{surface}
 \end{aligned}$$

A model that we shall discuss later in this chapter is illustrated in Figure 2.3. This is an attempt to tackle the problem of data sparsity in the surface form by independently translating less sparsely linguistic factors which are then reconstructed in the target language with a generation step. This decomposition takes inspiration from the analysis-synthesis approach common in rule-based MT systems such as (Carne

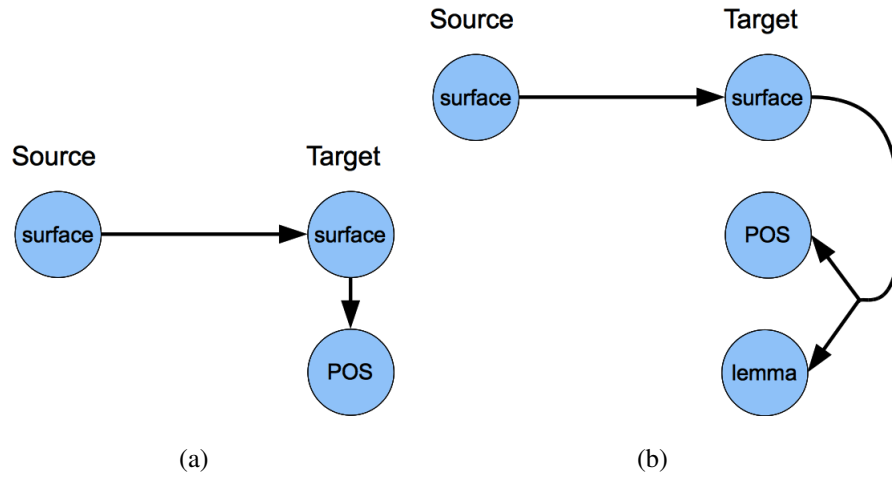


Figure 2.2: Factorizing into 1 translation model and 1 generation model

Armentano-Oller et al., 2005).

$$\begin{aligned}
 p_{TM}(t|s) &= p(t_{surface,lemma,POS}|t_{surface,lemma,POS}) & (2.29) \\
 &= p(t_{lemma}|s_{surface,lemma,POS})p(t_{pos}|s_{surface,lemma,POS})p(t_{surface}|t_{lemma,pos},s_{surface,lemma,POS}) \\
 &\approx p(t_{lemma}|s_{lemma})p(t_{pos}|s_{POS})p(t_{surface}|t_{lemma,pos})
 \end{aligned}$$

$$t_{lemma} \perp s_{surface}, s_{pos} | s_{lemma}$$

$$t_{pos} \perp s_{surface}, s_{lemma} | s_{pos}$$

$$t_{surface} \perp s_{surface}, s_{pos}, s_{lemma} | t_{pos}, t_{lemma}$$

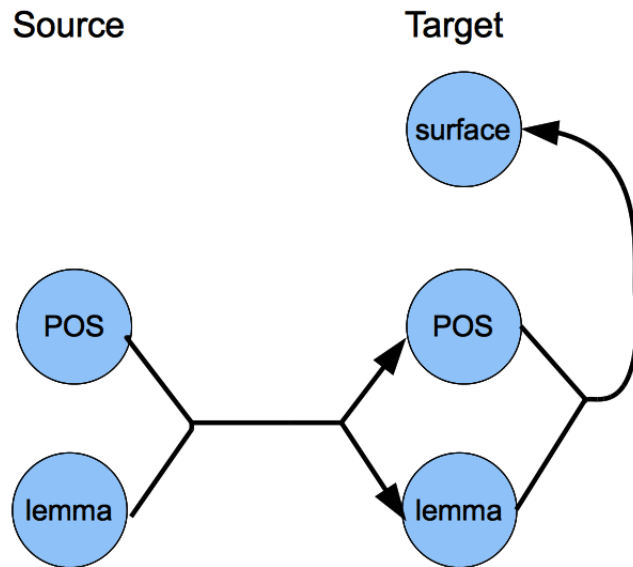


Figure 2.3: Analysis and Generation Translation Model



## 2.2 Decoding

In the previous section, we discussed the formal definition of decomposing a translation model into translation and generation steps. This section describe how to decode with translation and generation steps. The factored model is an extension of the phrase-based model so we will recap phrase-based decoding in detail before describing the factored model extension.

### 2.2.1 Phrase-Based Decoding

The task of SMT is to find the best translations, given a source sentence, which is formally encapsulated in the equation below:

$$\hat{t} = \arg \max_{t \in \mathcal{T}} p(t|s) \quad (2.30)$$

Decoding is the task of finding the most probable translation  $\hat{t}$  from all possible translations in  $\mathcal{T}$ , corresponding to the arg max function.

This section describes the phrase-based decoding algorithm, as implemented in Pharaoh (Koehn, 2004) and Moses (Hoang and Koehn, 2008). The algorithm is a dynamic program which has elements of both stack and beam searches (Jelinek, 1969). A heuristic is used to make hypotheses in the same stack comparable for pruning. Although pruning is applied, which makes the search algorithm not admissible, the results are good enough.

Target strings are constructed left-to-right on the target side. We describe the different phases of the decoding process below.

### 2.2.2 Translation Option

For each contiguous span of the source sentence, rules are found in the phrase table which matches the source phrase in the span. Only the top scoring rules are kept for the search stage while the rest are pruned according to the table pruning parameters. The future score of each rule is used to decide which rules to prune. This calculation is described in Section 2.2.6.

A *translation option* is a rule which has been retained after the table pruning that pertains to a specific source span. Translation options encapsulate the rule from the translation model as well as the local feature and estimated non-local feature scores.

Once all translation options for a particular sentence are created, the search algorithm can proceed.

### 2.2.3 Hypotheses

A translation of a source sentence is created by applying a series of translation options which together translate each source word once, and only once. Each partial translation is called a *hypothesis*, which is created by applying a translation option to an existing hypothesis. This process is called *hypothesis expansion* and starts with a hypothesis that has translated no source word and ends with a completed hypothesis that has translated all source words. The highest-scoring completed hypothesis, according to the model score, is returned as most probable translation,  $\hat{t}$ . Incomplete hypotheses are referred to as partial hypotheses.

Each translation option translates a contiguous sequence of source words but successive translation options do not have to be adjacent on the source side, depending on the distortion limit. However, the target output is constructed strictly left-to-right from the target string of successive translation options. Therefore, successive translation options which are not adjacent and monotonic in the source causes translation reordering.

To conserve memory and speed, the target output is not stored with each hypothesis. Instead, the hypothesis contains a reference to translation option and a backpointer to the best previous hypothesis it extended. The target output for any hypothesis can be computed by simply following the backpointer and recursively extracting each target phrase from the translation option (Ney and Ortmanns, 1997).

### 2.2.4 Beam Search

The aim of the search algorithm is to find the best scoring completed hypothesis. If the heuristic used to estimate the score to completion is admissible, that is, it never overestimates the score to translation the rest of the sentence of any partial hypothesis, it is guaranteed to find the optimal hypothesis. However, non-local feature functions such as language models and distortion models make estimating this heuristic intractable.

Neither is it computationally feasible to use a brute force method of evaluating every possible translation for anything but the smallest source input and very tight distortion limits. Instead we create a large number of completed hypotheses then choose the best hypothesis from this set. The set is a subset of all possible translations but it is hoped that this set will contain the best scoring hypothesis, or at least a good enough hypothesis.

A beam search algorithm is used to create the completed hypothesis set efficiently.

Partial hypotheses are organized into stacks where each stack holds a number of comparable hypotheses. Hypotheses in the same stack have the same coverage cardinality  $|C|$ , where  $C$  is the coverage set,  $C \subseteq \{1, 2, \dots, |s|\}$  of the number of source words translated. Therefore,  $|s| + 1$  number of stacks are created for the decoding of a sentence  $s$ . There exist other stack layouts (Ortiz-Martínez et al., 2006) but the use of coverage cardinality is the most common and the method we use.

The first stack,  $stack_0$ , is seeded with an empty hypothesis which has translated no source words. New hypotheses are created by extending the empty hypothesis with a translation option, the new hypothesis is inserted into the stack according to the number of source words translated.  $stack_1$  to  $stack_{|s|} - 1$  is processed in the same manner. At the end,  $stack_{|s|}$  contains completed hypotheses, i.e. those that have translated every word in the source sentence. The highest scoring hypothesis is returned from this set as the best translation. When extending hypotheses, a translation option can only be used if the resulting hypothesis translates each source word at most once and the distortion constraint is adhered to. The pseudocode for the stack decoding algorithm is shown in Figure 2.4.

```

insert empty hypothesis into  $stack_0$ 
for  $i = 0$  to  $|s| - 1$  do
    prune  $stack_i$ 
    for all hypothesis in  $stack_i$  do
        for all translation options do
            if hypothesis and translation option are compatible then
                expand hypothesis with translation option  $\rightarrow$  new hypothesis
                coverage of new hypothesis  $\rightarrow C$ 
                insert new hypothesis  $\rightarrow stack_{|C|}$ 
                recombine new hypothesis if possible
                prune  $stack_{|C|}$  if necessary
            end if
        end for
    end for
end for
return highest scoring hypothesis in  $stack_{|s|}$ 

```

Figure 2.4: Stack Decoding algorithm

## 2.2.5 Hypothesis Recombination

Hypothesis expansion can be optimized by considering the contextual information used by the language model. We start from the fact that two partial hypotheses which have identical coverage sets and are limited by identical distortion constraints can be expanded by the same set of translation options. The two hypotheses may differ in their derivations and score.

Therefore, in the absence of non-local feature functions, we can make a risk-free decision to discard the lower scoring hypothesis because it cannot surpass the higher scoring hypothesis. This decision is called *hypothesis recombination*.

However, in the presence of non-local feature functions such as a language model, we can still make the risk-free decision but we must change which properties of the hypotheses must match in order for the hypotheses to be recombined. A language model of  $n$ -gram order  $n$  requires the last  $n - 1$  target words of the previous hypothesis to score a new hypothesis. Two hypotheses which differ in their last  $n - 1$  target words may have different language model scores when expanded with the same translation option. Therefore, we can recombine the hypotheses only when their last  $n - 1$  target words are identical because only then can we guarantee that the relative scores of the hypotheses will remain the same. (In fact, language models often use a back-off strategy to deal with data sparsity where it uses information from a shorter  $n$ -gram if a long-gram does not exist. This information is captured in the language model state which we use to compare two strings, instead of the target words, and can result in more aggressive hypothesis recombination. This optimization is described in (Och and Ney, 2004).

Other non-local feature functions such as lexicalized re-ordering (Tillmann, 2004) use different contextual information to calculate their scores. This must also be taken into account when performing hypothesis recombination.

## 2.2.6 Pruning

Discarding hypotheses by recombination reduces the number of hypotheses that need to be evaluated. However, the search space for the phrase-based model is so large that hypothesis recombination alone is not enough to make the search algorithm tractable. We have to resort to risky pruning strategies which discard hypotheses that are unlikely, but not certain, to result in a good translation. A combination of *histogram pruning* and *threshold pruning* (Ney, 1992) is employed to discard low scoring hypotheses.

Pruning is isolated to individual stacks to enable some degree of comparability between hypotheses.

However, even in the same stack, hypotheses are not totally comparable. Hypotheses may exhibit systematic differences which favour those which translate ‘easy’ source phrases over hypotheses which have translated more difficult, ambiguous phrases. To offset this effect, the hypothesis’ estimated score to completion is used when considering them for pruning, rather than just their current score. The estimated score to completion is often called the *future cost* and is calculated by adding the current score with the estimated score of translating the remaining untranslated source words.

$$\begin{aligned} h(t, s) &\approx h(t', s, C) + h'(\bar{C}) \\ \bar{C} &= \{1, \dots, |s|\} - C \end{aligned} \quad (2.31)$$

where  $h(t', s, C)$  is the score of a hypothesis that has translated the source words in coverage set  $C$  as  $t'$ .  $h'(\bar{C})$  is the future cost.

The future cost of translating a contiguous span is calculated from the optimistic estimate of the set of translation options in the span using a dynamic programming algorithm. Formally,

$$\begin{aligned} h'([start, end]) &= \max \left[ \begin{array}{c} h'(O_{[start, end]}) \\ \max_{i \in [start, end]} h'(O_{[start, i]}) + h'([i + 1, end]) \end{array} \right] \\ h'(O_{[start, end]}) &= \begin{cases} \max_{o \in O} (h(o)) & \text{if } |O_{[start, end]}| > 0 \\ -\infty & \text{otherwise} \end{cases} \end{aligned} \quad (2.32)$$

where  $h'([i, j])$  is the estimate score for contiguous span  $[i, j]$ ,  $O_{[i, j]}$  is the set of translation options for the contiguous span  $[i, j]$ ,  $h'(O_{[i, j]})$  is best estimated score from the set of translation options  $O_{[i, j]}$ , where each translation option  $o$  has score  $h(o)$ .

$h(o)$  is the weighted local feature scores and estimates of non-local features. Local feature functions in the standard phrase-based model include the word count and phrase count features, and the translation model features.

Equation 2.32 computes the estimated score of all contiguous span. The estimated score of an arbitrary non-contiguous coverage set,  $h'(\bar{C})$ , can be calculated simply as the summation of the largest contiguous spans that composes it. The future cost,  $h'(\bar{C})$ , allows better comparison between hypotheses, reducing search errors due to stack pruning.

Language model require contextual information external to the phrase pair so cannot be calculated accurately before the search phase. Instead, an estimate of the language model score for each rule is used which models probability of the target words

in the rule but does not include the probability of n-grams that span adjacent rules. The language model probability is estimated as follows:

$$p'_{LM}(t^{1:|t|}) = p(t^1)p(t^2|t^1)\dots p(t^{|t|}|t^{1:|t|-1}) \quad (2.33)$$

where  $t$  is the target side of the rule and  $n$  is the n-gram order of the language model.

The distortion model is also a non-local feature. We do not estimate its contribution to the future score but some methods have been proposed to do this (Moore and Quirk, 2007; Green et al., 2010).

## 2.3 Factored Model Decoding

### 2.3.1 Construction of Translation Options

The phrase-based decoding algorithm creates translation options for each source sentence before decoding the sentence. Translation options in the standard phrase-based model are created by looking up phrase-pairs from a translation model.

The factored translation model extends the standard phrase-based model by composing translation options from a series of translation and generation steps, according to the pre-defined factorization. Each mapping step corresponds to a phrase table or generation model.

As defined in Equation 2.18, the creation of factorized translation options is made easier by constraining every translation step to using the same source and target segmentation during decoding. Therefore, each translation option contains exactly one phrase-pair from each translation step. All such phrase pairs for a particular translation option have the same source and target length and apply to the same source segment. Creation of translation options is also made easier by using the word-based generation steps of Equation 2.19.

The mapping steps are implemented as an ordered sequence rather than two unordered sets  $\tau$  and  $\gamma$  in Equation 2.14, however, this does not change the model. A sequence is chosen to reduce the computational burden of constructing translation options; the factors required as a condition in one step are created by the preceding steps. For example, the factorization

$$p_{TM}(t|s) = p(t_{surface}|s_{surface})p(t_{pos}|t_{surface}) \quad (2.34)$$

must be implemented as a sequence

1. translate surface forms
2. generate POS tags, given surface forms

The set of mapping steps must have at least one translation step. The first mapping step must be a translation step which creates an initial set of *partial translation options* by consulting its phrase table. To give an example, a two word French phrase *maison blanc* maps to several English phrases in the first translation step of the above factorization:

$$\text{maison blanc} \rightarrow \{ \text{white house, white home, white building} \}$$

A partial translation option is created from each English phrase. This is the full extent of the standard phrase-table procedure, but the factored model goes further.

In the factored model, the initial partial translation options seed the process of creating translation options. Each subsequent mapping step creates a set of partial translation options by merging the preceding partial translation options with the results of its own interrogation of its phrase table or generation table.

A subsequent *translation step* consults its phrases table to retrieve a set of phrase pairs. These phrase-pairs are merged with the preceding partial translation options by calculating the cartesian product of the phrase with the options. The result of a cartesian product is a set of tuples of a partial translation option and a target phrase. A new partial translation is formed if the partial translation option and the target phrase meet the following criteria:

1. The length of target side of the partial translation option and the target phrase are identical
2. The overlapping factors, if any, are identical

This can be defined formally as a binary function

$$f(ot_{fot}^{1:|ot|}, t_{ft}^{1:|t|}) = \begin{cases} 1 & (|ot| = |t|) \text{ and } (ot_{fot \cap ft}^{1:|ot|} = t_{fot \cap ft}^{1:|t|}) \\ 0 & \text{otherwise} \end{cases} \quad (2.35)$$

where  $ot_{fot}^{1:|ot|}$  is the target side of the partial translation option  $o$  with a set of factors  $fot$ , and  $t_{ft}^{1:|t|}$  is the target phrase with factors  $ft$ . The partial translation option and target phrase are said to be *compatible* if they meet the above criteria and can be used to create subsequent partial translation options.

A subsequent *generation step* consults a factor-for-factor generation table for each target word in each partial translation option from the the previous mapping step. Each

factor in the previous partial translation can generate a set of output factors. The Cartesian product of the set of output factors creates a set of output strings. Each string is then merged with the preceding partial translation option. The output string and preceding partial translation option is also checked for compatibility, as Equation 2.35. The target phrase length is guaranteed to be identical but the overlapping factors from the output string and the preceding option must also be identical.

The generation step is extended to allow sets of input and output factors, rather than a single factor as described above. Figure 2.5 contains the pseudocode for creating output strings for a particular previous partial translation option by a generation step.

```
Require: previous translation option, generation step
size ← size of target phrase of prev trans op
for i = 1 to size do
    find {output factors } from gen step for input factorsi
    insert into matrixi ← {output factors }
end for
return matrix1 × matrix2... × matrixsize
```

Figure 2.5: Creating output string for a generation step

In our example, the generation step searches for the POS tags of each English target word:

```
white → { ADJ, VB }
house → { NN, VB ADJ }
home → { NN}
building → { NN, VB}
```

Merging the POS tags with the preceding partial translation option increases the number of translation options from three from the preceding mapping step to 12 translation options.

```
maison blanc → {
white|ADJ house|NN, white|ADJ house|VB, white|ADJ house|ADJ,
white|VB house|NN, white|VB house|VB, white|VB house|ADJ,
white|ADJ home|NN, white|VB home|NN,
white|ADJ building|NN, white|ADJ building|VB,
white|VB building|NN, white|VB building|VB }
```



This process continues sequentially for every mapping step and returns the last created set of translation options to be used in the search stage. The complete pseudocode for the algorithm to create translation options for each sentence is shown in Figure 2.6.

```

Require: source phrase  $s_{span}$ , mapping steps
partial translation option set  $P \leftarrow \{\}$ 
for all mapping steps do
  partial translation option set  $P' \leftarrow \{\}$ 
  if first step then
     $P' \leftarrow$  find phrase-pairs from translation step for  $s_{span}$ 
  else if translation step then
     $\Phi \leftarrow$  find phrase-pairs from translation step for  $s_{span}$ 
    for all trans opt  $\in P$  do
      for all phrase-pair  $\in \Phi$  do
        if trans opt and phrase-pair are compatible then
          new trans opt  $\leftarrow$  merge trans opt with phrase-pair
          insert new trans opt into  $P'$ 
        end if
      end for
    end for
  else if generation step then
    for all trans opt  $\in P$  do
      create output strings  $S'$  from trans opt for step {see Figure 2.5 }
      for all output string  $\in S'$  do
        if output string and trans opt is compatible then
          new trans opt  $\leftarrow$  merge trans opt with output string
          insert new trans opt into  $P'$ 
        end if
      end for
    end for
  end if
  prune  $P'$  if necessary
   $P \leftarrow P'$ 
end for
return  $P$ 

```

Figure 2.6: Creating translation options

### 2.3.2 Over-Creation of Factored Translation Options

The creation of factorized translations can have high computational complexity under some circumstances, both in time and memory requirements. The factored model differs from the standard phrase-based model in that the table pruning parameters in the phrase-based model define the maximum number of translation options per contiguous span. For example, a table pruning limit of 20 entails a maximum of 20 translation options per span.

However, in the factored model, the table pruning only applies to the table for each mapping step. The application of a series of mapping steps can produce an explosion of intermediate partial translation options and consumes large resources.

Given a reasonable table pruning limit of 20 phrase pairs from each translation step, a factorization of two translation steps can create a maximum of 400 translation options.

The explosion of translation options is even more pronounced with generation steps since the output string from the generation steps are themselves created by the Cartesian product of sets of factors. For example, with a table limit of 20 for the translation and generation steps, and a previous translation option with a 3 word target phrase, the generation step can create a maximum of  $20 \times 20^3 = 160,000$  translation options.

For a single preceding partial option with target phrase  $t$ , the maximum number of translation options created by a translation step  $r$  with table pruning limit  $L_r$  is  $O(L_r)$ . For a generation step, the maximum number of translation option created is  $O(L_r^{|t|})$ .

Intermediate translation options must be pruned for efficiency but this risks discarding intermediate options that are needed to create good complete translation options. As with the pruning of incomplete hypotheses, this can adversely affect translation quality.

## 2.4 Experiments

We conduct experiments that test the utility of the factored approach and the use of word-level linguistic information in phrase-based models. We concentrate on German to English translation, using part-of-speech and lemma factors, in addition to surface forms. Performance was measured with BLEU (Papineni et al., 2001) and focused manual evaluation. Only two linguistically motivated factors are compared and contrasted but the study is applicable to other factors, such as morphological tags.

We trained on the New Commentary 2007 corpus data that was released for the workshop on Statistical Machine Translation<sup>1</sup>, tuning on an out-of-domain set, testing on in-domain and out of domain. The out-of-domain data was from the Europarl corpus<sup>2</sup>. Table 2.1 gives more details on the datasets used.

		German	English	Corpus ID
Train	Sentences	52,184		news-commentary
	Words	1,105,665	1,070,646	
Tune	Sentences	2000		dev2006
Test (in-domain)	Sentences	1064		nc_test2007
Test (out-of-domain)	Sentences	2000		devtest2006

Table 2.1: Training, tuning, and test conditions

Word alignment was performed using GIZA++ (Och and Ney, 2003). The training procedures from the Moses toolkit (Koehn et al., 2007; Hoang and Koehn, 2008) was used and standard extraction heuristics was used throughout. The Brill Tagger (Brill, 1995) for English, and the LoPar Tagger (Schmidt and Schulte im Walde, 2000) for German, were used to create part-of-speech and lemma factors.

### 2.4.1 Source Factors

We used the standard non-factored phrase-based model as the baseline throughout and added POS tags and lemma source features. The baseline %BLEU score out and in-domain was 14.55 and 18.23, respectively, Table 2.2, Model (1).

In Model (2), source words were augmented with POS and lemma factors. The translation model probability becomes:

$$p_{TM}(t|s) = p(t_{surface} | s_{surface}, s_{lemma}, s_{POS}) \quad (2.36)$$

The motivation for this model is the belief that ambiguous source surface forms reduces translation accuracy in non-factored models because they conflate different meanings, or interpretations. Augmenting the source surface forms with lemma and POS tags disambiguate homographs, leading to more accurate translation models and overall better

<sup>1</sup><http://www.statmt.org/wmt07/>

<sup>2</sup><http://www.statmt.org/europarl/>

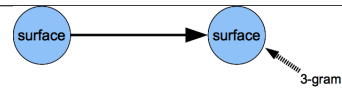
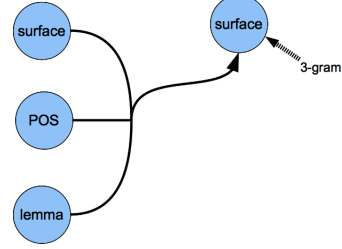
	Configuration	Out-of-domain	In-domain
1. Phrase-based baseline		14.55	18.23
2. Source factor model		<b>14.73 (+0.18)</b>	<b>18.78 (+0.55)</b>

Table 2.2: Source factors helps to disambiguate source words

translation performance. Table 2.2 Model (2) shows that this increases performance out and in-domain by 0.18% and 0.55%, respectively.

The affect of adding source factors were analyzed by observing the characteristics of the source sentences. The (in-domain) test set of the 1064 German sentences contains 6339 unique surface forms, see Table 2.3. Augmenting source words with lemma and POS tags increases the number of unique combined tokens by 366 as surface forms are disambiguated with different lemmas or POS tags.

Table 2.4 shows that the majority of source surface forms only have one interpretation but 27% of tokens in the test set have the same surface form but multiple lemma or POS tags.

Number of sentences	1,064
Number of surface tokens	26,898
Number of surface types	6,339
Number of combined surface/lemma/POS types	6,705

Table 2.3: Statistics for source language of (in-domain) test set

We examine surface forms with multiple interpretations to find out if augmenting surface forms with linguistic information aids in disambiguation.

Firstly, we examine the surface word *das* which has three interpretations: (1) as an article corresponding to 'the', (2) as a demonstrative pronoun 'this' or 'that', and, (3) as a relative pronoun 'which'. Each instance of the word and its local context was compared with the reference and manually evaluated for translation quality. We

<i>Surface types</i>	Unique types	%age	# occurrence in test set	%age
1 lemma/POS interpretation	6,002	95%	19,636	73%
2 lemma/POS interpretation	309	5%	3,980	15%
3 lemma/POS interpretation	27	0%	3,213	12%
4 lemma/POS interpretation	1	0%	69	0%
TOTAL	6,339	100%	26,898	100%

Table 2.4: Disambiguation of source test set

see a small improvement in the translation of the demonstrative pronoun and relative pronoun interpretations of the word, Table 2.5. These two interpretations are rarer in the test and training corpus than its use as an 'article'. Without the knowledge of the POS tags and lemma during training, the probability estimates for rare interpretations are subsumed into the same distribution as other interpretations of the surface word. The addition of source linguistic factors allows different meanings to be distinguished and more accurate statistics can be collected for each interpretation, especially for rare forms. Overall, using source factors has improved the translation of 7 of the 194 occurrences of the word *das*. Examples that were improved are shown below:

baseline: he saw *the* than the surest way

factored: he saw *this* as the surest way

reference: he saw it as his surest

baseline: *the* the use of force

factored: *that* the use of force

reference: which concerns the use of force

baseline: *the* justifies some

factored: *this* justifies some

reference: justifying some

In the second manual evaluation, the German word *sein* has two interpretations: (1) as a possessive pronoun 'his' or 'its' and, (2) as the infinitive of the verb 'to be'. Each occurrence was manually evaluated for translation quality and, again, we see that the addition of source factors improves the translation of rarer forms the most, Table 2.6.

Some of the improvements are due to the special case where the adjacent words are out-of-vocabulary, restricting the language model from using context to make a better

	article	dem. pronoun	relative pronoun	TOTAL
# occurrence in test set	132	33	27	194
Baseline	95 (72%)	17 (49%)	11 (41%)	123 (63%)
Factored	95 (72%)	20 (57%)	15 (56%)	130 (67%)

Table 2.5: Percentage of correct translation of *das*

lexical choice. But other examples show that source factors help to make better lexical choice even when the language model has adjacent context. However, we also see that the poor quality of translating the more frequent verb infinitive interpretation of the word has not significantly improved. The cause of poor quality for this interpretation is not poor lexical choice, but inadequate reordering. The German verb, especially if it is in the infinitive form, usually needs to be reordered from the end of the sentence to a position before the object noun in English.

baseline: innerpolitischen *be* rivals  
 factored: *his* innerpolitischen rivals  
 reference: *his* domestic opponents

baseline: which once *be* dream seemed to be  
 factored: that once *his* dream seemed to be  
 reference: that once seemed to be *his* dream

	verb infinitive	possessive pronoun	TOTAL
# occurrence in test set	49	14	63
Baseline	15 (31%)	5 (36%)	20 (32%)
Factored	16 (33%)	9 (64%)	25 (40%)

Table 2.6: Manual evaluation of translation of *sein*

Enriching the source language using factors has been also been taken up by Avramidis and Koehn (2008) who focus on augmenting nouns and verbs with syntactic information in a morphologically poor source language when translating to a morphologically rich language. Our experiments using POS tags and lemma information is a more general approach for a morphologically rich-to-poor language pair. Nevertheless, the disambiguation of source words still leads to a small improvement in translation.

## 2.4.2 Generating Target Factors

The aim of the models in this section is two-fold. Firstly, language models are an important component of the standard phrase-based models that improves output fluency and grammaticality of SMT system. Using sequence models over factors is an attempt to generalize language models by modelling sequences of linguistically motivated factors. Therefore, we would like to know if sequence models over non-surface factors can benefit translation, and if so, which factors. Secondly, if sequence models over factors are a useful resource in the factored model, what is the optimal method of creating target factors?

We test the utility of factorization using the generation step by generating target factors from the surface form. For ease of comparison, we stick to a decomposition which factorizes the translation model into a translation of the surface form, followed by a generation step which models all other target factors given the surface form.

All three factorizations below model the translation of the surface form in a single step. Each factorization then add target factors using a generation step which conditions on the target surface form created by the translation step.

$$\begin{aligned}
 p_{TM}(t|s) &= p(t_{POS}|t_{surface})p(t_{surface}|s_{surface}) \\
 p_{TM}(t|s) &= p(t_{lemma}|t_{surface})p(t_{surface}|s_{surface}) \\
 p_{TM}(t|s) &= p(t_{POS,lemma}|t_{surface})p(t_{surface}|s_{surface})
 \end{aligned}$$

Target POS and lemma factors were added using the factorization, above, in Models (3) to (5) of Table 2.7. The results from these models are compared with experiments using joint translation models to create the same target factors, Models (6) to (8). For each model, a trigram sequence model is attached to each target factor, trained on the target side of the training data.

The results show that sequence models over part-of-speech factors always improve translation quality, by as much as 0.5% BLEU in one model. However, creating lemma factors and using them in sequence models can decrease quality.

When comparing the performance of the factorized models (3) to (5) which jointly model the translation of all factors in one model, we see that the joint model is often the better choice. This result is also confirmed by Cettolo et al. (2008).

Models (4), which creates lemmas using a generation step, significantly underperform the baseline. The model can be reduced to approximately baseline model if the contribution of the generation and factor sequence models by setting their log-linear weights to zero.

	Configuration	Out-of-domain	In-domain
(1) Baseline	Non-factored baseline	14.55	18.23
3. Generating POS		<b>14.85 (+0.30)</b>	18.27 (+0.05)
4. Generating lemma		14.01 (-0.54)	17.72 (-0.51)
5. Generating POS & lemma		14.42 (-0.13)	18.39 (+0.16)
6. Translating POS		14.83 (+0.28)	<b>18.71 (+0.48)</b>
7. Translating lemma		14.75 (+0.20)	18.05 (-0.18)
8. Translating POS & lemma		14.73 (+0.18)	18.35 (+0.12)

Table 2.7: Generating target factors

However, when using *minimum error rate training (MERT)* (Och, 2003), which optimizes the log-linear weights to maximize BLEU, the weights for the generation step and factor sequence models are not zeroed out. This suggests that the addition of unhelpful factors or mapping steps can destabilize the MERT tuning algorithm to



converge on suboptimal weight settings. We will also see a repeat of this behaviour in the following sections. Models (5) and (7), which all use lemma factors, also show some decreased performance.

The better performance of non-factorized models over factorized models can be explained from the fact that using a non-factorized translation model avoids making the independence assumptions discussed in Section 2.1.2 and the use of the word-based generation step. Modelling of target factors conditioned by other target factors, is pursued in the following sections, as well by other studies such as (Yeniterzi and Oflazer, 2010). However, as a general-purpose method to create target factors, the use of generation step is inferior to a translation model which models all factors jointly.

### 2.4.3 Using Source and Target Factors

We see from Section 2.4.1 that augmenting the source language with POS tags disambiguates source surface forms and improves translation. We also see from Section 2.4.2 that translation models that jointly model the target surface and POS tags outperform models using lemma factors, or factorizing the translation model. A trigram sequence model on the POS factors was also used.

We combine the two approaches, modelling the translation of both source surface and POS tags to target source surface and POS tags, as illustrated in Figure 2.7. This combines the quality of both source and target factors to achieve performance better than either individually, Table 2.8, Model (9).

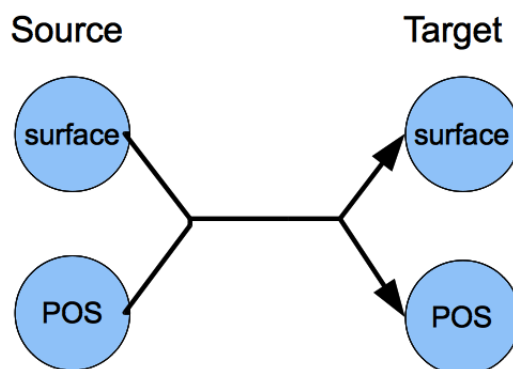


Figure 2.7: Joint translation model of surface and POS tags

	Configuration	Out-of-domain	In-domain
(1) Baseline	Non-factored baseline	14.55	18.23
(2) Source factor model	$p(t_{surface} s_{surface,POS,lemma})$	14.73	18.78
(6) Translating POS	$p(t_{surface,POS} s_{surface})$	14.83	18.71
9. Source & target POS	$p(t_{surface,POS} s_{surface,POS})$	<b>15.04 (+0.49)</b>	<b>18.84 (+0.61)</b>

Table 2.8: Translating source and target POS factors

## 2.4.4 N-Gram Sequence Models on Factors

Target POS and lemma factors were added in Section 2.4.2 in order to use sequence models on the factors. The conjecture is that sequence models generalize language models by modelling sequences of less sparsely, linguistically motivated factors derived from the surface forms. We saw that the use of POS factors improves the overall translation quality but using lemma often decrease performance. In this section, we investigate whether the relative sparsity of POS sequence models allows higher n-grams models to be effectively used.

The translation model of Section 2.4.3 was used as the foundation for the experiments in this section. We then varied the n-gram order of the POS sequence model, retuning and testing each time. Table 2.9, lines (10) to (14), shows small but consistent improvements for the in-domain test set with higher sequence model order. However, tests on out-of-domain data show little increase for sequence models with context over 4-grams and even some decreasing. This indicates that the higher order n-gram model fails to generalize for out-of-domain test sets.

## 2.4.5 Reducing Sparsity

A shortcoming of the standard phrase-based model is its poor handling of morphology. Each surface word form is treated as a separate token which is parameterized independently of its morphological variants. Conversely, the standard phrase-based model conflates parameterization of homographs. Section 2.4.1 described a factored approach for disambiguating homographs by augmenting surface forms with linguistically motivated factors. However, this approach increases the sparsity of the training data used to estimate the factored translation model.

Data sparsity can lead to issues with accurate translation. As an example, the News Commentary 2007 training corpus contain 18,827 examples of 1-word inflections of

	Configuration	Out-of-domain	In-domain
9	No POS LM	14.71	18.30
10	bigram POS LM	14.85 (+0.14)	18.59 (+0.29)
11	trigram POS LM	15.04 (+0.33)	18.84 (+0.54)
12	4-gram POS LM	<b>15.05 (+0.34)</b>	18.90 (+0.60)
13	5-gram POS LM	14.71 (+0)	18.94 (+0.64)
14	6-gram POS LM	15.02 (+0.32)	19.03 (+0.73)
15	7-gram POS LM	14.90 (+0.19)	<b>19.08 (+0.78)</b>
15	8-gram POS LM	14.84 (+0.13)	18.64 (+0.34)

Table 2.9: POS sequence model

the German word *sein* ('to be') unevenly distributed amongst its different inflections, Table 2.10. Inflections which are often seen in the training data have accurate probability estimates but inflections with a small number of examples can often have inaccurate statistics, as can be seen for the translation of *seid* ('are') which is seen twice in the training data:

$$p(\text{you are}|\textit{seid}) = 0.5$$

$$p(\text{you}|\textit{seid}) = 0.5$$

Nießen and Ney (2000, 2004) note that 40% of surface forms in their training corpus appear only once, so inaccurate estimations such as these can be widespread.

In the most extreme case, there is no translation for a surface form, such as for *warst*, *wart*, *wärest*, *wäret* in Table 2.10, but ample evidence for other inflections of the same lemma. In standard SMT models, there is no method which can extrapolate the translation of the often seen inflection to translate the unseen forms; most models either drop the unseen word or translate the word ad-verbatim.

This raises the issue of how can we use the knowledge that two words have a linguistic relationship to improve translation, such as a common lemma or stem. In the case where the surface form does not exist, how can we use this knowledge to extrapolate a translation for the surface form?

This question was studied by Nießen and Ney (2000, 2004) who created translation models for a hierarchy of equivalence classes of the surface form. The most coarse hierarchy consists of only the lemma, while the finest translation model jointly models the surface form and all the 'observational tuples' such as lemma and syntactic tags.

Word form	# example	Word form	# example
bin	132	bist	4
ist	9791	sind	4370
seid	2	war	2102
warst	-	waren	886
wart	-	sei	380
seien	112	seist	-
seiet	-	wäre	809
wärest	-	wären	239
		wäret	-

Table 2.10: Number of occurrences of 1-word conjugations of *sein*

More reliable statistics for word pairs were calculated by interpolating the hierarchy of translation models, leading to better translation.

In this section, we study the result of factorizing the translation model to reduce data sparsity of each mapping step, compared to that of a standard phrase-based model. We hope to improve parameter estimation in this way, resulting in better overall translation quality. We explore the factored approach model in (Koehn and Hoang, 2007) which takes as inspiration the analysis-synthesis approach common in rule-based MT systems such as (Carme Armentano-Oller et al., 2005). This is a shallow-transfer strategy for machine translation which analyses the input sentence into component linguistic properties. Linguistic properties are transferred to the target language, then the final target word forms are generated from the target linguistic properties. The intuition behind the approach is that by using translation models which separately translate lemma and POS tags, we take advantage of the reduced sparsity to improve the reliability of the translation model and increase translation quality.

Transforming surface forms to lemma reduces the number of unique types in the training data by 30% on the German side and 20% on the English side, Table 2.11. Part-of-speech tags and morphological information for the German source words are also created.

Koehn and Hoang (2007) reported that the model performs poorly when used on its own, but using the analysis-synthesis model in addition to a standard phrase-based translation model, positive results were achieved. We experiment with similar decompositions to seek an explanation for this. Again, we use the standard phrase-based

	German	English
Surface forms	68,069	37,254
Lemma	48,149	29,658
POS	54	45

Table 2.11: Number of unique tokens

model as the baseline, Table 2.12, Model (1). A trigram language model was used throughout, sequence models were not used on other factors.

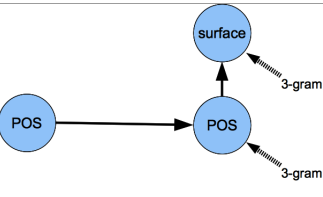
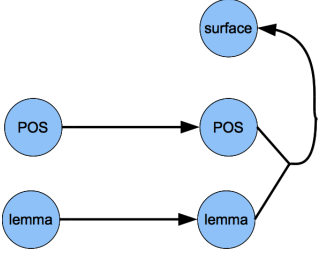
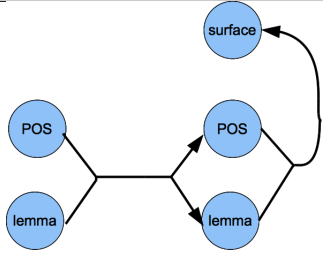
	Model	Out-of-domain	In-domain
1	Baseline (non-factored)	<b>14.6</b>	<b>18.2</b>
2		13.5 (-1.1)	17.3 (-0.9)
3		8.8 (-5.8)	11.3 (-6.9)
4	Model (3), using weights from (2)	13.5 (-1.1)	17.3 (-0.9)
5		13.8 (-0.8)	17.4 (-0.8)

Table 2.12: Decoding with decomposed model

Model (2) reduces the sparsity of the translation model by translating a less sparse factor, followed by a word-by-word generation step. However, this causes a decrease in BLEU but is still surprisingly good, considering the translation model decodes without the benefit of source linguistic information, either implied in the surface form or separated as a factor.

In Model (3), a limited amount of linguistic information is added back to the translation model by independently translating the POS and subsequently utilized by the generation step to model the surface form. However, rather than improving performance, it significantly worsens, caused by the interaction of the two translation steps which destabilizes the tuning procedure.

In experiment (4), the contribution of the POS translation step is ignored by zeroing out its feature function weights and reusing the tuning parameters from Model (2). Resulting in identical performance to that model.

In Model (5), we achieve better performance by jointly translating the same factors with only one translation step and using the same generation step. This achieved the best performance of all the models which reduces data sparsity. However, the results are still not able to match those of the standard phrase-based baseline.

### 2.4.6 Intermediate Translation Option Pruning

A reason for the poor performance of Model (3) is the method for creating factored translation options. Translation options are created by applying the series of mapping steps in sequence where the mapping steps must be arranged so that the factors for a mapping step have been created by the set of previous steps. In Model (3), the order of mapping steps is

1. Translate source to target lemma
2. Translate source to target POS tags
3. Generate target surface form from target lemma and POS tags

Recall from Section 2.3 that the creation of partial translation options involve Cartesian products which can multiply the number of partial translation options.

For example, the two translation steps in Model (3) do not have overlapping factors, many translation options are created, constrained only by the compatibility check on the phrase length. The subsequent application of the generation step intersects the output of both translation steps and drastically cuts down on the number of translation options. This is due to most of the partial translation options not satisfying the constraint of Equation 2.35 on overlapping target factors. Table 2.13 shows the magnitude of the problem. The problem is worse for short phrases where the number of intermediate translation options multiplies to 250 times the number of options from a standard phrase-based translation model, before being reduced to less than 5 times that of a single translation model.

# source word covered	Decode Step		
	Translate lemma	Translate POS	Generate
1	118,466	33,077,154 ( $\times 279$ )	445,255 ( $\times 3.8$ )
2	15,169	4,087,423 ( $\times 269$ )	66,542 ( $\times 4.4$ )
3	3,820	439,395 ( $\times 115$ )	11,289 ( $\times 3.0$ )
4	467	37,107 ( $\times 79$ )	1,338 ( $\times 2.9$ )
5	39	288 ( $\times 7$ )	69 ( $\times 1.8$ )
6	3	3 ( $\times 1$ )	3 ( $\times 1$ )
7	1	1 ( $\times 1$ )	1 ( $\times 1$ )

Table 2.13: Number of partial translation options for 100 input sentences

To cap memory usage, we use a pruning strategy that occurs after each mapping step during translation option creation. However, this is a risky pruning strategy which may discard partial translation options needed to create good completed translation options. Figure 2.8 gives the pseudocode for the better translation option algorithm which reduces the intermediate translation option bloat and the resulting excessive memory usage and pruning.

The algorithm still forms a cartesian product of the two translation steps but it processes each item in the first translation step individually rather than as a whole, containing the intermediate translation option explosion to a single entry in the first translation step. The each partial translation option from the first mapping step, in line 3 & 4, is processed to create completed translation options before another partial option is processed. Also, if pruning is required then it is done on the set of completed translation options. Therefore, there is no risk of discarding partial translation options required to create good completed options, as occurred in the original algorithm of Figure 2.6.

Replacing the translation option creation algorithm with this algorithm results in BLEU scores of 9.94% out-of-domain (12.5% in domain), above comparable results of Table 2.12 Model (3), but still well below simpler decomposition of Model (5), as well as the phrase-based baseline of Model (1).

**Require:** source phrase  $s_{span}$ , mapping steps

```

1: 1st translation options  $P \leftarrow \{\}$ 
2: final translation options  $Q \leftarrow \{\}$ 
3:  $P \leftarrow$  find phrase-pairs from 1st translation step for  $s_{span}$ 
4: for all 1st-trans-opt  $\in P$  do
5:     old translation options  $R \leftarrow \{\}$ 
6:     insert 1st-trans-opt into R
7:     for all mapping steps - 1st translation step do
8:         new translation options  $S \leftarrow \{\}$ 
9:         if translation step then
10:             $\Phi \leftarrow$  find phrase-pairs from translation step for  $s_{span}$ 
11:            for all old-trans-opt  $\in R$  do
12:                for all phrase-pair  $\in \Phi$  do
13:                    if old-trans-opt and phrase-pair are compatible then
14:                        new-trans-opt  $\leftarrow$  merge old-trans opt with phrase-pair
15:                        insert new-trans-opt into S
16:                    end if
17:                end for
18:            end for
19:        else if generation step then
20:            for all old-trans-opt  $\in R$  do
21:                 $\Phi \leftarrow$  create output strings from old-trans-opt for step
22:                for all output-string  $\in \Phi$  do
23:                    if output-string and old-trans opt are compatible then
24:                        new-trans-opt  $\leftarrow$  merge old-trans-opt with output-string
25:                        insert new-trans-opt into S
26:                    end if
27:                end for
28:            end for
29:        end if
30:         $R \leftarrow S$ 
31:    end for
32:     $Q \leftarrow R$ 
33:    prune Q if necessary
34: end for
35: return Q

```

Figure 2.8: Creating translation options 2



### 2.4.7 Out-of-Vocabulary Words

The decomposition of the translation model to less sparse mapping steps of the last section does not outperform a non-decomposed translation model which models all factors in one translation step. However, the reduction in sparsity can map lemmas of surface forms not seen in the training corpora. This makes it possible to gather at least some statistics of out-of-vocabulary words from their linguistic relationship with other words in the training corpus. We saw in the previous section that decomposing to less sparse features reduces performance in general but using it only for out-of-vocabulary words can help improve overall translation. This is the approach described in (Koehn and Hoang, 2007), its utility is diminished for low morphological languages and as more parallel data is available for a language pair and out-of-vocabulary (OOV) word becomes rarer. However, if lemmatization or stemming tools are available, this approach offers a convenient, fallback solution to dealing with this particular data sparsity problem of OOV words.

We repeat their experiment to analyze more indepth how the approach affects processing of OOV words. Once again, the baseline is the non-factored model of Table 2.12, Model (1). Both the baseline translation model and the decomposed lemma translation of Model (2) are used during decoding. However, the two phrase tables are not translation steps in one translation model but alternative translation models. During decoding, translation options from either model can be used for hypothesis expansion. Each model is weighted separately during tuning. Using this approach, we obtain a 0.7% BLEU gain over the baseline for in-domain translation but out-of-domain results are imperceptible, Table 2.14, Model (6).

	Model	Out-of-domain	In-domain
(1)	Baseline (non-factored)	14.6	18.2
6	Baseline + Model (2)	<b>14.7 (+0.1)</b>	<b>19.1 (+0.9)</b>

Table 2.14: Decoding with decomposed model and phrase-based model

The analysis of the test sets, Table 2.15 and 2.16, shows that by using the combination of both translation models, OOV rate was reduced by over 1% (20% relative reduction), in both in and out-of-domain, the majority of which are nouns and adjectives.

The first 100 OOV words in the in-domain and out-of-domain test sets were manu-

	Baseline	Baseline + Decomposed Model	Reduction
nn	2021	1815	-7%
adj	452	228	-7%
vvfin	126	21	-3%
vvinf	58	28	-1%
vvpp	57	19	-1%
vvizu	20	4	-1%
<b>TOTAL OOV</b>	<b>3048 (5.62%)</b>	<b>2424 (4.47%)</b>	<b>-20%</b>

Table 2.15: Number and rate of OOV words (Out-of-domain)

	Baseline	Baseline + Decomposed Model	Reduction
nn	813	742	-5%
adj	280	153	-9%
vvfin	72	16	-4%
vvpp	35	19	-1%
vvinf	33	14	-1%
<b>TOTAL OOV</b>	<b>1354 (5.03%)</b>	<b>1054 (3.92%)</b>	<b>-22%</b>

Table 2.16: # and rate of OOV words (In-domain)

	Out-of-Domain	In-Domain
Not translated	70%	68%
Incorrectly translated	25%	25%
Correctly translated	5%	7%

Table 2.17: Manual Evaluation of 100 OOV words

ally evaluated to see whether the factorized translation model helps. From Table 2.17, we see that 5% (out-of-domain, 7% in-domain) of words are correctly translated, while the majority of OOV words remain untranslated or are translated incorrectly.

## 2.5 Large Training Corpora

It is informative to compare the relative performance of the factored translation model when trained with more data. We therefore used the Europarl<sup>3</sup> corpora to train larger models for a German-English system. The training corpus contain 1,540,549 parallel sentences but once cleaned this was reduced by about 2% . We used in-domain, hold-out data for MERT tuning and tested on in and out-of-domain test sets. Table 2.18 gives more details on the datasets used.

		German	English	Corpus ID
Train	Sentences	1,509,017		Europarl v5
	Words	38,919,074	41,103,428	
Tune	Sentences	2000		dev2006
Test (out-of-domain)	Sentences	1057		nc_test2007 v2
Test (in-domain)	Sentences	2000		devtest2006

Table 2.18: Training, tuning, and test conditions

The translation models was trained on the parallel data described above, using standard phrase-based heuristics as implemented in the Moses toolkit. One difference is the use of Good-Turing frequency estimate for phrase probability calculation (Foster et al., 2006), whereas previous sections used the maximum likelihood estimates.

A trigram language model was trained on the 1,843,035 sentence of the target side of the corpus which also contains non-parallel sentences. POS sequence models were also created using tagged sequences of the same data. No lexicalized reordering models (Tillmann, 2004) were used.

As with the experiments in the previous sections, we use as a baseline a non-factored model. We use the most promising factored translation model from the last sections which jointly translate source POS and surface forms to their target-language counterparts. The POS tags for each language were produced by the same tools as the previous sections, namely the Brill Tagger for English and the LoPar Tagger for German.

As can be seen from Table 2.19, the availability of more training data has diminished the utility of better parameterization through linguistic information. The factored model with low-order sequence models shows worse performance than the baseline,

<sup>3</sup><http://www.statmt.org/europarl/>

but nevertheless, there is a small but consistent increase over the standard phrase-based baseline for the in-domain test set for higher order sequence models. This is consistent with the results we found with smaller training where we saw small gains when translating out-of-domain, and bigger gains with in-domain.

	Configuration	Out-of-domain	In-domain
1	Non-factored baseline	21.04	26.39
Factored model			
2	No POS LM	20.65 (-0.39)	26.22 (-0.17)
3	trigram POS LM	20.57 (-0.47)	26.40 (+0.01)
4	4-gram POS LM	20.77 (-0.27)	26.34 (-0.05)
5	5-gram POS LM	20.81 (-0.23)	26.53 (+0.14)
6	6-gram POS LM	<b>21.26 (+0.22)</b>	26.57 (+0.18)
7	7-gram POS LM	20.97 (-0.07)	26.60 (+0.21)
8	8-gram POS LM	21.00 (-0.04)	<b>26.63 (+0.24)</b>
9	9-gram POS LM	20.89 (-0.15)	<b>26.63 (+0.24)</b>

Table 2.19: Results when trained with Europarl corpus

## 2.6 Conclusion

From the experiments described in this chapter, we see that the following characteristics have helped improved translation quality:

1. ability to use source factors to disambiguate source words,
2. multiple language models, with longer n-gram order over non-surface factors,
3. ability to translate unknown words by backing off to lemma and part-of-speech tags.

We used two factors with different characteristics. Firstly, POS tags were used as factors for their syntactic analysis and because the small number of POS tags for each language reduces the data sparsity of models based on them. Morphological or semantic tags may also have similar ideal properties when used as factors. We also used lemma factors. While the determination of lemmas from surface form may be non-trivial, we do not believe they produce complementary information to the surface

form to be useful for decoding. The relative usefulness of the two factors is borne out by experimental results.

The results also show the limitation of factoring the translation model into multiple translation and generation steps. Factorization is helpful only to increase the translation coverage of unknown words but in most situations a translation which jointly decodes all factors is superior.



# Chapter 3

## Factored Template Model

Phrase-based translation can accurately translate words or small contiguous phrases by memorizing parts of the training data. We have also shown that adding linguistically motivated factors can improve translation by disambiguating source words or improving grammaticality by using high-order n-gram models over linguistic factors.

A major problem with SMT is reordering. In phrase-based SMT, the driver for reordering is the language model feature function which prefers fluent word order. The language model is a powerful and convenient model which can be created from cheap monolingual corpora. However, relying on the language model to dictate reordering is problematic since it reorders the output to resemble fragments of the training data without consideration of the input sentence. Neither can a language model based on surface forms generalize to words or phrases it has not seen, which is an especially important issue for highly inflected target languages.

Clear reordering patterns emerge from a POS-tagged corpus which can be used as a basis for improving reordering. For instance, the French-English News Commentary corpus<sup>1</sup> finds over 7000 instances of the source language NOUN ADJECTIVE phrase. The alignment data strongly suggest that this 2-word phrase translates to ADJECTIVE NOUN, i.e, it overwhelmingly suggests to swap the order of the two words (Table 3.1). Strong reordering patterns can also be seen in Table 3.2 for the longer POS tag sequence

NOM ADJ KON ADJ

In this chapter, we directly tackle short-range reordering in the phrase-based model by presenting an extension of the factored translation which makes use of the strong

---

<sup>1</sup><http://www.statmt.org/wmt07/shared-task.html>

Source	Target	Alignment	# examples	%
nom adj	jj nn	1-0 0-1	6704	29%
nom adj	jj nns	1-0 0-1	3244	14%
nom adj	nnp nnp	1-0 0-1	814	3%
nom adj	nn nn	1-0 0-1	558	2%
nom adj	nn	0-0 1-0	556	2%
nom adj	jj	1-0	483	2%
nom adj	nnp nn	1-0 0-1	376	2%
nom adj	nns	0-0 0-1	361	2%
Phrase pairs with less than 2%			10,198	44%
TOTAL			23,294	100%

Table 3.1: French–English translation of *nom adj*

Source	Target	Alignment	# examples	%
nom adj kon adj	jj cc jj nns	1-0 2-1 3-2 0-3	103	15%
nom adj kon adj	jj cc jj nn	1-0 2-1 3-2 0-3	98	14%
nom adj kon adj	jj cc jj	1-0 2-1 3-2	21	3%
nom adj kon adj	jj cc jj nns	3-0 2-1 1-2 0-3	18	3%
nom adj kon adj	jj cc jj nn	3-0 2-1 1-2 0-3	17	2%
Phrase pairs with less than 2%			435	63%
TOTAL			692	100%

Table 3.2: French–English translation of *nom adj kon adj*



ordering patterns of non-surface factors such as POS tags. We show that the proposed model outperforms the lexicalized reordering model (Tillmann, 2004) which is also focused on improving reordering in phrase-based models. In fact, the proposed model can completely replace the linguistically unmotivated distance-based reordering model and lead to overall better sentence translation.

In our tests, we obtained 1.0 increase in absolute BLEU for French-English translation, and 0.7 BLEU increase for German-English translation with the News Commentary corpora <sup>2</sup>.

### 3.1 Reordering in Phrase-Based Models

The phrase-based translation model carries out certain types of reordering adequately, outperforming more complex models such as the hierarchical phrase model when most of the reorderings in a particular language pair are reasonably short (Birch et al., 2009). Phrase-based models implicitly perform short-range reordering by memorizing multi-word phrase-pairs. However, this is not always possible when the phrase-pair does not exist in the training corpora. This issue is especially acute for small parallel corpora, highly inflectional languages, or out-of-domain test sets. In such cases, the phrase-based model resort to other models to inform reordering.

The simplest, distance-based reordering model (Brown et al., 1993) is based on the implicit assumption that most translations do not involve reordering, penalizing hypotheses that do reorder phrases. The penalty is proportional to the amount of non-monotonicity. For closely related languages such as English or Romance languages, the assumption of monotonicity is generally correct, hence the good performance of short-range reordering in phrase-based models compared to more complicated models.

However, the distance-based model is unlexicalized and not learnt from data, the same penalty applies regardless of the phrase being reordered. This ignores the fact that some words or phrases are more likely to be reordered than others. For example, adjectives in French are usually moved before the noun in a French-English translation:

*chat noir* → *black cat*

The reordering penalty is offset by the potential increase in the n-gram language model probability (Manning and Schütze, 1999). Therefore, the language model has a significant effect on reordering by preferring hypotheses which are more fluent. However,

---

<sup>2</sup> <http://www.statmt.org/wmt07/shared-task.html>

the language model is ill-equipped to model re-ordering as it has no information of the source sentence, or what constitutes good re-ordering. Therefore, using it to guide re-ordering conflates the separate objectives of improving fluency and correct reordering, risking an unsatisfactory solution for both objectives.

Another issue with using language models is data sparsity, even when trained on a large amount of data. This is especially true with highly inflection languages as lexical types are modelled separately. Data sparsity prevents the creation of high-order models so limiting the context window. Out-of-domain test data and rare words hinders the usefulness of language models further.

Lexicalized reordering (Tillmann, 2004) introduces a probability distribution for each phrase-pair that indicates the likelihood of being translated monotone, swapped, or placed discontinuous to its adjacent phrase. However, whether a phrase is reordered also depends on its neighbouring phrases, which the lexicalized reordering model does not take into account. For example, the French phrase *noir* would be reordered if preceded by a noun when translating into English, as in as in *chat noir*, but would remain in the same relative position when preceded by a conjunction such as *rouge et noir*.

The lexicalized reordering model also has similar problems to language models in that it does not generalize the lexicalized phrases with less sparse features and is, therefore, similarly prone to data sparsity issues.

### 3.1.1 POS-Based Reordering

The phrase-based approach of memorizing phrase-pairs works well for phrase-pairs it has seen in the training data. For example, the following (French) source phrase is correctly reordered because its translation occurs often in training:

*Union Européenne* → *European Union*

However, phrase-based models may not reorder even small two-word phrases if the source phrase is not in the training data. This situation worsens for longer phrases where the likelihood of the source phrase being previously seen is lower.

Although the surface string may not have occurred during training, the underlying POS tag sequence may have occurred many times. For example, if the surface string

*difficultés économiques et sociales*

does not occur in the training corpus, a phrase-based model will often incorrectly translate it by monotonically concatenating the translation of its words or subphrases:

**difficulties** *economic and social*

However, the training data may contain many similar phrases with the same underlying POS tags. The correct translation of the underlying POS tags of the source phrase can be extracted from the training data because it has been observed many times. Furthermore, the alignment information in the training corpus shows exactly how the individual words in this phrase should be ordered. The POS-based phrase-pair is shown below, with alignment information denoted by co-indexes:

$$\text{NOUN}_1 \text{ ADJ}_2 \text{ CONJ}_3 \text{ ADJ}_4 \rightarrow \text{ADJ}_2 \text{ CONJ}_3 \text{ ADJ}_4 \text{ NOUN}_1$$

The challenge addressed later on in this chapter is the use of part-of-speech phrase-pairs such as the one above in a phrase-based model to improve reordering.

The use of word-class ‘phrases’ and alignment information to inform reordering has precedent in early phrase-based models. The alignment template model (ATS) (Och and Ney, 2004) uses phrase-pairs composed of automatically learnt word classes for intraphrase reordering of lexical translations. A phrase-pair, called *alignment template*, is a triple  $(F_1', E_1', \tilde{A})$  that describes the alignment  $\tilde{A}$  between a source class sequence  $F_1'$  and a target class  $E_1'$ . The alignment guides the ordering of words within each template.

Tomas and Casacuberta (2003) extends the alignment template model by replacing automatic word classes on the source side with POS tags. In this model, a template is a tuple  $(F, R)$  where  $F$  is a sequence of source POS tags and  $R$  is a set of indexes which describes the ordering of each word position in  $F$ . The surface string translation continues to be a word-for-word correspondence.

Liberato et al. (2010) recast the use of POS templates as *phrase prototypes* in a phrase-based model. Prototypes are composed of source and target POS tags, translation rules based on prototypes are created by using a word-to-word dictionary to fill the surface string translation, based on the ordering dictated by the prototype. However, it differs from Tomas and Casacuberta (2003) in that the translation rules are created in isolation, rather than during decoding.

In all of the above research, the surface string is created with a word-to-word model, using a template to inform word ordering. Unaligned target positions in the templates that cannot be filled using the word-to-word model are often modelled with

surface forms in the template. For example, a possible template would be the following translation rule with an unaligned surface form  $de$

$$NN_1 NN_2 \rightarrow NN_2 de NN_1$$

Later phrase-based models such as Pharaoh (Koehn, 2004) and Moses (Koehn et al., 2007) combine templates and word translations to create phrasal translation rules. Word-class and alignment information in the translation rules are no longer needed during decoding and dispensed with for standard phrase-based decoding.

## 3.2 Translation Using Templates of Factors

A major motivation for the factored approach to machine translation is to use less sparse linguistic factors such as POS tags to generalize models. For example, we saw in Section 2.4.4 that the n-gram order of sequence models over different factors can be adjusted to take advantage of the relative sparsity of the factor. Higher n-gram orders are possible for sequence models of POS tags than of surface strings, leading to better target side grammaticality and translation.

Similarly, we would like to generalize the translation model with factors such as POS tags. Using the translation model factorization described in Section 2.4.5 we can decompose the translation model into multiple translation steps based on factors with differing sparsity. Long phrase-pairs can be extracted for translation steps based on less sparsified factors such as POS tags, while shorter phrase-pairs can be extracted on sparse factors such as surface strings.

For decoding, however, the factored model is formulated to constrain every translation step to identically segment the source and target phrase during decoding. This can be seen in Equation 2.18 in the previous chapter, which we reproduce below. (From hereon in, we will ignore generation steps  $\gamma$  as it is not relevant to future discussions).

$$\begin{aligned} p_{TM}(t|s) &= \prod_{r \in \tau} p(t_{out(r)} | s_{in(r)}) \times \prod_{g \in \gamma} p(t_{out(g)} | t_{in(g)}) \\ \prod_{r \in \tau} p(t_{out(r)} | s_{in(r)}) &= \prod_{r \in \tau} \prod_{b \in d} p_{TM}(t_{out(r)}^b | s_{in(tr)}^b) \\ (\gamma &= \emptyset) \end{aligned}$$

The translation model probability is the product of each translation step  $r \in \tau$ . In turn, the probability of each translation step is the product of a set of phrase-pair probabilities  $p_{TM}(t_{out(r)}^b | s_{in(tr)}^b)$ . Recall  $b = (start_s, end_s, start_t, end_t)$  is a coupled source and

target span.  $d = \{b^*\}$  is a set of  $b$  which covers all words in the source and target sentence once and only once. For the factored translation model,  $d$  is identical for all translation steps.

Within the log-linear formulation of SMT, the parameters of the translation steps included are feature functions in the log-linear model of Equation 3.1. Again, we invoke the maximum derivation approximation.

$$\begin{aligned} \arg \max_t p(t|s) &\approx \arg \max_t h(t, s, d) \\ h(t, s, d) &= \sum_{r \in \tau} \lambda_r \sum_{b \in d} h_r(t_{out(r)}^b, s_{in(r)}^b) + \sum_{m' \notin \tau} \lambda_{m'} h_{m'}(t, s, d) \end{aligned} \quad (3.1)$$

$h_r(t_{out(r)}^b, s_{in(r)}^b)$  is the feature function of translation step  $r$  for a phrase-pair spanning  $[start_s, end_s]$  on the source side and  $[start_t, end_t]$  on the target.  $\lambda_r$  is weighting of translation model  $r$ .  $h_{m'}(t, s, d)$  and  $\lambda_{m'}$  are the same for all other feature functions. The feature functions for translation steps are their log probabilities:

$$h_r(t_{out(r)}^b, s_{in(r)}^b) = \log p(t_{out(r)}^b | s_{in(r)}^b) \quad (3.2)$$

Simply speaking, every translation option contains exactly one phrase-pair from each translation step, which must be of the same source and target length. This simplifies the construction of translation options so that they can continue to be constructed as a preprocessing step prior to decoding, making the decoding algorithm identical to a standard non-factored algorithm. However, this constraint, which we term the *synchronous constraint*, prohibits decoding with short phrase-pairs of sparse factors together with long phrase-pairs of less sparse factors such as POS tags.

In this chapter, we present the *factored template model* which relaxes the synchronous constraint so that translation steps do not need to identically segment the source and target phrase. We will focus on factorizing the translation model into two translation steps. The first step models POS tags. The second translation step jointly models the POS tags and surface form.

For example, in translating the following phrase that has been pre-tagged with part-of-speech information.

$$\begin{bmatrix} \text{difficultés} \\ \text{NOUN} \end{bmatrix} \begin{bmatrix} \text{économiques} \\ \text{ADJ} \end{bmatrix} \begin{bmatrix} \text{et} \\ \text{CONJ} \end{bmatrix} \begin{bmatrix} \text{socials} \\ \text{ADJ} \end{bmatrix}$$

We would like the ability to use long POS phrase-pairs such as:

$$\text{NOUN}_1 \text{ ADJ}_2 \text{ CONJ}_3 \text{ ADJ}_4 \rightarrow \text{ADJ}_2 \text{ CONJ}_3 \text{ ADJ}_4 \text{ NOUN}_1$$

in conjunction with several phrase-pairs that jointly translate surface and POS tags:

$$\begin{aligned} & \begin{bmatrix} \text{difficultés} \\ \text{NOUN} \end{bmatrix}_1 \rightarrow \begin{bmatrix} \text{difficulties} \\ \text{NOUN} \end{bmatrix}_1 \\ & \begin{bmatrix} \text{économiques} \\ \text{ADJ} \end{bmatrix}_1 \begin{bmatrix} \text{et} \\ \text{CONJ} \end{bmatrix}_2 \begin{bmatrix} \text{socials} \\ \text{ADJ} \end{bmatrix}_3 \rightarrow \begin{bmatrix} \text{economic} \\ \text{ADJ} \end{bmatrix}_1 \begin{bmatrix} \text{and} \\ \text{CONJ} \end{bmatrix}_2 \begin{bmatrix} \text{social} \\ \text{ADJ} \end{bmatrix}_3 \end{aligned}$$

### 3.2.1 Factored Template Model

The *factored template model* is a factored phrase-based model where the translation model is decomposed into two translation steps:

1. translate POS tags
2. jointly translate POS tags and surface forms

The first translation step can be viewed as the *template* which guides reordering of phrase-pairs from the second step. Equation 3.3 is the decomposition of the translation model probability in the *standard factored model*.

$$\begin{aligned} p_{TM}(t|s) &= p(t_{surface,POS}|s_{surface,POS}) & (3.3) \\ &= p(t_{surface,POS}|s_{surface,POS})^\lambda p(t_{surface,POS}|s_{surface,POS})^{1-\lambda} \\ &\approx p(t_{surface,POS}|s_{surface,POS})^\lambda p(t_{POS}|s_{POS})^{1-\lambda} \end{aligned}$$

This formulation approximates the joint translation model with a translation model on the POS tags. It then interpolates both models to arrive at the translation probability,  $p_{TM}(t|s)$ .

A *phrase-pair* for a translation step  $r$  in the factored template model is defined as a triple  $(s_{in(r)}^b, t_{out(r)}^b, \Omega_r^b)$  where  $s_{in(r)}^b$  and  $t_{out(r)}^b$  are a sequence of words in the source and target, respectively, and  $\Omega_r^b$  is the alignment information which is retained from the phrase extraction stage. The structure of the alignment information will be described shortly. This definition of phrase-pair is identical to the templates of (Och and Ney, 2004). However, whereas (Och and Ney, 2004) uses word-based translation for surface forms, the factored template model uses phrase-pairs for both templates and surface form translation.

Therefore, the feature function in Equation 3.2 is expanded to include the alignment information:

$$h_r(t_{out(r)}^b, s_{in(r)}^b, \Omega_r^b) = \log p(t_{out(r)}^b, \Omega_r^b | s_{in(r)}^b) \quad (3.4)$$

The factored template model relaxes the constraint of Equation 3.1 so the segmentation used by step 1 and 2 are not required to be identical.

$$\hat{t} = \arg \max_t h(t, s) \quad (3.5)$$

$$\approx \arg \max_{t, d_1, d_2} h(t, s, d_1, d_2) \quad (3.6)$$

$$\begin{aligned} h(t, s, d_1, d_2) &= \lambda_1 h_1(t_{out(1)}, s_{in(1)}, \Omega_1, d_1) + \lambda_2 h_2(t_{out(2)}, s_{in(2)}, \Omega_2, d_2) \\ &\quad + \sum_{m' \notin \{1, 2\}} \lambda_{m'} h_{m'}(t, s, d_1, d_2) \\ &= \sum_{b_1 \in d_1} \lambda_1 h_1(t_{out(1)}^{b_1}, s_{in(1)}^{b_1}, \Omega_1^{b_1}) + \sum_{b_2 \in d_2} \lambda_2 h_2(t_{out(2)}^{b_2}, s_{in(2)}^{b_2}, \Omega_2^{b_2}) \\ &\quad + \sum_{m' \notin \{1, 2\}} \lambda_{m'} h_{m'}(t, s, d_1, d_2) \end{aligned}$$

where  $d_1$  and  $d_2$  are the segmentation for translation step 1 and 2, respectively.  $\Omega_r$  and  $\Omega_r^b$  are alignment information which will be defined shortly.

Again, we invoke the maximum derivation principle to approximate the maximum translation during optimization.

### 3.2.2 Constraints

The two translation steps in the factored template model are subject to three constraints.

Firstly, the segmentation of each translation step are no longer identical but they are not independent. Phrase-pairs from the first translation step must wholly contain a number of phrase-pairs of the second step. No phrase-pairs from the second translation step can straddle two or more phrase-pairs from the first. Formally:

*Template Constraint*

$$\begin{aligned} \forall b_2 &= (start_s^2, end_s^2, start_t^2, end_t^2) \quad (3.7) \\ \exists b_1 &= (start_s^1, end_s^1, start_t^1, end_t^1) \\ \text{such that} \quad &start_s^1 \leq start_s^2, end_s^1 \geq end_s^2, \\ &start_t^1 \leq start_t^2, end_t^1 \geq end_t^2 \\ \text{where} \quad &b_1 \in d_1, b_2 \in d_2 \end{aligned}$$

Note the definition of the coupled span  $b_r = (start_s^r, end_s^r, start_t^r, end_t^r)$  for a translation step has been expanded from the standard factored model of Equation 3.1 to clearly distinguish the segmentation of each translation step.

Secondly, reordering of phrase-pairs from the second translation step within the first translation step is constrained by the word alignment which is formally defined

now. An alignment  $(i, j)$  is a relationship which maps a position  $i$  in a phrase in the source language to a position  $j$  in its translation. For a phrase-pair or aligned sentence of source and target sentence  $s$  and  $t$ , respectively, the set of alignments  $\Omega$  is defined as

$$\Omega = \{(i, j) \mid 1 \leq i \leq |s|, 1 \leq j \leq |t|\} \quad (3.8)$$

Note that  $i$  and  $j$  are defined as indices over the entire input sentence  $s$  and  $t$ , not over the phrase pair.

The alignment can be visualised as a matrix of binary values. For example, the alignment set  $\{(1,2)(3,1)\}$  for a 3-word source and 2-word target sentence can be visualized as Figure 3.1.

		target	
		1	2
s o u r c e	1		
	2		
	3	T	

Figure 3.1: Example alignment matrix for 3-word source, 2-word target sentence

$\Omega_r$  are the alignments when  $s$  is translated to  $t$  by a translation step  $r$ .

The *alignment constraint* for the factored template model allow only translations where the intersection of the alignment set for each translation step,  $\Omega_1$  and  $\Omega_2$  meet the following criteria:

*Alignment Constraint*

$$\Omega = \Omega_1 \cap \Omega_2 \quad (3.9)$$

$$\exists(i, \bullet) \in \Omega \quad \forall 1 \leq i \leq |s|$$

$$\exists(\bullet, j) \in \Omega \quad \forall 1 \leq j \leq |t|$$

Simply speaking, both translation steps must agree on at least one alignment for every position in the source and target phrase.

The alignment information  $\Omega_r^b$  of each phrase pair  $(s_{in(r)}^b, t_{out(r)}^b, \Omega_r^b)$  are similarly defined as in Equation 3.8. Positions which remain unaligned are artificially aligned to every position in the other language for all phrase-pairs. This ensure that unaligned positions can be translated by both translation steps.



$\Omega_r$  is created by merging all  $\Omega_r^b$  from the phrase-pairs in  $r$  used to translate  $s$  to  $t$ .

$$\Omega_r = \bigcup \Omega_r^b, \quad \forall b \in d_r \quad (3.10)$$

Note that the alignment information in each phrase-pair is also defined relative to the entire source and target sentence.

$$\Omega_r^b = \{(i, j) \mid start_s \leq i \leq end_s, start_t \leq j \leq end_t\} \quad (3.11)$$

where  $b = (start_s, end_s, start_t, end_t)$ . However, the alignment information is estimated relative to each phrase-pair and shifted relative to  $s$  and  $t$  during decoding.

The third constraint inherits from the factored approach of overlapping target factors formalized in Equation 2.35. The two translation steps in the factored template model both emit POS tags in the target language. Hypotheses can only be formed if the POS tags emitted by each translation step are identical for all target positions. As with Equation 2.35, this can be formalized as a binary function:

$$f(t_{out(1)}^{b_1}, t_{out(2)}^{b_2}) = \begin{cases} 1 & (t_{out(1)}^{b_2} = t_{out(2)}^{b_2} \cap \{POS\}) \\ 0 & otherwise \end{cases} \quad (3.12)$$

### 3.2.3 Example of Constraints

We show an example of the construction of a template phrase-pair and the alignment consistency check below.

Table 3.3 shows phrase-pairs from the first and second translation steps of a factored template model.

The first translation step contains only one phrase-pair, therefore, its alignment information comes directly from the phrase-pair. However, the phrase-pair contains only two alignment points, (1, 2) and (3, 1), shown in solid lines; the second source word is unaligned and is therefore artificially aligned to both positions in the target (dashed lines). Therefore,

$$\Omega_1 = \{(1, 2)(3, 1)(2, 1)(2, 2)\}$$

Similarly, the first phrase-pair in translation step 2 also has an unaligned word which is artificially aligned to the target. The alignment information for the second translation step,  $\Omega_2$ , is calculated by taking the union of the alignment of all its phrase-

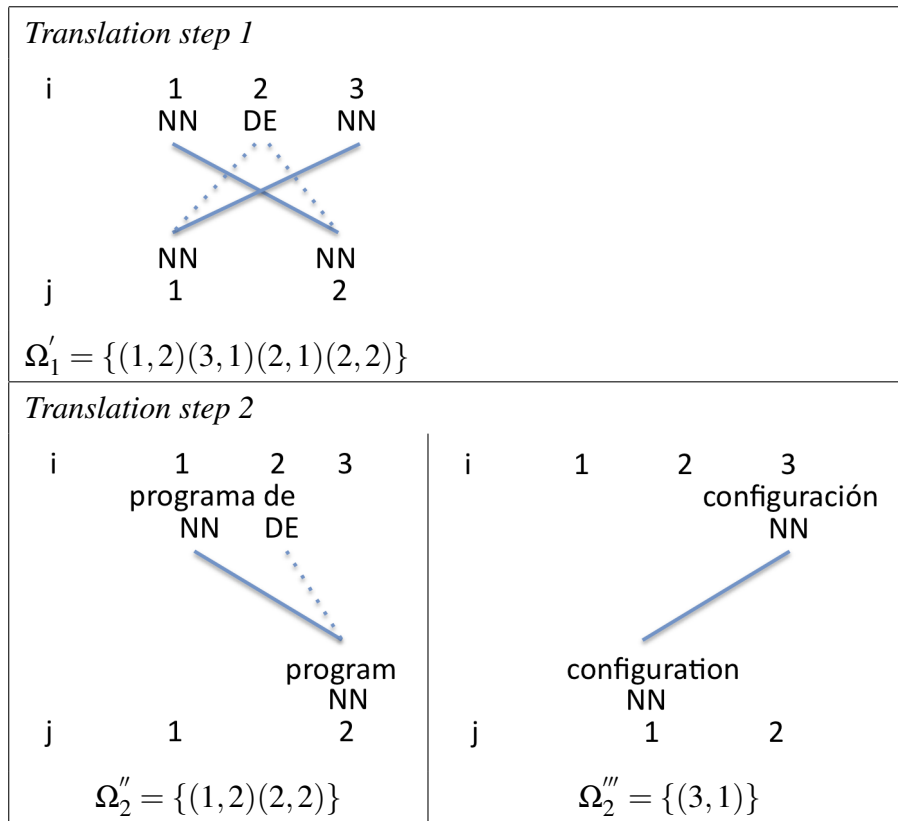


Table 3.3: Example phrase-pairs with alignment information

pairs.

$$\begin{aligned}
 \Omega_2 &= \Omega''_2 \cup \Omega'''_2 \\
 &= \{(1,2)(2,2)\} \cup \{(3,1)\} \\
 &= \{(1,2)(2,2)(3,1)\}
 \end{aligned}$$

The intersection of the alignments from both translation steps,  $\Omega$ , contains an alignment for every source and target position.

$$\begin{aligned}
 \Omega &= \Omega_1 \cap \Omega_2 \\
 &= \{(1,2)(2,2)(3,1)\}
 \end{aligned}$$

Therefore, the combined phrase-pair created by merging all three phrase-pairs is valid as it satisfies the alignment constraint. The alignments from the combined phrase-pair is pictured in Figure 3.2.

Notice that the surface forms translated by the second translation step have been reordered. Also, the surface form ‘de’ in the source sentence is translated by a phrase-pair which also translates an adjacent word. This is possible as the unaligned word is

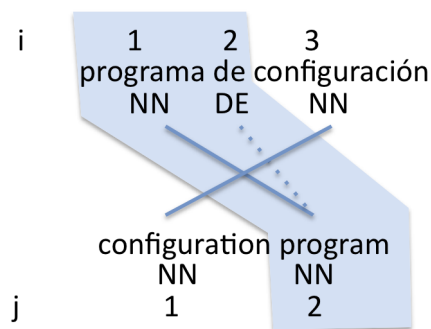


Figure 3.2: Combined phrase-pair

artificially aligned to all words in the phrase pair, thereby allowing them to be translated by adjacent phrases.

### 3.3 Decoding

In phrase-based decoding, a translation option strictly contains one phrase pair. The factored model described in Chapter 2 allows the the translation model to be decomposed into multiple translation and generation steps. However, each step in the factored model does not decode independently, but is limited by constraints discussed in the previous chapter. These include a constraint that all translation steps identically segment the translation.

The constraint allows the factored decomposition to be implemented as a preprocessing step of decoding, before the search process. Specifically, the construction of translation options is extended from the standard phrase-based model by using a phrase-pair from each translation step, and one generation-pair for each target word, to create a translation option. Once the translation options are created, the search process can proceed unchanged from standard phrase-based decoding.

Translation options are the intermediate representation between phrase-pairs and generation-pairs, and the hypotheses in the stack decoder. Translation options relate to a specific source span of a sentence and are applied to hypotheses to create new hypotheses.

Extending the creation of translation options while keeping the search process unchanged is also the approach that was followed for the *factored template model*. Each translation option is created from a single phrase-pair from the first translation step and multiple phrase-pairs from a second translation step. Therefore, the construction

of translation options is the core implementation of the factored template model and which we will describe below.

### 3.3.1 Creating Translation Options

A set of translation options is created for every continuous source span. The creation of translation options for each span is independent of other source spans.

The creation of translation options for a continuous source span is modelled as a search process similar to a phrase-based model. Formally, for every source subphrase,  $s$ , the model searches for the best translation  $\hat{t}$ , subject to the constraints described previously. Translation options are composed of one phrase-pair,  $(s_1, t_1, \Omega_1)$ , from the first translation step and multiple phrase-pairs,  $(s^{b_2}, t_2^{b_2}, \Omega_2^{b_2})$ , from the second translation step.

$$\begin{aligned}
 \hat{t} &= \arg \max_t h(s, t) & (3.13) \\
 &\approx \arg \max_{t, d_2} h(s, t, d_2) \\
 &\approx \arg \max_{t_1, d_2} \left\{ \lambda_1 h_1(s_1, t_1) + \lambda_2 h_2(s_2, t_2) + \sum_{m \notin \{1, 2\}} \lambda_m h_m(s, t) \right\} \\
 &\approx \arg \max_{t_1, b_2 \in d_2} \left\{ \lambda_1 h_1(s_1, t_1) + \right. \\
 &\quad \left. \lambda_2 \sum_{b_2 \in d_2} h_2(s_2^{b_2}, t_2^{b_2}) + \sum_{m \notin \{1, 2\}} \lambda_m h_m(s, t) \right\}
 \end{aligned}$$

where  $d_2$  is the set of source and target spans,  $b_2 \in d_2$ , which covers every word in the source and target position,  $s$  and  $t$ , respectively, once and only once.

The search process is extended to search for the best set of translation options in the same way as the search for n-best translations in a phrase-based model.

The search is implemented as a simplified phrase-based decoder which concatenates phrase-pairs from the second translation step. This *intra-phrase* decoder simplifies a standard phrase-based decoder in a number of ways.

Firstly, partial translation options are evaluated using only local features, non-local features are evaluated only once completed translation options are formed. This enables fast decoding at the expense of search errors due to stack pruning. Pruning parameters are set wide to minimize any pruning but prevent excessive resource usage.

Secondly, phrase-pairs from the second translation step that are reordered within a template do not incur distortion cost. No distortion limits are set for reordering within

a template. The only constraints are those described in Section 3.2.2.

Once all source words are covered, all target words in the template must have also been covered. A translation option is said to be *completed* when all source and target positions of the template have been covered. Translation options which have not covered all target words in the template are discarded.

Also, the intra-phrase decoder returns only the completed translation options with the least number of phrase-pairs from the second translation step. For example, if translation options can be created using only one phrase-pair from the second translation step, then only these translation options will be returned. This aims to promote the use of longer phrase-pairs in templates and reduces memory and time resource.

The outline of the algorithm is described in Figure 3.3.

## 3.4 Training

The training procedure is identical to the factored phrase-based training described in (Koehn and Hoang, 2007). The phrase model retains the word alignment information found during training. Where multiple alignment exists in the training data for a particular phrase pair, the most frequent is used. This is consistent with the treatment of alignment used in the calculation of the lexicalized probabilities.

## 3.5 Experiments

Experiments were performed with the News Commentary corpus<sup>3</sup> which contains 60,000 parallel sentences for German–English and 43,000 sentences for French–English. Tuning was done on a 2000 sentence subset of the Europarl corpus (Koehn, 2005) and tested on a 2000 sentence Europarl test set (out-of-domain), and 1064 news commentary sentences (in-domain).

The training corpus is aligned using GIZA++ (Och and Ney, 2003). To create POS tag translation models, the surface forms on both source and target language training data are replaced with POS tags before phrases are extracted. The taggers used were the Brill Tagger (Brill, 1995) for English, the Treetagger for French (Schmid, 1994), and the LoPar Tagger (Schmidt and Schulte im Walde, 2000) for German. The training script from the Moses toolkit (Koehn et al., 2007; Hoang and Koehn, 2008) was used,

---

<sup>3</sup><http://www.statmt.org/wmt07/shared-task.html>

**Require:** source subphrase  $s$ , 1st and 2nd translation steps

```

{”consult translation steps”}
 $P_1 \leftarrow \{\}, P_2 \leftarrow \{\}$ 
 $P_1 \leftarrow$  phrase-pairs from 1st trans step for  $s$ 
for all subspan of  $s$  do
     $P_2 \leftarrow$  phrase-pairs from 2nd trans step for  $s^{subspan}$ 
end for
{”decode”}
Insert  $stack_0 \leftarrow$  empty trans-opt
for  $i = 0$  to  $|s|$  do
    for all trans-opt in  $stack_i$  do
        for all phrase-pairs  $\in P_2$  do
            Expand new-trans-opt  $\leftarrow$  trans-opt + phrase-pair
            if new-trans-opt is consistent with at least phrase-pair in  $P_1$  then
                 $C \leftarrow$  coverage of new-trans-opt
                insert  $stack_{|C|} \leftarrow$  new-trans-opt
            end if
        end for
    end for
end for
{”completed translation options”}
discard incomplete trans-opts in  $stack_{|s|}$ 
 $TP_2 \leftarrow \arg \min_{t \in stack_{|s|}} segments(t)$ 
return  $TP_2 \times$  consistent phrase-pair in  $P_1$ 

```

Figure 3.3: Creating Template Translation Options

extended to enable alignment information for each phrase pair. The vanilla Moses MERT tuning script was used throughout.

Results are also presented for models trained on the larger Europarl corpora<sup>4</sup>.

### 3.5.1 German-English Results

The traditional, *non-factored* phrase-based model was used as a baseline which obtained a BLEU score of 14.6% on the out-of-domain test set and 18.2% on the in-

<sup>4</sup> <http://www.statmt.org/europarl/>

domain test set (see Table 3.4, line 1) for German-English.

A second baseline is a *joint* model (Chapter 2) in which POS tags for both source and target languages are augmented to the training corpus and used in decoding with an additional trigram POS sequence-model. Using the joint model increased translation performance (line 2). This model has the same input and output factors, and the same language models, as the following factored template models therefore offers a fairer comparison than the non-factored baseline.

#	Model	out-domain	in-domain
1	Unfactored Model	14.6	18.2
2	Factored Model (FM)	15.0 (+0.4)	<b>18.8 (+0.6)</b>
3	Factored template Model (FTM)	13.3 (-1.3)	16.1 (-2.1)
4	FM + FTM	<b>15.3 (+0.7)</b>	<b>18.8 (+0.6)</b>

Table 3.4: German–English results, in %BLEU

Decoding purely with the *factored template* model (line 3) results in a significant performance degradation. However, when the factored template model is used in combination with the factored model which jointly translate all factors (line 4), this outperforms the baseline on the out-of-domain test set. We will refer to this as the *combination model*.

When using the combination model, the factored template model is further constrained to create translation options that consist of a minimum of two phrase-pairs from the second translation step. This avoids the duplication and allows translation options from the factored template model to complement those of the factored model.

The results illustrate the utility of using the factored template model as an additional model to the factored model.

However, we believe the language pair German–English is not particularly suited for the factored template approach as many of the short-range ordering properties of German and English are similar.

### 3.5.2 French-English Results

Repeating the same experiments for French–English produces bigger gains by using the combination model with gains of 1.0 BLEU (1.1 in-domain) over the unfactored baseline and 0.8 (1.2 in-domain) over the factored model, Table 3.5.

#	Model	out-domain	in-domain
1	Unfactored Model	19.6	23.1
2	FM	19.8 (+0.2)	23.0 (-0.1)
3	FM + FTM	<b>20.6 (+1.0)</b>	<b>24.2 (+1.1)</b>

Table 3.5: French–English results, in %BLEU

### 3.5.3 Maximum Size of Templates

The maximum phrase length of seven words was used to extract phrase-pair templates for the first translation step in the factored template model. The phrase table for this translation step was retrained with varying maximum template lengths. Decreasing the maximum length made little difference, Table 3.6. However, increasing the maximum template length caused performance to drop dramatically. This results suggested that the pruning parameters which limit resource usage during factored template translation option creation has an adverse effect for long templates.

Maximum template length	out-domain	in-domain
5	20.6	<b>24.2</b>
7	<b>20.6 (+0.4)</b>	<b>24.2 (+0.0)</b>
8	<b>20.7 (+0.1)</b>	<b>24.1(-0.1)</b>
9	15.1 (-5.5)	17.5 (-6.7)
10	7.0 (-13.6)	7.7 (-16.5)

Table 3.6: Varying the maximum template lengths and effect on translation quality

### 3.5.4 Lexicalized Reordering Models

There has been considerable effort to improve reordering in phrase-based systems. One of the most well known is the lexicalized reordering model (Tillmann, 2004) which calculates the probability that a phrase is reordered from the word alignment information. The reordering is relative to the source or target phrase, and to the next or previous phrase-pair.

The lexicalized reordering model was trained and tested on the same data as the factored template model above to compare their performance, Table 3.7. A lexicalized



reordering model over the surface word and the part-of-speech tags were evaluated and were found to improve on the baseline joint model. However, using a combination model outperform both of these reordering models.

#	Model	out-domain	in-domain
1	FM (baseline)	19.8	23.1
2	FM + FTM	<b>20.6 (+0.8)</b>	<b>24.2 (+1.1)</b>
3	FM + Lexicalized Reordering	20.2 (+0.4)	24.1 (+1.1)
4	FM + Lexicalized Reordering on POS	20.3 (+0.5)	24.0 (+0.9)

Table 3.7: French–English results for lexicalized reordering model

## 3.6 Analysis

### 3.6.1 Use of Factored Template Model during Decoding

We analyzed the usage of translation options during the decoding of the out-of-domain French-English test set from the previous section. From Table 3.8, the translation options from the factored template model account for only 11% of the total number of phrase pairs used. However, of those 11%, nearly half of those translation options involve some re-ordering of the internal phrases.

The reordering of phrases from the joint translation model within a template does not incur a distortion penalty. Therefore, using the factored template model, in conjunction with a standard factored model offers the decoder an alternative, linguistically motivated method of explaining distortion to the simple linear distortion model of the standard phrase-based model.

We see from Figure 3.4 when the decoder is given this alternative, it overwhelmingly prefers to account for reordering with the factored template model. The normal phrasal reordering is almost completely absent during decoding and, in fact, allowing only monotonic translation is sufficient to obtain the same translation performance.

Table 3.9 compares the source phrase lengths of the translation options used by the baseline factored model and the combination model. A noticeable effect of using the combination model is that the average length of the source segmentation *decreases* as more words are translated singularly. However, the combination model

	# of trans opt used		Re-ordered	
Factored model	33,785	89%		
Factored template model				
2 internal phrases	2,890	7%	1,239	43%
3 internal phrases	1,001	3%	406	41%
4 internal phrases	327	1%	201	61%
5 internal phrases	69	0%	54	78%
6 internal phrases	15	0%	15	100%
7 internal phrases	2	0%	2	100%
<b>TOTAL</b>	<b>4260</b>	<b>11%</b>	<b>1917</b>	<b>45%</b>

Table 3.8: Number of translation options to decode out-of-domain test set, French–English

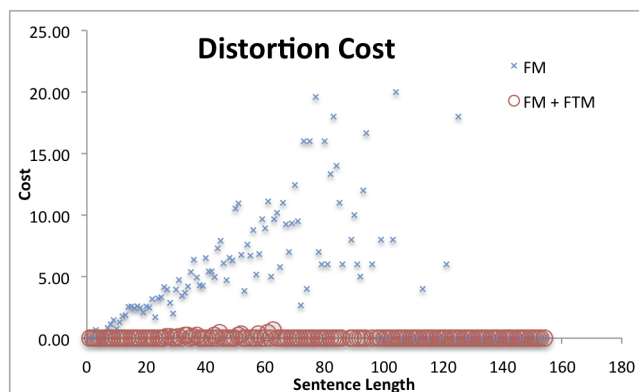


Figure 3.4: Average distortion cost per sentence

uses longer translation options more often, such as those which translate four or more source words.

To reiterate, the combination model decodes with translation options from the factored template model in addition to the joint factored model. This creates an ideal environment to study the relative utility of each translation model which competes for application during decoding. Of the 4,260 factored template translation options used in Table 3.9, only 8 were used in preference to a translation option from the joint model.

In the other 4252 cases, template translation options are used when a translation option from the factored model does not exist. This decoding pattern was also seen when we experimented with alternative translation models in Section 2.4.7 for

Segment length	Factored Model		Factored Model + Factored Template Model					
			Factored Model		Factored Template		Total	
1	21,191	56%	25,825	76%			25,825	68%
2	10,750	28%	4,912	15%	975	23%	5,887	15%
3	4,495	12%	2,132	6%	815	19%	2,947	8%
4	1,233	3%	650	2%	924	22%	1,574	4%
5	261	1%	171	1%	722	17%	893	2%
6	91	0%	73	0%	504	12%	577	2%
7	14	0%	22	0%	320	8%	342	1%
TOTAL	38,035	100%	33,785	100%	4,260	100%	38,045	100%

Table 3.9: Size of source segments decoding out-of-domain French–English

out-of-vocabulary words of the previous chapter. In that case, the linguistically informed translation model was only used in specific circumstances, when the standard phrase-based translation model was unable to propose any translation options for out-of-vocabulary words. The same is true of combination model; the factored template model is rarely used unless translation options from the factored model do not exist.

### 3.6.2 POS Sequences Used

Table 3.10 lists the top 10 source POS tag sequences where template translation options were used.

We see that the factored template model is applied most often to translate the source POS sequence, NOUN ADJ. This is consistent with the fact that NOUN ADJ phrases occur often with different surface forms of NOUN and ADJ. We saw in Table 3.1 that it is frequently translated by swapping the order of the translated words. However, unless the surface phrase has been seen in training, the standard phrase-based model will often incorrectly translate the phrase by monotonically concatenating the lexical translation. In contrast, the factored template model can guide reordering better and is therefore used in preference to the joint model.

The source POS sequence, NOM PUN NOM, is interesting as it is mostly used to translate noun phrases which have been mistokenized and mistagged. For example, the French noun phrase, *l'article* exists in the training data as the following tokenization and POS sequence

	Template	# times used		Example
1	nom adj	655	17%	rapport général → general report
2	nom pun nom	58	1%	l' article → the article
3	prp:det nom adj	57	1%	des règles démocratiques → democratic rules
4	det:art nom adj	49	1%	la différenciation culturelle → cultural differentiation
5	pun ver:pres	39	1%	' est → ' is
6	nom prp nom	35	1%	lieux travail → work places
7	nom adj adj	34	1%	produits alimentaires meilleurs → better food products
8	pro:per ver:infi	33	1%	les rejoindre → join them
9	nom ver:pper	28	1%	croissance soutenue → sustained growth
10	ver:pres adv	28	1%	offrent aussi → also provide
Others		3244	76%	
TOTAL		4,260	100%	

Table 3.10: Top 10 POS tag sequence translated by factored template phrases

l' (DET:ART) article(NOM)

but occurs twice in the test set with a different tokenization and tag sequence. The factored template model allows the system to recover from incorrectly tokenized phrases.

l (NOM) ' (PUN) article(NOM)

Other templates in Table 3.10 concurs with our understanding of translating French to English.

Table 3.10 also shows that usage of the factored template model is dispersed over many different POS sequences. The distribution of use of the factored template model follows a Zipfian distribution, in fact, 35% of the POS sequences are translated by a factored template option only once, Figure 3.5.

It is not just the usage of the factored template model that follows a Zipfian distribution. Translations for a particular source POS sequence also follows a similar distribution. For example, as we saw in Table 3.1 and 3.2 at the beginning of this chapter, rare phrase-pairs that translate NOUN ADJECTIVE account for 44% seen in the corpus. For translations of NOM ADJ KON ADJ, this account for 62%. (Here, rare phrase-pairs are defined as those that account for less than 2%). In both cases, there are a significant amount of phrase-pairs with just one example in the corpus. The phrase-pair distribution of these POS sequences can be seen in Figure 3.6.

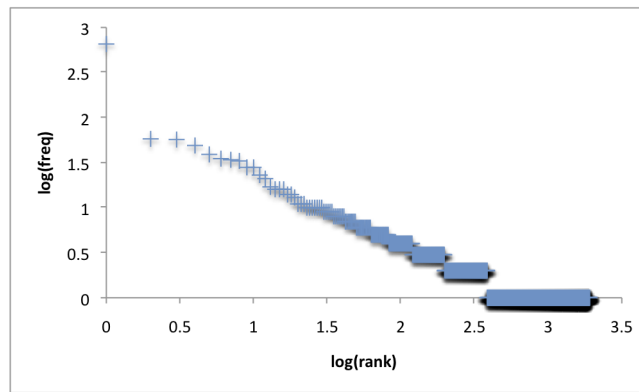


Figure 3.5: Zipfian distribution of usage of factored template model

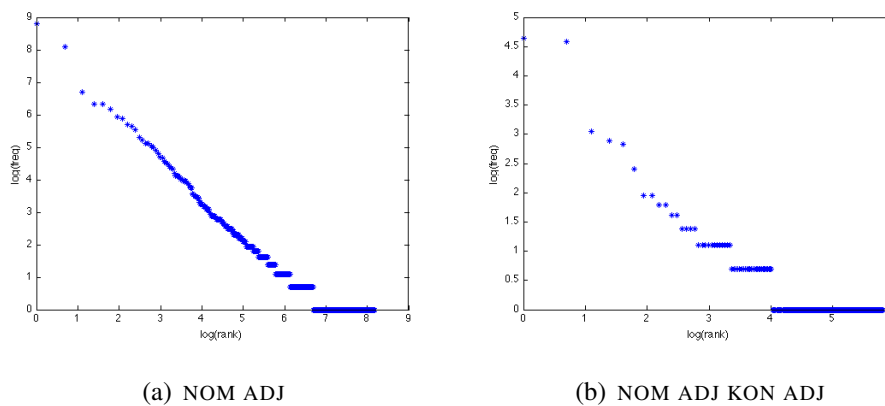


Figure 3.6: Zipfian distribution: # examples against rank of phrase-pair (log-log scale)

The Zipfian distribution of POS phrase-pairs limits the generalizability of such phrase-pair as much of the translation probability is accounted for in rare phrase-pairs.

The Zipfian distribution of POS phrase-pairs also account for the degraded performance when using the factored template model alone. The phrase table, including the POS template phrase table, must be pruned for efficient decoding but this will also discard many of the phrase-pairs needed to construct good translation options. In the factored template model, the POS phrase-table effectively filters out reorderings which do not much match one of the top ranked POS phrase-pairs. However, the Zipfian distribution means that this filtering will discard much good reordering which cannot be accounted for by POS sequences alone.

Therefore, a solution that we have explored is the combination model which uses POS phrase-pairs but can also bypass it with alternative translation options from the joint model. As we have seen, the combination model combines the advantages of both

models to produce better translation results.

### 3.6.3 Manual Evaluation

We manually evaluated the translations of two source phrases with POS sequences that standard phrase-based models often fail to reorder between French and English. As we discussed at the beginning of the chapter, the following POS sequence is often incorrectly translated due to reordering if it has not been seen in the training data even though the POS sequence will have been seen many times and will strongly suggest a reordering.

NOUN ADJ → ADJ NOUN

A random sample of source phrases containing this POS sequence were analyzed. The translation of each phrase was judged on coherency and correctness according to the reference.

The baseline factored model correctly reorders 58% of those phrases. Adding a lexicalized reordering model or the factored template model significantly improves the reordering to above 70%, Figure 3.7.

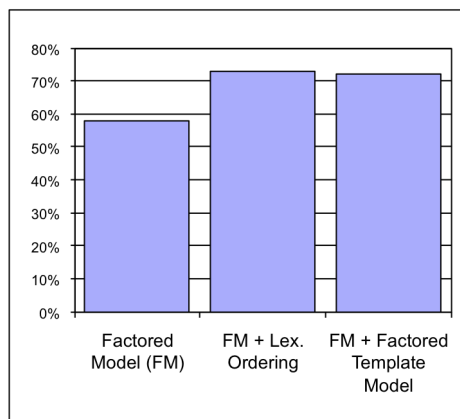


Figure 3.7: Percentage of correctly ordered NOUN ADJ phrases (100 samples)

A more challenging phrase to translate, such as

NOUN ADJ CONJ ADJ → ADJ CONJ ADJ NOUN

was judged in the same way and the results show a dramatic improvement of the combination of the factored model and factored template model over the factored model alone. This combination model also outperforms the factored model with lexicalized reordering (Figure 3.8).

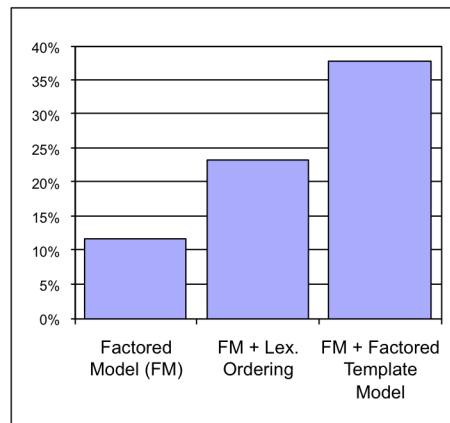


Figure 3.8: Percentage of correctly ordered NOUN ADJ CONJ ADJ phrases (69 samples)

### 3.6.4 Larger training corpora

It is informative to compare the performance of the factored template model when trained with more data. We therefore train on the larger Europarl corpora and tune the models for French to English translation. The BLEU scores are shown in Table 3.11, showing no significant advantage to adding POS tags or using the factored template model. This result is similar to many others which have shown that the large amounts of additional data diminishes the improvements from better models.

#	Model	out-domain	in-domain
1	Unfactored Model	31.8	32.2
2	Factored Model (FM)	31.6	32.0
3	FM + Factored Template Model	31.7	32.2

Table 3.11: French–English results, trained on Europarl corpus

## 3.7 Conclusion

We have shown the limitations of the current factored decoding model which restrict the use of long phrase translations of less-parsed factors. This negates the effectiveness of decomposing the translation process, dragging down translation quality.

An extension to the factored model was implemented which showed that using POS tag translations to create templates for surface word translations can create longer phrase translation and lead to higher performance, dependent on language pair.

For French–English translation, we obtained a 1.1 BLEU increase on the out-of-domain and in-domain test sets, over the non-factored baseline.



# Chapter 4

## Mixed-Syntax Translation

So far, we have defined our factored representation only over words. Using this word representation to hold linguistically motivated annotation, we have sought to disambiguate source words and improve grammaticality of the target language output.

In Chapter 3 on factored templates, templates of POS tags were also used to improve reordering. We showed how a linguistically-motivated reordering constraint, in the form of factored templates, can be used in combination with distance-based constraints. We also see that the use of factored templates negates the need for distance-based constraints and almost completely replaces distance-based reordering during decoding. Even when phrasal reordering is limited only to factored templates, the performance of the model was a significant improvement over the standard phrase-based model.

However, we also saw that the factored template model on its own produces poor performance, and that it must be used in combination with a standard factored phrase-based model to achieve competitive results.

We also experimented in Chapter 2 with an analysis-generation translation model which analyses the source, transfers the linguistic properties, and reconstitutes these properties in the target language. This also resulted in significantly decreased performance.

Nevertheless, the use of linguistic information to improve translation is an appealing concept, and one that should work. In both of the examples mentioned, attempting to rigidly impose a linguistic transfer mechanism results in decreased performance, but combining them with a non-linguistically motivated phrase-based model allows the combination to use the strength of both.

However, the word-level representation of the previous chapters limits the potential

of using linguistic annotation to a small window. With more training data, existing models such as language models and translation models become better parameterized, diminishing the gains that can be achieved from better, linguistically-motivated models acting in a small window.

In this chapter, we take the lessons learnt from the factored phrase-based and factored template approaches to expand from a word level to a phrase-level linguistic annotation of the source language. We transition from a phrase-based model to a hierarchical phrase-based, or syntactic model. These models extend the notion of a phrase from a sequence of words to a sequence of words and *subphrases*. Subphrases are represented by non-terminals, which can be *undecorated* in hierarchical models (Chiang, 2005), or *decorated* in syntax models (Marcu et al., 2006; Ambati and Lavie, 2008). This approach offers a natural fit with multi-word linguistic annotation that we are interested in studying.

By using multi-word annotation, we hope to better model phenomena that occur over a larger window, such as long-distance reordering. Syntactic structures such as parse trees offer a simpler explanation of reordering, for example, reordering an SOV language to an SVO language.

This chapter presents a novel tree-to-string model, the *mixed-syntax model*, which combines the specificity of syntactic models and the generality of the hierarchical phrase-based model. The aim of the model is two-fold. Firstly, the model can make use of syntactic information but also permits linguistically unmotivated mappings. Secondly, we take advantage of the specificity of syntactic constraints to permit different rule forms when compared to the hierarchical model but still maintain efficient training and decoding.

## 4.1 Past Work

### 4.1.1 Overview

Translation rules in the standard phrase-based model consist of a continuous sequence of words (*phrase*) in the source language and its corresponding translation in the target language.

The hierarchical phrase-based model extends the phrase-based model by allowing *hierarchical phrases* that contain subphrases. Translation rules are comprised of a source and target hierarchical phrase, and a one-to-one mapping between their sub-

phrases. Subphrases are represented in translation rules as non-terminal symbols. This model has certain advantages over the standard phrase-based model. Firstly, hierarchical rules can express the reordering of subphrases from the position of non-terminals in the source and target phrase. Hierarchical models promise better reordering as the reordering rules are lexicalized and an implicit part of the translation model. This contrasts with the phrase-based model where reordering is modelled separately from translation.

Secondly, hierarchical rules can model the translation of discontinuous phrases, such as the French, *ne...pas*. Thirdly, hierarchical rules follow the recursive structure of the sentence, reflecting the linguistic notion of language.

The rules in hierarchical phrase-based models are examples of a *synchronous context-free-grammar (SCFG)*. The general form of a SCFG rule is two linked CFG rules, shown below. The source rewrite rule parses the input while the target rule simultaneously outputs the translation.

$$\langle A \rightarrow \alpha , B \rightarrow \beta, \sim \rangle \quad (4.1)$$

where  $A$  and  $B$  are non-terminals in the source and target language, respectively,  $\alpha$  and  $\beta$  are strings of terminals and non-terminals, and  $\sim$  is a one-to-one mapping of non-terminals in  $\alpha$  and  $\beta$ .

Note that this notation, due to Satta and Peserico (2005); Huang et al. (2009), is slightly different from the one used by Chiang (2005). It is more flexible in the sense that it allows different symbols to be synchronized, which is essential to capture the syntactic divergences between languages, however, it can be reduced to Chiang (2005) notation simply by combining the linked source and target non-terminal symbols, eg.  $X = (A, B)$ .

$$X \rightarrow \langle \alpha , \beta , \sim \rangle \quad (4.2)$$

The hierarchical phrase-based model simplifies the SCFG by using only two non-terminal symbols,  $X$  and  $S$ . Translation rules extracted from the grammar all use the label  $X$ . The label  $S$  is used by ‘glue’ rules to monotonically concatenate partial derivations of the source sentence, ensuring that a translation can be produced for every input sentence.

The most common algorithm for decoding with SCFG is currently CKY+ (Chapelier et al., 1998) with cube pruning (Huang and Chiang, 2007), as described in Chiang (2007) and implemented in Chiang (2005); Li et al. (2009); Hoang et al. (2009). This can be used for hierarchical and some syntactic models.

Our focus in this chapter is on improving translation with a tree-to-string model using a synchronous CFG. However, we shall continue by discussing syntactic models in general as many of the techniques and arguments are applicable to any syntactic formalism.

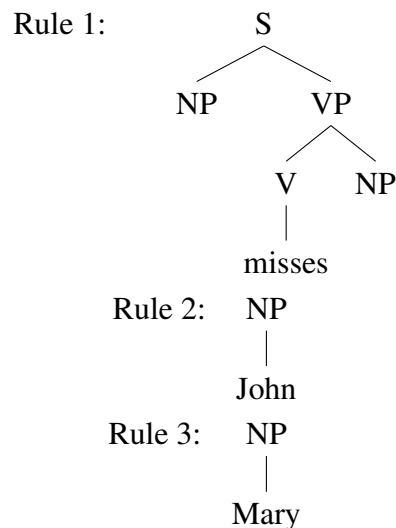
### 4.1.2 Synchronous Tree-Substitution Grammar

A more powerful formalism than SCFG which has been applied to machine translation is synchronous tree-substitution grammar (STSG) (Eisner, 2003) which expresses rewrite rules as multi-level structures. This goes some way to alleviating the non-isomorphism problem between languages. Chiang (2006) gives a good introduction to STSG which we will summarize.

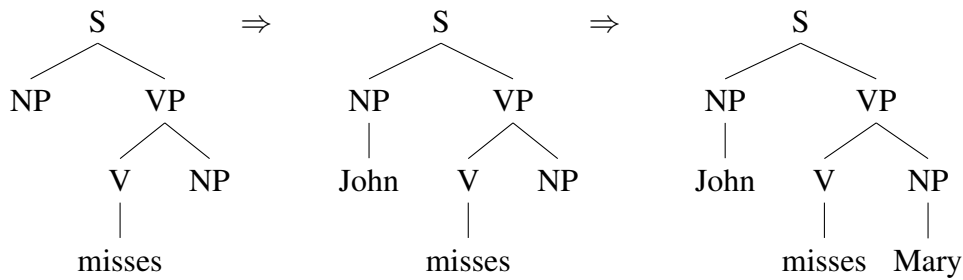
We start by explaining the (monolingual) tree-substitution grammar (TSG). Rules in a TSG are tree fragments whose leaf nodes are terminals or non-terminals. To parse a sentence with a TSG, we start with a tree fragment rooted at the start symbol and repeatedly choose a leaf non-terminal *NT* to which a tree fragment rooted in *NT* can be attached. The parse completes when the leaf nodes of the tree correspond to the words in the sentence and there are no leaf non-terminals. For example, to parse the sentence

John misses Mary

with the following TSG rules:



The parsing method can proceed as follows:

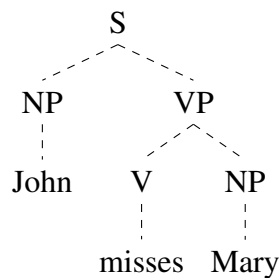


*Synchronous TSG* extends the monolingual TSG in which rules are a pair of tree-fragments where the leaf non-terminals are linked. The source side of the STSG analyzes the input while the target side simultaneously constructs a target derivation.

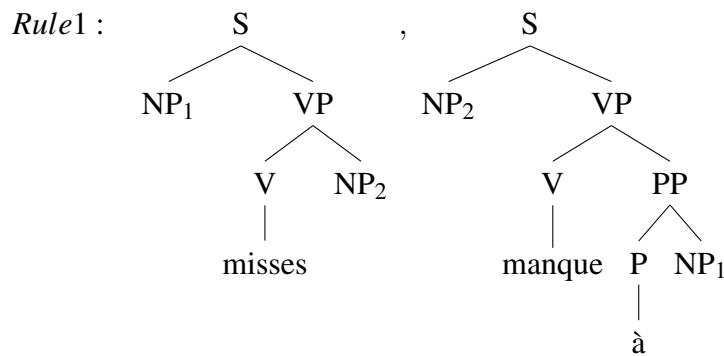
In tree-to-string and tree-to-tree models such as those described in Huang et al. (2006b), the input is a parsed sentence. Translation involves recursively converting the input parse tree into a string or tree of the target language.

For example, the tree below is a parse of the previous example sentence:

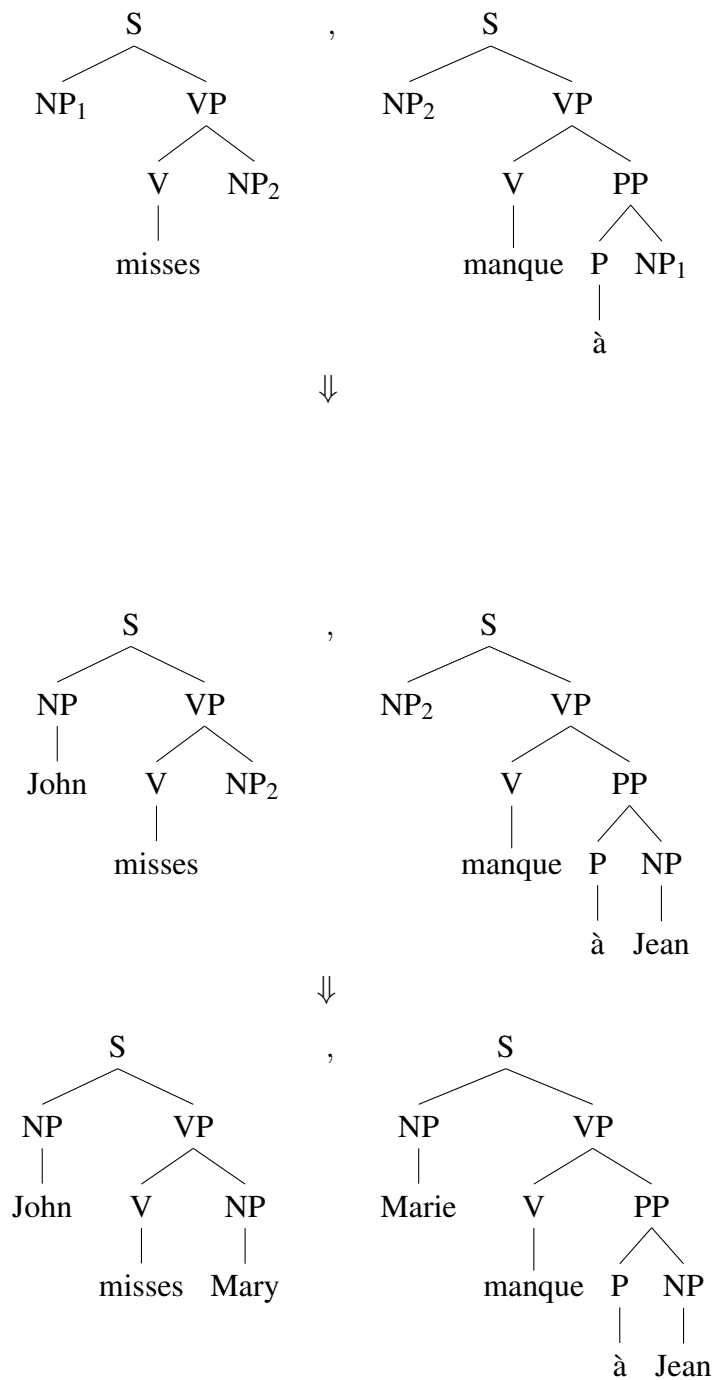
*Input Parse*



This parsed sentence can be converted to a target tree using the following STSG:



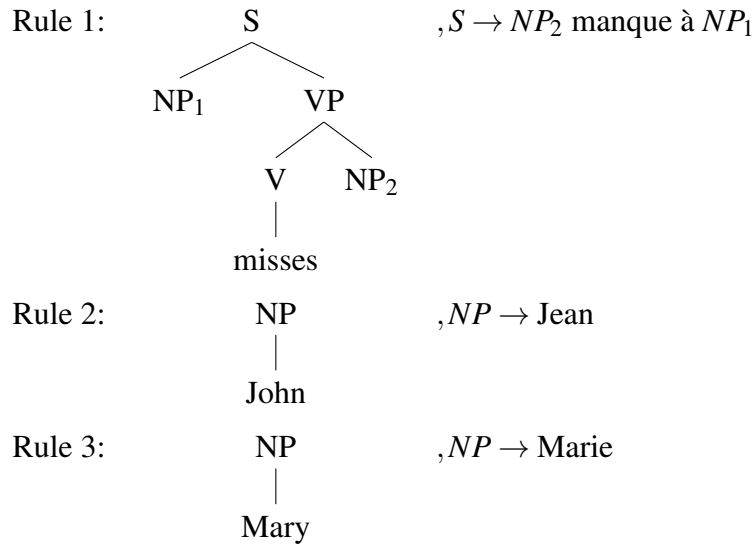
The decoding algorithm converts fragments of the source parse tree into a target tree:



Notice that the use of STSG has freed us from creating isomorphic derivations, as with a SCFG.

The tree-to-string models of Huang et al. (2006a); Liu et al. (2006, 2007); Mi et al. (2008); Mi and Huang (2008) follow Galley et al. (2004) by using *tree-transducers* which are equivalent to a tree substitution grammar on the source side and a CFG on the target side. The CFG is created by collapsing the tree-fragment of the target TSG.

The STSG rules in the previous example can be written as a tree-transducer grammar:



The tree-to-string translation models of Huang et al. (2006a); Liu et al. (2006) take as input a parse tree and apply tree-transducer rules until all terminals and non-terminals in the parse tree have been covered. The result of decoding is a source derivation which matches the input parse, and the target string.

The tree-transducer approach can be approximated by flattening both the source and target side of STSG rules to create SCFG rules with labelled source non-terminals. During decoding, the non-terminals are constrained to match the syntactic labels for the span in the input parse tree. This is roughly equivalent to decoding with a STSG where the leaf non-terminals and root nodes of a tree-fragment match a part of the input parse tree, but the interior nodes of the tree-fragment are ignored.

The above rules can be converted to a SCFG formalism of Equation 4.1.

$$\langle S \rightarrow NP_1 \text{ misses } NP_2 \quad , \quad S \rightarrow NP_2 \text{ manque à } NP_1 \rangle$$

$$\langle NP \rightarrow \text{John} \quad , \quad NP \rightarrow \text{Jean} \rangle$$

$$\langle NP \rightarrow \text{Mary} \quad , \quad NP \rightarrow \text{Marie} \rangle$$

If we wish to make explicit a tree-to-string model, target non-terminals symbols can be replaced with an uninformative label  $X$ :

$$\langle S \rightarrow NP_1 \text{ misses } NP_2 \quad , \quad X \rightarrow X_2 \text{ manque à } X_1 \rangle$$

$$\langle NP \rightarrow \text{John} \quad , \quad X \rightarrow \text{Jean} \rangle$$

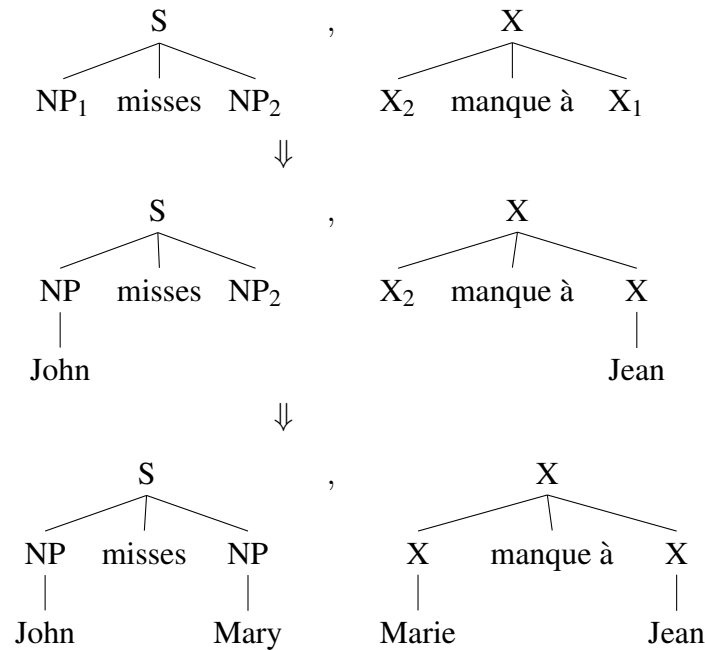
$$\langle NP \rightarrow \text{Mary} \quad , \quad X \rightarrow \text{Marie} \rangle$$

The result of decoding is a target string and source derivation which match the input parse, or a ‘flattened’ version of the parse. Therefore, decoding with this grammar

may not include all of the intermediate nodes of the source parse tree. However, our primary goal is translation, not reproducing the source parse tree.

The same parsed sentence above can be decoded with this synchronous CFG grammar.

### *Decoding*



### 4.1.3 Syntax Models

The use of syntax in statistical machine translation has a long history. Many of the formalisms have been studied and returned to many times by different researchers. Chiang (2010) listed the main syntax approaches which we reproduce in Table 4.1. The table also lists some of the early contributors, roughly before the introduction of phrase-based models by Zens et al. (2002); Koehn et al. (2003); Och and Ney (2004), and later contributors after its introduction. In the following section, we shall review these and other research into syntactic machine translation.

Wu (1997) introduced the stochastic inversion transduction grammar (ITG) which restricts rules with non-terminals to two configurations. An ITG is equivalent to a SCFG of rank two where the two permitted configurations for hierarchical phrases are unlexicalized rules with two non-terminals, linked so that their subphrases are concate-



Formalism	Early research	Later research
String-to-string	ITG (Wu, 1997)	Hiero (Chiang, 2005)
String-to-Tree	(Yamada and Knight, 2001)	(Galley et al., 2004, 2006)
Tree-to-String		(Huang et al., 2006a; Liu et al., 2006)
Tree-to-Tree	(Poutsma, 2000; Eisner, 2003)	(Lavie, 2008)
		(Zhang et al., 2008a)
		(Liu et al., 2009a)

Table 4.1: Syntax Formalisms

nated or swapped. These can be written using the familiar SCFG notation:

$$\langle X \rightarrow X_1 X_2, X \rightarrow X_1 X_2 \rangle$$

$$\langle X \rightarrow X_1 X_2, X \rightarrow X_2 X_1 \rangle$$

The hierarchical phrase-based model (Chiang, 2005) extends the phrase-based model by allowing phrases containing subphrases in the translation rule. In contrast to the ITG model, it does not limit ordering to only two possibilities but instead learns lexicalized reordering rules from the parallel corpus. The ITG and hierarchical models are classified as string-to-string models because they do not use syntactic information.

#### 4.1.4 String-to-Tree Models

The aim of generating syntactic trees is to improve output grammaticality by constraining output derivations to grammatically correct parse trees according to the target syntax.

Yamada and Knight (2001, 2002) studied the use of English parse trees for Japanese-English translation which typically exhibit large scale reordering due to the SVO and SOV characteristics of English and Japanese respectively. Lexical translation is performed with a word-based and phrase-based translation model while syntactic translation is performed through a series of structural operations which create and manipulate the target parse tree. The performance exceeded IBM Model 5 but the training data used was very small.

Ambati and Lavie (2008) tested a string-to-tree and tree-to-tree model on the large Europarl corpus (Koehn, 2002) and showed that the lack of coverage adversely affects

the recall of the extracted translation model. The resulting translations are significantly worse than those of hierarchical or phrase-based models.

Galley et al. (2004) describes an extraction procedure which creates a set of minimal translation rules to explain a parallel sentence pair. This work was extended in Galley et al. (2006) which creates larger rules composed of multiple minimal rules. However, the reported performance of string-to-tree models with composed rules were still significantly worse than the ATS phrase-based model of Och and Ney (2004).

As Wang et al. (2010) pointed out, it would be surprising if the parse structures and word alignments were perfectly suited for syntax-based SMT out-of-the-box. The objectives of the parsing model do not necessarily concur with the objectives of machine translation, therefore, to successfully exploit syntactic information it is necessary to use those parts which are useful to SMT, and alter or overcome those parts which are not. This insight is a reason why string-to-tree models have shown great improvements in recent years. Many such models consistently outperform phrase-based and hierarchical phrase-based models, at least for some language pairs.

Zollmann and Venugopal (2006) manages to create a competitive string-to-tree model by generating artificial constituents of a parsed sentence by merging the syntactic labels of adjacent, parent and child constituents, which they called *syntax augment machine translation (SAMT)*. This improves the coverage of the syntactic translation model as more translation rules can be extracted where non-terminals do not exactly span a target constituent. The non-terminals are then labeled with complex SAMT labels. The cost of this model is a high number of SAMT labels, resulting in data sparsity issues as well as an increase in resource utilization.

The issue of data sparsity using SAMT is tackled in (Venugopal et al., 2009) which maps all SAMT labels to a single hierarchical style non-terminal. The syntactic labels are used to calculate feature scores to softly constrain derivations.

*Binarization* of SCFG-based translation rules (Zhang et al., 2006; Wang et al., 2007; Huang, 2007; Xiao et al., 2009; DeNero et al., 2009) reduces the rank of the grammar to two to simplify and speed up decoding. However, the language generated from the binarized grammar is equivalent, or weakly equivalent, to the original grammar. This contrasts with the approach described by Wang et al. (2010) which restructures and relabels the parse trees using binarization methods to create a better grammar for machine translation.

### 4.1.5 Tree-to-String Models

Tree-to-string models take as input a parse tree of the source sentence which is then recursively converted into a string of the target language. These models suffer from similar coverage problems and lack of generality as string-to-tree models. However, Huang (2008) evaluated a tree-to-string model trained on a small dataset for a Chinese-English system and reported better performance than phrase-based models.

Zhang et al. (2008b) introduces the *synchronous tree sequence substitution grammar* (STSSG) based on an extension to the synchronous tree-substitution grammar (STSG) described in Section 4.1.2. The model differs from STSG-based models in that translation rules are composed of multiple tree fragments. This gives the model more expressive power by allowing it to capture non-syntactic phrases, leading to better translation. They show that the STSSG is a generalization of STSG, SCFG, and tree-transducer models.

An alternative to rigidly converting the source parse structure is to use the parse of the source sentence as the basis for feature functions to softly constrain the derivation during decoding. This approach provides better coverage as the translation model can use the source syntactic structure but is not constrained by it.

Marton and Resnik (2008) add feature functions to penalize or reward non-terminals which cross constituent boundaries of the source sentence. This follows on from unsuccessful earlier work by Chiang (2005) but this time they see gains when used with finer grain feature functions with different constituency types. The weights for feature function are tuned in batches due to the deficiency of the Minimum Error Rate Training when presented with many features.

Chiang et al. (2008) rectify the deficiency of using MERT by replacing it with MIRA (Crammer et al., 2006) to tune all feature function weights in combination. Chiang et al. (2009) add thousands of linguistically-motivated features to hierarchical and syntax systems, including source syntax features derived from the research above.

Shen et al. (2009) discuss soft syntax constraints and context features in a dependency tree translation model. The POS tag of the target head word is used as a soft constraint when applying rules. Also, a source context language model and a dependency language model are used as features.

An issue which affects all syntactic models is parse errors. In string-to-tree models, this adversely impacts the quality of the translation model. In tree-to-string models, parse errors of the input sentence can also propagate to translation errors. A solution to

input parse errors is to postpone the disambiguation of the best parse by providing the machine translation training and decoding systems with many possible parses generated by the parser. The obvious method to accomplish this is the use of an n-best list of possible parses. During extraction, translation rules are constrained with non-terminals from each of the n-best parses. During decoding, the decoder is free to choose which input parse to convert to a target string.

An extension of this method uses a parse forest which compactly encode many parses within an efficient structure. It was found that using an n-best list of parses achieved better results than using the 1-best parse. However, only by using a forest of translations, during both training and decoding, does the performance exceed that of hierarchical models. Tree-to-string models using packed forest input are described by Huang et al. (2006a); Liu et al. (2006, 2007); Mi et al. (2008); Mi and Huang (2008).

#### **4.1.6 Tree-to-Tree Models**

Early work by Poutsma (2000) and Eisner (2003) describes theoretical foundations and prototype implementation for such a model but the training data size was too small to judge the results against current SMT systems.

As can be expected, the naïve implementation of tree-to-tree models which doubly constrain extraction and decoding with both source and target parses reduces coverage and performance even more than other syntax models. This result has been found by Ambati and Lavie (2008), amongst others.

To increase coverage, the same techniques such as using packed forest for source sentences instead of 1-best parse can be applied. This was studied by Liu et al. (2009a) who reported good performance for a Chinese-English system trained on small data but no significant gain when used with large training data.

Zhang et al. (2008a) uses tree sequences which allows tree fragments and phrases in translation rules.

Chiang (2010) continued the line of research from (Chiang et al., 2009) by adding both source and target syntactic feature functions into the log-linear framework to soft-constrain the model.

#### **4.1.7 Summary of Past Work**

From the discussion in the last sections, the key to competitive syntax models is leveraging syntactic information while not decreasing translation model coverage. Rigidly

imposing a syntactic transfer mechanism significantly decreases performance. This has been reported in Koehn et al. (2003); Och et al. (2004); Ambati and Lavie (2008); Liu et al. (2009b), amongst others, and has been the lesson we also learnt from our own work on factored translation and factored templates.

The tree-to-string model of Huang et al. (2006b) constrains the derivation with a source parse. To improve coverage and performance, Mi et al. (2008); Liu et al. (2006) also use phrase-based bilingual phrase-pairs to increase coverage, however, these bilingual phrases are limited to span syntactic constituents.

An extension of the tree-to-string model is the forest-to-string model. However, there are two main arguments against the forest-based approach as described by Huang et al. (2006a); Liu et al. (2006, 2007); Mi et al. (2008); Mi and Huang (2008); Liu et al. (2009a). Firstly, the forest must be pruned to make decoding tractable but pruning is a heuristic that offers no theoretical guarantees that the forest will contain the correct parse. The pruning parameters and algorithm must be tuned and these are critical to the performance of the model.

The second issue with forest-based models is the stated claim that deferring parse disambiguation until decoding allows the decoder to recover from parsing errors. However, there is scant analysis of whether better translation performance is due to recovery from parse errors, or because using a forest improves coverage.

Zhang et al. (2009) combines the forest approach with tree sequences where they note that the two methods are complementary. While forest-based modelling allows the system to recover from parse errors, using tree sequences enables the system to model cross-lingual structural divergence.

Our proposed mixed-syntax model does not require a parse forest as input, just the 1-best parse. Rather than be strictly constrained to a parse and having to assert an incorrect parse, we can choose to ignore parts of the derivation if they are not helpful to translation. We do not try to improve coverage by propagating uncertainty from the parser, but build flexibility into the translation rule.

Nor does our proposed model require ad-hoc schemes such as those of SAMT to categorize spans without syntactic labels. The model only uses labels that are presented by the parser and ignores syntax when they are not present.

Soft-constraints have been used instead of hard constraints to improve coverage. Chiang et al. (2008, 2009); Chiang (2010) all use models which have the same translation rule form as the linguistically unaware hierarchical phrase-based model then parameterize the model with syntax-based feature functions. However, this approach

means that they miss the opportunity to use syntactic information to use rule forms not found in the hierarchical model without over-generation during extraction and spurious ambiguity in decoding.

We have seen in Chapters 2 and 3 that better parameterization of existing models can be achieved with more data. Yet long-range dependency, such as long-distant re-ordering, remains one of the biggest issues facing machine translation. Rather than using syntax for improved parameterization but continuing to constrain the model using linguistically unjustified distortion limits and heuristics to contain its computational feasibility, we will use syntax to alter the constraints to tackle long-range dependency.

## 4.2 Model

### 4.2.1 Preamble

The key difference between the hierarchical model and the various forms of syntactic models is that non-terminals in the hierarchical model are undecorated. Therefore, rewrite rules in hierarchical models have general applicability and broad coverage. However, the general applicability of undecorated non-terminals allows rules to be used in inappropriate situations. This creates spurious ambiguity, increases running time and causes translation errors. In contrast, the decorated non-terminals in syntactic models constrain the application of rewrite rules, increasing specificity but limiting coverage.

We present the *mixed-syntax model* which combines the hierarchical and tree-to-string models, utilizing the broad coverage of hierarchical decoding and the insights that syntactic information can bring. The model seeks to balance the generality of using undecorated non-terminals with the specificity of decorated non-terminals.

The model uses syntactic labels from a source language parser to label non-terminals. However, it is not necessary that the syntactic information form a tree structure, such as a parse tree. The mixed-syntax model is able to make use of syntactic labels as long as they relate to contiguous spans on the source sentence. We can therefore utilize other syntactic tools such as shallow taggers.

In the coming section, we will define the model and decoding algorithm as a deductive proof system (Shieber et al., 1995). A deductive proof system is a set of items,  $A_i$  and  $B$ , and inference rules of the form

$$\frac{A_1 \dots A_k}{B} \quad (\text{side conditions on } A_1 \dots A_k, B)$$

If all the items  $A_1 \dots A_k$  (called the *antecedent*) are provable, then item  $B$  (called the *consequence*) is provable. The deductive process starts with *axioms*, rules are applied to more and more items until a *goal* is proven.

### 4.2.2 CKY for Parsing

As an example, for parsing a sentence  $s = s_1 \dots s_{|s|}$ , the CKY parsing algorithm can be expressed as a deductive proof system, below.

*Item forms*

$[X, i, j]$  is a subtree spanning from  $i$  to  $j$  rooted in  $X$ .

$(X \rightarrow \gamma)$  - a rule where  $X$  is rewritten as  $\gamma$

*Axioms*

$$\frac{}{[X \rightarrow \gamma]} \quad X \rightarrow \gamma \in G$$

*Goal*

$$[S, 0, |s|]$$

*Inference rules*

$$\frac{[Z \rightarrow s_{i+1}]}{[Z, i, i+1]}$$

$$\frac{[Z \rightarrow XY] \quad [X, i, k] \quad [Y, k, j]}{[Z, i, j]}$$

where  $S$  is the start symbol of the grammar.

### 4.2.3 CKY+ for Parsing

The CKY+ algorithm (Chappelier et al., 1998) generalizes the CKY algorithm by permitting grammar rules not in Chomsky normal form, allowing for partially lexicalized rules, and rules with more than two non-terminals on the right-hand-side. The CKY+ algorithm can be expressed as a deductive proof system using dotted chart notation.

The dotted chart notation is used to describe an active chart parser which applies rewrite rules by recognizing each symbol on the right-hand-side, one symbol at a time. A symbol can be a terminal or non-terminal. The recognition of symbols in the rewrite rules is performed strictly left-to-right.

The form of a dotted rule for a monolingual parser is

$$A \rightarrow \alpha \bullet \beta$$

where  $\alpha$  and  $\beta$  are string of terminals and non-terminals.  $\alpha$  and  $\beta$  can be empty, but not both. Symbols in  $\alpha$  have been recognized ('consumed'), while symbols in  $\beta$  have not. A rule is said to be active if  $\beta$  still contain symbols. The rule is passive if all symbols on the right-hand-side have been consumed. The non-terminal  $A$  can then be rewritten as the string  $\alpha\beta$ .

The CKY+ algorithm therefore begin with a set of rules from the grammar where no symbol has been consumed. It ends when a passive rule exist that spans the entire sentence.

#### *Item forms*

1.  $(A \rightarrow \alpha)$  is the source side of the synchronous translation rule which rewrites the source non-terminal  $A$  as the string  $\alpha$ .
2.  $[A \rightarrow \alpha \bullet \beta, i, j]$ , a subtree spanning rooted in  $A$  where the string  $\alpha$  has been recognized in the span  $i$  to  $j$ . String  $\beta$  has yet to be recognized.

#### *Axioms*

$$\frac{}{[A \rightarrow \bullet \alpha, i, i]} \quad (A \rightarrow \alpha) \in G \quad (4.3)$$

#### *Terminal Symbol*

$$\frac{[A \rightarrow \alpha \bullet s_{j+1} \beta, i, j]}{[A \rightarrow \alpha s_{j+1} \bullet \beta, i, j + 1]} \quad (4.4)$$

#### *Non-Terminal Symbol*

$$\frac{[A \rightarrow \alpha \bullet B \beta, i, j] \quad [B \rightarrow \gamma \bullet, j, k]}{[B \rightarrow \alpha B \bullet \beta, i, k]} \quad (4.5)$$

#### *Goal*

$$[S \rightarrow \alpha \bullet, 0, |s|] \quad (4.6)$$

We use the convention of denoting non-terminals with capital letters ( $X, Y, S$ ), and strings of terminals and non-terminals with Greek letters ( $\alpha, \beta, \gamma$ ).

Equation 4.4 recognizes terminal symbols in the input string. Equation 4.5 is known as the fundamental rule of chart parsing (Kay, 1980) which recognizes non-terminals.

### 4.2.4 CKY+ for Hierarchical Phrase-Based Model

A context-free-grammar and the CKY+ algorithm can be used for translation by extending the CFG rules to a synchronous CFG which simultaneously parses the source sentence while generating the target sentence. The hierarchical phrase-based model



simplifies the SCFG translation rules of Equation 4.1 by making linked source and target non-terminals to be identical symbols.

Furthermore, the hierarchical phrase-based model limits the number of non-terminal symbols to one (two if we count the non-terminal  $S$  used by the glue rule). Therefore, the deductive proof for the hierarchical phrase-based translation model is identical to the monolingual parsing model, above, except for Equation 4.5, which can be simplified to:

*Non-Terminal Symbol*

$$\frac{[X \rightarrow \alpha \bullet X \beta, i, j] \quad [X \rightarrow \gamma \bullet, j, k]}{[X \rightarrow \alpha X \bullet \beta, i, k]} \quad (4.7)$$

Many hierarchical and syntax-based SMT decoders, such as Hiero (Chiang, 2005), Joshua (Li et al., 2009) and Moses (Hoang et al., 2009) use the CKY+ algorithm for decoding.

#### 4.2.5 Syntax Decoding using a SCFG

A translation model based on a SCFG can approximate a tree-to-tree model by requiring the source non-terminals to match the labels of the input parse tree. The target non-terminals of the parent rule also need to match the left-hand-side of the child rule.

An input parse tree can be approximated with a tuple  $\langle s, V \rangle$  of words  $s_{1:|s|}$  in the sentence and a set of span labels  $V_{i,j}$  for all spans  $[i, j]$  where  $0 \leq i < |s|$  and  $i + 1 < j \leq |s|$ .

The general form of translation rules in a SCFG-based model is two CFG rules, linked by their non-terminals, Equation 4.8.

$$\langle A \rightarrow \alpha, B \rightarrow \beta, \sim \rangle \quad (4.8)$$

where  $\alpha$  and  $\beta$  are strings of terminals and non-terminals.  $A$  and the string  $\alpha$  are drawn from the same source alphabet,  $\Delta_s$ .  $B$  and  $\beta$  are drawn from the target alphabet  $\Delta_t$ .  $\sim$  is the one-to-one correspondence between non-terminals in  $\alpha$  and  $\gamma$ .

We extend the dotted chart notation from Section 4.2.3 to require the matching of the source left-hand-side of each translation rule, in addition to consuming symbols on the right-hand-side. Therefore, entries in the active parser for the translation model have the form:

$$(\bullet A \rightarrow \alpha' \bullet \alpha'', B \rightarrow \beta)$$

Only when all symbols in the source left-hand-side *and* right-hand-side are consumed is the rule be considered passive.

$$(A\bullet \rightarrow \alpha\bullet, B \rightarrow \beta)$$

The deductive proof system for the SCFG-based syntax model is shown following.

*Item forms*

1.  $(A \rightarrow \alpha, B \rightarrow \beta)$  is a synchronous translation rule which rewrites the source non-terminal  $A$  as the string  $\alpha$ , and simultaneously rewrites the target non-terminal  $B$  as the string  $\beta$ .
2.  $[\bullet A \rightarrow \alpha' \bullet \alpha'', B \rightarrow \beta, i, j]$  is a source subtree rooted in  $A$  and a target subtree rooted in  $B$ . The string  $\alpha'$  has been recognized in the span  $i$  to  $j$ . String  $\alpha''$  and the source non-terminal  $A$  has yet to be recognized.
3.  $[\bullet A \rightarrow \alpha' \bullet C_n \alpha'', X \rightarrow \beta' D_n \beta'', i, j]$  is a source subtree rooted in  $A$  and a target subtree rooted in  $B$ . The string  $\alpha'$  has been recognized in the span  $i$  to  $j$ . The non-terminal  $C_n$  on the source is linked to  $D_n$  on the target. The string  $C_n \alpha''$  and the source non-terminal  $A$  has yet to be recognized.

*Axioms*

$$\frac{}{[\bullet A \rightarrow \bullet \alpha, B \rightarrow \beta, i, i]} \quad (A \rightarrow \alpha, B \rightarrow \beta) \in G \quad (4.9)$$

*Terminal Symbol*

$$\frac{[\bullet A \rightarrow \alpha' \bullet s_{j+1} \alpha'', B \rightarrow \beta, i, j]}{[\bullet A \rightarrow \alpha' s_{j+1} \bullet \alpha'', B \rightarrow \beta, i, j+1]} \quad (4.10)$$

*Non-Terminal Symbol*

$$\frac{[\bullet A \rightarrow \alpha' \bullet C_n \alpha'', B \rightarrow \beta' D_n \beta'', i, j] \quad [C \bullet \rightarrow \gamma \bullet, D \rightarrow \delta, j, k]}{[\bullet A \rightarrow \alpha' C_n \bullet \alpha'', B \rightarrow \beta' D_n \beta'', i, k]} \quad C \in V_{j,k} \quad (4.11)$$

*Left Hand Side*

$$\frac{[\bullet A \rightarrow \alpha \bullet, B \rightarrow \beta, i, j]}{[A \bullet \rightarrow \alpha \bullet, B \rightarrow \beta, i, j]} \quad A \in V_{i,j} \quad (4.12)$$

*Goal*

$$[S \bullet \rightarrow \alpha \bullet, S \rightarrow \beta, 0, |s|] \quad (4.13)$$

The fundamental rule in Equation 4.11 expresses the constraint that non-terminals in the parent rule must match the child left-hand-side for both the source and target.

Equation 4.12 constrain the source left-hand-side non-terminal to also match an input constituent label of the span the rule covers.

The above formalism specify a tree-to-tree model. For a tree-to-string model, the target non-terminals is limited to one label (two counting the non-terminal  $S$  used by the glue rule). Therefore, Equation 4.11 can be simplified to:

*Non-Terminal Symbol*

$$\frac{[\bullet A \rightarrow \alpha' \bullet C_n \alpha'', X \rightarrow \beta' X_n \beta'', i, j] \quad [C \bullet \rightarrow \gamma \bullet, X \rightarrow \delta, j, k]}{[\bullet A \rightarrow \alpha' C_n \bullet \alpha'', X \rightarrow \beta' X_n \beta'', i, k]} \quad C \in V_{j,k} \quad (4.14)$$

#### 4.2.6 Mixed-Syntax Model

We believe that the tree-to-string tree-transducer model of Huang et al. (2006a) limits the use of empirically well-founded but linguistically unmotivated translation rules. We know from Koehn et al. (2003); Och et al. (2004); Ambati and Lavie (2008), amongst others, that rigidly restricting translation to syntactic constituents negatively affects translation performance.

Therefore, the *mixed-syntax model* relaxes the constraints of the tree-to-string model of Huang et al. (2006a) in three main ways.

Firstly, non-terminals in translation rules are not required to have syntactic labels, but can also be ‘undecorated’. Syntactic, or ‘decorated’, non-terminals can only apply to source spans which have been labelled with the same symbol by the parser. Undecorated non-terminals can apply to any span. Rules can have a mixture of decorated and undecorated non-terminals.

This is implemented by using a special symbol  $X$  to denote an undecorated non-terminal. The constituents labels of the parse tree are stored in cells of a chart. In addition, every cell also contains the special  $X$  symbol.

Secondly, the mixed-syntax model uses the formalism described in Section 4.2.3 to approximate the tree-transducer translation model by using SCFG rules with the same root and leaf nodes but ignores the interior nodes of the tree-fragment.

Thirdly, the leaf non-terminal of rules in a derivation is not constrained to match the root node of their child rules. Therefore, each node in the source derivation is not consistently labelled by the root-node of the child rule and the non-terminal symbol of the parent rule. However, decorated non-terminals are constrained by the source parse.

#### 4.2.7 Formal Definition of the Mixed-Syntax Model

In common with most current SMT systems, and with the phrase-based model described in Chapter 2, the objective of decoding is to find the target translation  $\hat{t}$  with

the maximum probability, given a source sentence,  $s$ . As in Chapter 2, we use the maximum derivation as an approximation to maximum translation.

$$\hat{t} = t \left[ \arg \max_{d \text{ s.t. } s(d)=s \& \text{Sync}(d,V)} p(d) \right] \quad (4.15)$$

where  $p(d)$  is the model and the *argmax* function is the search. The ‘*derivation*’  $d$  is an ordered set of translation rules that completely translate  $s$  to  $t$ .  $s(d)$  and  $t(d)$  is the source and target yield of  $d$ , respectively.  $\text{Sync}(d, V)$  is the syntactic constraint that we will describe below in the algorithm section.

$p(d)$  is calculated within a log-linear model described in Chapter 2 which allows arbitrary component models to be used and weighting each model according to its contribution to the total probability.

$$p(d) = \frac{1}{Z} \exp \left[ \sum_m \lambda_m h_m(d) \right] \quad (4.16)$$

where  $h_m(d)$  is the feature function for component  $m$  and  $\lambda_m$  is the weight given to component  $m$ . The normalization factor  $Z$  does not affect optimization, and the  $\exp()$  function is monotonic, therefore, the log-linear model can be expressed as a linear model.

$$\hat{t} = t \left[ \arg \max_{d \text{ s.t. } s(d)=s \& \text{Sync}(d,V)} \sum_m \lambda_m h_m(d) \right] \quad (4.17)$$

#### 4.2.8 Decoding with the Mixed-Syntax Model

The definition of the mixed-syntax model is a variation on the model described in Section 4.2.5.

Again, the input sentence is a tuple  $\langle s, V \rangle$  of words  $s_{1:|s|}$  and a set of span labels  $V_{i,j}$  for all spans  $[i, j]$  where  $0 \leq i < |s|$  and  $i + 1 < j \leq |s|$ . In addition, a source non-terminal symbol,  $X$ , representing the undecorated label is added to all spans.

Translation rules are synchronous context-free-grammar formulated in Equation 4.8. However, the output is non-syntactic, therefore, we can simplify the target side by replacing target non-terminals with a non-syntactic  $X$ , Equation 4.18.

$$\langle A \rightarrow \alpha, X \rightarrow \beta, \sim \rangle \quad (4.18)$$

where  $\alpha$  and  $\beta$  are strings of terminals and non-terminals.  $A$  and the string  $\alpha$  are drawn from the same source alphabet,  $\Delta_s$ .  $\sim$  is the one-to-one correspondence between non-terminals in  $\alpha$  and  $\beta$ .

Again, we formally describe the algorithm using the deductive proof system.

*Item forms*

1.  $(A \rightarrow \alpha, X \rightarrow \beta)$  is a synchronous translation rule which rewrites the source non-terminal  $A$  as the string  $\alpha$ , and simultaneously rewrites the target non-terminal  $X$  as the string  $\beta$ .
2.  $[\bullet A \rightarrow \alpha' \bullet \alpha'', X \rightarrow \beta, i, j]$  is a source subtree rooted in  $A$  and a target subtree rooted in  $X$ . The string  $\alpha'$  has been recognized in the span  $i$  to  $j$ . String  $\alpha''$  and the source non-terminal  $A$  has yet to be recognized.
3.  $[\bullet A \rightarrow \alpha' \bullet C_n \alpha'', X \rightarrow \beta' X_n \beta'', i, j]$  is a source subtree rooted in  $A$  and a target subtree rooted in  $B$ . The string  $\alpha'$  has been recognized in the span  $i$  to  $j$ . The non-terminal  $C_n$  on the source is linked to  $X_n$  on the target. The string  $C_n \alpha''$  and the source non-terminal  $A$  has yet to be recognized.

*Axioms*

$$\overline{[\bullet A \rightarrow \bullet \alpha, X \rightarrow \beta, i, i]} \quad (A \rightarrow \alpha, X \rightarrow \beta) \in G \quad (4.19)$$

*Terminal Symbol*

$$\frac{[\bullet A \rightarrow \alpha' \bullet s_{j+1} \alpha'', X \rightarrow \beta, i, j]}{[\bullet A \rightarrow \alpha' s_{j+1} \bullet \alpha'', X \rightarrow \beta, i, j+1]} \quad (4.20)$$

*Non-Terminal Symbol*

$$\frac{[\bullet A \rightarrow \alpha' \bullet C_n \alpha'', X \rightarrow \beta' X_n \beta'', i, j] \quad [B \bullet \rightarrow \gamma \bullet, X \rightarrow \delta, j, k]}{[\bullet A \rightarrow \alpha' C_n \bullet \alpha'', X \rightarrow \beta' X_n \beta'', i, k]} \quad C \in V_{j,k} \quad (4.21)$$

*Left Hand Side*

$$\frac{[\bullet A \rightarrow \alpha \bullet, X \rightarrow \beta, i, j]}{[A \bullet \rightarrow \alpha \bullet, X \rightarrow \beta, i, j]} \quad A \in V_{i,j} \quad (4.22)$$

*Goal*

$$[S \bullet \rightarrow \alpha \bullet, S \rightarrow \beta, 0, |s|] \quad (4.23)$$

From Equations 4.19, 4.20, 4.23, the mixed-syntax model is initialized, recognizes terminal symbols, and completes identically to the syntax model of Section 4.2.5.

The biggest change is in the fundamental rule of Equation 4.21. The side condition constrain the non-terminal to be consumed,  $C$ , to match a symbol in the span. However,  $C$  is not constrained to match the left-hand-side of the child rule,  $B$ .

Also, every span  $V_{i,j}$  in the input sentence also contains the undecorated non-terminal  $X$ , therefore, the non-terminal  $C$  in Equation 4.21 and  $A$  in Equation 4.22 can be syntactic or non-syntactic.

The decoding algorithm closely follows the CKY+ algorithm described by Chiang (2007). A difference concerns the definition of the span limit. The span limit in (Chiang, 2007) specifies the maximum number of source words the entire translation rule can cover. The mixed-syntax model defines it as the maximum number of source words each non-terminal can cover. The span limit for decorated and undecorated non-terminals can be set independently.

## 4.2.9 Feature Functions

Five component models were used, for all decoding models. The feature function  $h_m$  for each component model is shown below:

1. Translation probability.

$$h_{TM}(d) = \log \prod_{[A \rightarrow \alpha, X \rightarrow \gamma] \in d} p(X \rightarrow \gamma | A \rightarrow \alpha) \quad (4.24)$$

2. Language model

$$h_{LM}(d) = \log p \left[ t(d) \right] \quad (4.25)$$

3. Rule count

$$h_{RC}(d) = |d - d_g| \quad (4.26)$$

4. Word count

$$h_{WC}(d) = |t(d)| \quad (4.27)$$

5. Glue count

$$h_{GLUE}(d) = |d_g| \quad (4.28)$$

$d_g \subset d$  are the glue rules which monotonically concatenate partial derivations.

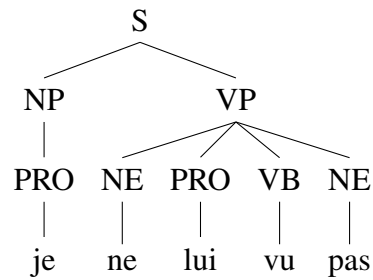
Phrasal probabilities are calculated using fractional counts as described in (Chiang, 2005) which gives a count of one to each initial phrase occurrence, then distributes its weight evenly among the rules obtained by subtracting subphrases from it. The counts are also smoothed using Good-Turing discounting (Foster et al., 2006). The translation model parameterization includes a phrase count feature function, however, the lexical and backward probabilities are not used.

The minimum error rate training algorithm (Och, 2003) was used to optimize the weights,  $\lambda_m$ .

### 4.2.10 Example Decoding with the Mixed-Syntax Model

Supposing a parsed input sentence and the mixed-syntax grammar, below:

Input:



Grammar:

$$\langle S \rightarrow NP_1 VP_2, X \rightarrow X_1 X_2 \rangle$$

$$\langle PRO \rightarrow je, X \rightarrow I \rangle$$

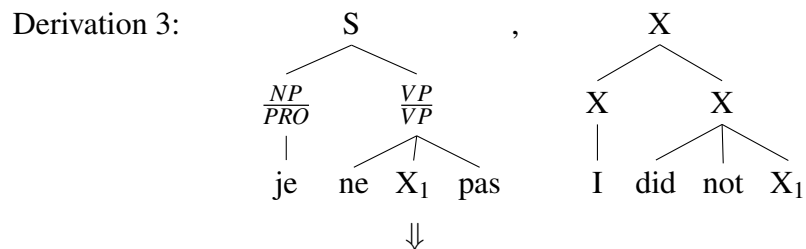
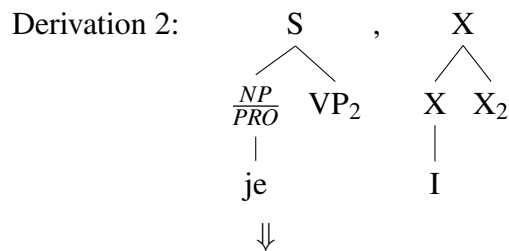
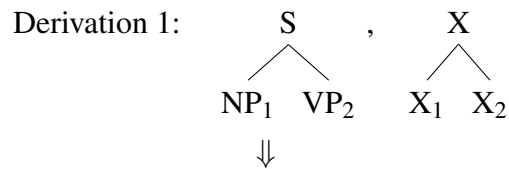
$$\langle PRO \rightarrow lui, X \rightarrow him \rangle$$

$$\langle VB \rightarrow vu, X \rightarrow see \rangle$$

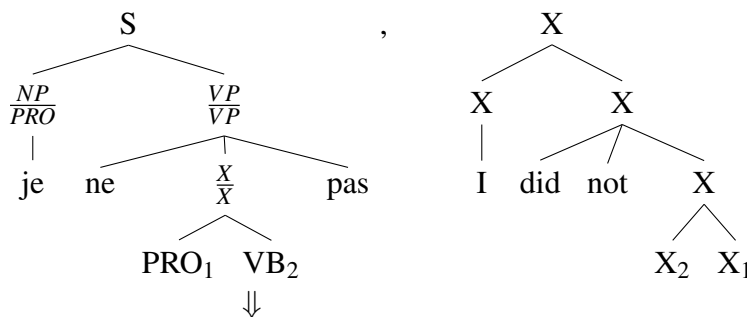
$$\langle VP \rightarrow ne X_1 pas, X \rightarrow did not X_1 \rangle$$

$$\langle X \rightarrow PRO_1 VB_2, X \rightarrow X_2 X_1 \rangle$$

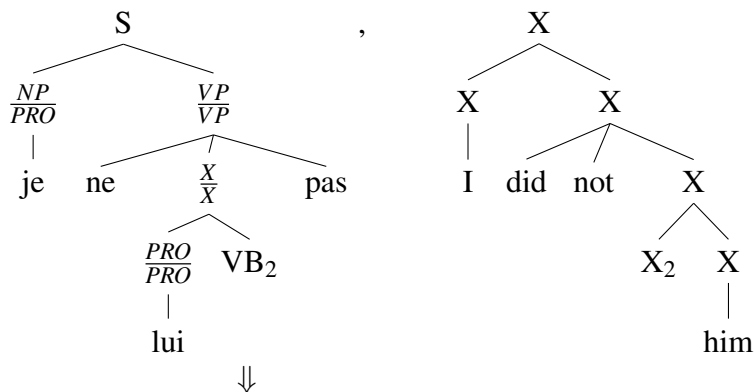
The input can be translated as follows:



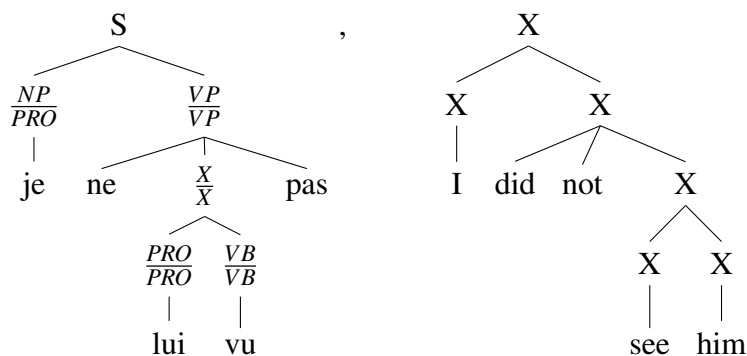
Derivation 4:



Derivation 5:



Derivation 6:



Notice in derivation 2 that the non-terminal label from the parent node, *NP*, does not match the label of the child's root node, *PRO*. However, since both labels cover the source word, *je*, this is a valid inference.

In derivation 3, the free non-terminal does not span a source syntactic constituent. This is valid but permitted only for undecorated non-terminals, denoted by the symbol *X*.

Notice also that the completed source derivation 6 is not isomorphic to the original source parse. The mixed-syntax model allows derivations which is suited for the primary objective of translation, rather than being constrained to replicate the source parse.

#### 4.2.11 Tree-to-String Model

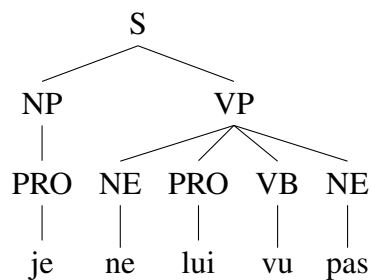
Huang et al. (2006a) described a tree-to-string model using a tree-transducer which



recursively transforms a source parse tree to a target string by repeated applications of translation rules. The translation rules consist of multi-level tree fragments on the source side, and a CFG rewrite rule on the target side, linked by their leaf non-terminals. We approximate this model by using CFG rewrite rules on both source and target side of the translation rule. Translation rules can still transform multiple levels of the parse tree but the interior nodes are ignored, as with the mixed-syntax model.

We approximate the tree-transducer based tree-to-string model with a SCFG-based model. The model is identical to the mixed-syntax model, however, only syntactic translation rules are extracted from the training corpus. Therefore, decoding with such a grammar compels the derivation to closely follow the input parse. The same parsed sentence above can be decoded with a purely syntactic grammar.

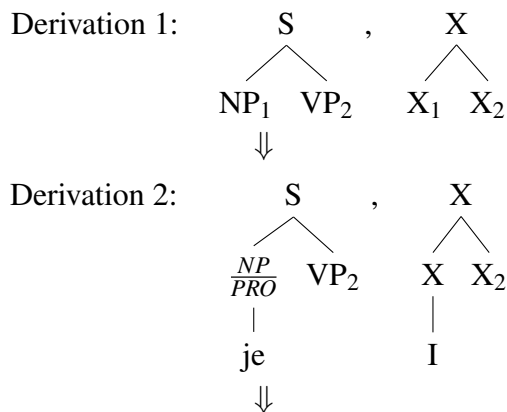
*Input:*



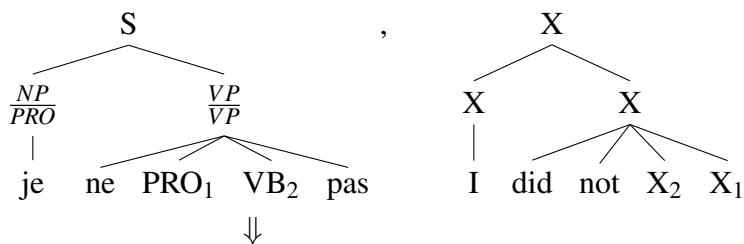
*Grammar:*

$$\begin{aligned}
 &\langle S \rightarrow NP_1 VP_2 \quad , \quad X \rightarrow X_1 X_2 \rangle \\
 &\quad \langle PRO \rightarrow je \quad , \quad X \rightarrow I \rangle \\
 &\quad \langle PRO \rightarrow lui \quad , \quad X \rightarrow him \rangle \\
 &\quad \langle VB \rightarrow vu \quad , \quad X \rightarrow see \rangle \\
 &\langle VP \rightarrow ne \ PRO_1 \ VB_2 \ pas \quad , \quad X \rightarrow did \ not \ X_2 \ X_1 \rangle
 \end{aligned}$$

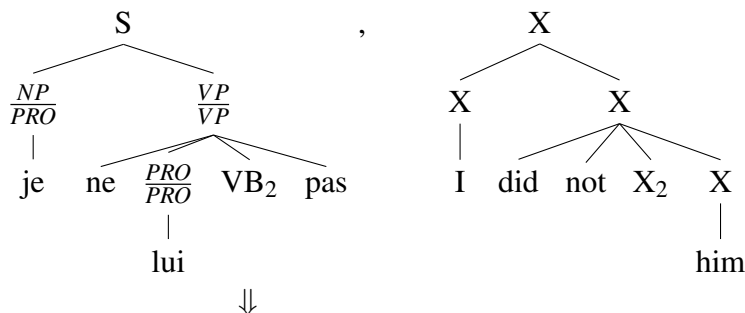
*Derivation:*



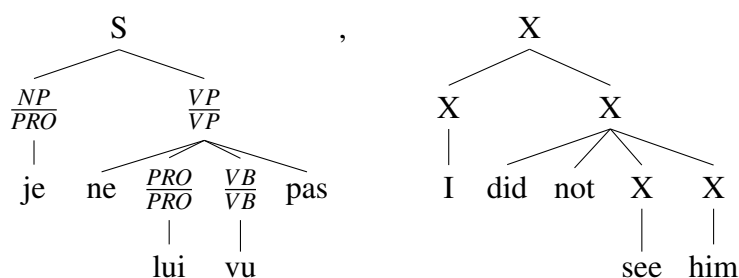
Derivation 3:



Derivation 4:



Derivation 5:



## 4.3 Rule Extraction

We have given the formal definition of the mixed-syntax model and described the algorithm for decoding such a model. In this section, we describe how the synchronous grammar for this model can be learned from a parallel corpus with parsed source sentences.

### 4.3.1 Hierarchical Phrase-Based Model Extraction

The rule extraction heuristics for hierarchical phrase-based, described in (Chiang, 2005), starts with the extraction of initial bilingual phrases from an aligned parallel corpus using the same criterion as most phrase-based extraction heuristics (Och and Ney, 2004).

From the initial phrases, hierarchical phrase-based rules are extracted by finding phrases which contain other phrases and replacing the subphrases with a non-terminal symbol. We give the formation definition in Figure 4.1, based on (Chiang, 2007):

However, this scheme generates a large number of rules which consumes resources during training and decoding, and also leads to a decrease in translation performance

due to spurious ambiguity. Therefore, the hierarchical phrase-based extraction heuristics filter the ruleset with the following constraints:

1. initial phrases are limited to a maximum length of 10 words on either side
2. maximum of 5 source symbols
3. a maximum of two non-terminals per rule
4. non-terminals cannot be adjacent on the source side
5. all rules must be at least partially lexicalized

---

The set of rules,  $G$ , is the smallest set which satisfies the following conditions:

1. If  $\langle s_i^j, t_{i'}^{j'} \rangle$  is an initial phrase, then

$$\langle X \rightarrow s_i^j, X \rightarrow t_{i'}^{j'} \rangle$$

is a rule in  $G$ .

2. If  $\langle X \rightarrow \alpha, X \rightarrow \gamma \rangle$  is a rule in  $G$  and  $\langle s_i^j, t_{i'}^{j'} \rangle$  is an initial phrase such that  $\alpha = \alpha_1 s_i^j \alpha_2$  and  $\gamma = \gamma_1 t_{i'}^{j'} \gamma_2$ , then

$$\langle X \rightarrow \alpha_1 X_k \alpha_2, X \rightarrow \gamma_1 X_k \gamma_2 \rangle$$

is a rule in  $G$ .  $k$  is an index not used in  $\alpha$  and  $\gamma$ .

---

Figure 4.1: Definition of hierarchical phrase-based rules extraction from bilingual phrases

### 4.3.2 Tree-to-string Model Extraction

In the tree-to-string model, non-terminals are restricted to syntactic spans during both training and decoding, severely limiting coverage. The formal definition of the extraction procedure is an extension of the hierarchical phrase-based extraction, Figure 4.2. We use the same rule filtering heuristic as the mixed-syntax model, which will be described below.

The set of rules,  $G$ , is the smallest set which satisfies the following conditions:

1. If  $\langle s_i^j, t_{i'}^{j'} \rangle$  is an initial phrase which spans a syntactic category  $A$ , then

$$\langle A \rightarrow s_i^j, X \rightarrow t_{i'}^{j'} \rangle$$

is a rule in  $G$ .

2. If  $\langle A \rightarrow \alpha, X \rightarrow \gamma \rangle$  is a rule in  $G$  and  $\langle s_i^j, t_{i'}^{j'} \rangle$  is an initial phrase which spans a syntactic category  $B$ , such that  $\alpha = \alpha_1 s_i^j \alpha_2$  and  $\gamma = \gamma_1 t_{i'}^{j'} \gamma_2$ , then

$$\langle A \rightarrow \alpha_1 B_k \alpha_2, X \rightarrow \gamma_1 X_k \gamma_2 \rangle$$

is a rule in  $G$ .  $k$  is an index not used in  $\alpha$  and  $\gamma$ .

---

Figure 4.2: Definition of tree-to-string rules extraction from bilingual phrases

### 4.3.3 Mixed-Syntax Model Extraction

To extract rules for the mixed-syntax model, we extend the tree-to-string extraction by relaxing the constraint of requiring non-terminals, including the root node, to cover parse constituents. Non-terminals which do not cover parse constituents are undecorated, which we denote with the symbol  $X$ , otherwise the non-terminal is decorated with the syntactic label. A rule can contain a mixture of decorated and undecorated non-terminals.

The formal definition for the mixed-model extraction is shown in Figure 4.3. It is clear that the unfiltered extraction method for the hierarchical phrase-based model described in Figure 4.1 and the mixed-syntax extraction of Figure 4.3 extract the same rules, given identically aligned corpora, except that the mixed-syntax extraction heuristic decorate some of the non-terminals with syntactic labels.

If the grammar were to be filtered with the same heuristic as the hierarchical grammar, described in Section 4.3.1, this would create a grammar whose rules have the same form as the filtered hierarchical phrase-based grammar. Namely, the grammar will only contain translation rules with a maximum of two non-terminals, no rules with adjacent source non-terminals, and all rules must be at least partially lexicalized.

However, we believe that a deficiency of the hierarchical phrase-based model is its inability to extract or decode with certain rule-forms without over-generation dur-

The set of rules,  $G$ , is the smallest set which satisfies the following conditions:

1. If  $\langle s_i^j, t_{i'}^{j'} \rangle$  is an initial phrase which spans a syntactic category  $A$ , then

$$\langle A \rightarrow s_i^j, X \rightarrow t_{i'}^{j'} \rangle$$

is a rule in  $G$ . Otherwise

$$\langle X \rightarrow s_i^j, X \rightarrow t_{i'}^{j'} \rangle$$

is a rule in  $G$ .

2. If  $\langle A \rightarrow \alpha, X \rightarrow \gamma \rangle$  is a rule in  $G$  and  $\langle s_i^j, t_{i'}^{j'} \rangle$  is an initial phrase which spans a syntactic category  $B$ , such that  $\alpha = \alpha_1 s_i^j \alpha_2$  and  $\gamma = \gamma_1 t_{i'}^{j'} \gamma_2$ , then

$$\langle A \rightarrow \alpha_1 B_k \alpha_2, X \rightarrow \gamma_1 X_k \gamma_2 \rangle$$

is a rule in  $G$ . If  $\langle s_i^j, t_{i'}^{j'} \rangle$  does not span a syntactic category

$$\langle A \rightarrow \alpha_1 X_k \alpha_2, X \rightarrow \gamma_1 X_k \gamma_2 \rangle$$

is a rule in  $G$ .  $k$  is an index not used in  $\alpha$  and  $\gamma$ .

3. Similarly, if  $\langle X \rightarrow \alpha, X \rightarrow \gamma \rangle$  is a rule in  $G$ ,

$$\langle X \rightarrow \alpha_1 B_k \alpha_2, X \rightarrow \gamma_1 X_k \gamma_2 \rangle$$

or

$$\langle X \rightarrow \alpha_1 X_k \alpha_2, X \rightarrow \gamma_1 X_k \gamma_2 \rangle$$

can be created in exactly the same way.

---

Figure 4.3: Definition of mixed-syntax rules extraction from bilingual phrases

ing extraction and creating spurious ambiguity during decoding. Rules with adjacent source non-terminals, unlexicalized rules, and rules with more than two non-terminals can be useful for translation but are filtered from the grammar to reduce these problems.

The use of constrained syntactic non-terminals decreases the spurious ambiguity from the mixed-syntax model. We therefore take this opportunity to alter the filtering heuristic to extract and decode with a grammar that contains rule-forms described

above. However, this must still be balanced with the need to maintain tractability.

The filtering constraints for the mixed-syntax, and tree-to-string model, is set down below:

1. initial phrases are limited to a maximum length of 10 words on either side
2. maximum of 5 source symbols (i.e words and non-terminals)
3. a maximum of three non-terminals per rule
4. adjacent non-terminals are allowed on the source side if at least one non-terminal is decorated
5. prohibit translation rules where the source side is a unary rule and a non-terminal.

As can be seen, constraints (1) and (2) are identical to those of the hierarchical phrase-based filter. We increased the maximum number of non-terminals to three in constraint (3) after experimentation.

Constraint (4) of the mixed-syntax filter encapsulates and extends the non-adjacent non-terminal constraint in the hierarchical phrase-based filter. The new constraint continues to prohibit adjacent undecorated non-terminals to prevent spurious ambiguity. However, the filter allows rules where one or both adjacent non-terminals are decorated.

The requirement that all rules must be at least partially lexicalized is dropped but unlexicalized unary rewrite rules are prohibited.

The translation rules forms resulting from this filter are a superset of the rule forms created using the hierarchical phrase-based filter. The differences include rules with adjacent source non-terminals, unlexicalized rules and rules with more than two non-terminals. Of course, the rules are different due to the syntactic label on some of the non-terminals in the mixed syntax rules.

We use the same filtering constraints for extracting rules for the tree-to-string model. Since this model has no undecorated non-terminals, the prohibition on adjacent undecorated non-terminals is redundant.

#### **4.3.4 Examples of Extracted Rules**

We give an example of the rules extracted for the mixed-syntax model from an aligned sentence in Figure 4.4, with a parse tree on the source side. The following initial

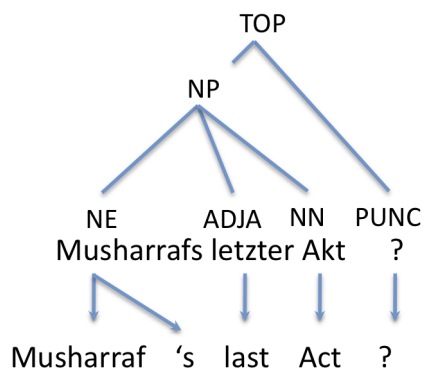


Figure 4.4: Aligned parsed sentence

bilingual phrases are extracted from this aligned sentence:

⟨Musharrafs # Musharraf 's⟩  
 ⟨Musharrafs letzter # Musharraf 's last⟩  
 ⟨Musharrafs letzter Akt # Musharraf 's last Act⟩  
 ⟨Musharrafs letzter Akt ? # Musharraf 's last Act ?⟩  
 ⟨letzter # Last⟩  
 ⟨letzter Akt # Last Act⟩  
 ⟨letzter Akt ? # Last Act ?⟩  
 ⟨Akt # Act⟩  
 ⟨Akt ? # Act ?⟩  
 ⟨? # ?⟩

Fully lexicalized translation rules are created from the initial phrases by adding a source root node to each phrase. The root node is the syntactic label that the source side of the phrase covers, or the symbol  $X$  denoting an undecorated symbol if a syntactic

label does not exist.

$$\begin{aligned}
 &\langle NE \rightarrow \text{Musharrafs} \quad \# \quad X \rightarrow \text{Musharraf 's} \rangle \\
 &\langle X \rightarrow \text{Musharrafs letzter} \quad \# \quad X \rightarrow \text{Musharraf 's last} \rangle \\
 &\langle NP \rightarrow \text{Musharrafs letzter Akt} \quad \# \quad X \rightarrow \text{Musharraf 's last Act} \rangle \\
 &\langle TOP \rightarrow \text{Musharrafs letzter Akt ?} \quad \# \quad X \rightarrow \text{Musharraf 's last Act ?} \rangle \\
 &\quad \langle ADJA \rightarrow \text{letzter} \quad \# \quad X \rightarrow \text{last} \rangle \\
 &\quad \langle X \rightarrow \text{letzter Akt} \quad \# \quad X \rightarrow \text{last Act} \rangle \\
 &\quad \langle X \rightarrow \text{letzter Akt ?} \quad \# \quad X \rightarrow \text{last Act ?} \rangle \\
 &\quad \quad \langle NN \rightarrow \text{Akt} \quad \# \quad X \rightarrow \text{Act} \rangle \\
 &\quad \quad \langle X \rightarrow \text{Akt ?} \quad \# \quad X \rightarrow \text{Act ?} \rangle \\
 &\quad \quad \langle PUNC \rightarrow ? \quad \# \quad X \rightarrow ? \rangle
 \end{aligned}$$

Other rules containing non-terminals on the right-hand-side are created recursively according to Figure 4.3 and filtered according to the description in Section 4.3.3. The following is a non-exhaustive list of rules created from the initial phrase of the complete sentence below. These include rules with syntactic non-terminals:

$$\begin{aligned}
 &\langle TOP \rightarrow NP_1 ? \quad \# \quad X \rightarrow X_1 ? \rangle \\
 &\langle TOP \rightarrow \text{Musharrafs letzter Akt } PUNC_1 \quad \# \quad X \rightarrow \text{Musharraf 's last Act } X_1 \rangle \\
 &\quad \langle TOP \rightarrow NP_1 PUNC_2 \quad \# \quad X \rightarrow X_1 X_2 \rangle \\
 &\quad \langle TOP \rightarrow NE_1 \text{ letzter Akt } PUNC_2 \quad \# \quad X \rightarrow X_1 \text{ last Act } X_2 \rangle \\
 &\langle TOP \rightarrow \text{Musharrafs } ADJA_1 NN_2 PUNC_3 \quad \# \quad X \rightarrow \text{Musharraf 's } X_1 X_2 X_3 \rangle
 \end{aligned}$$

Hierarchical phrase-based style rules are also extracted where the span does not exactly match a parse constituent. The root node can also be an undecorated non-terminal but since the initial phrase in this example spans the source constituent *TOP* so the root node remains decorated. We list two examples below.

$$\begin{aligned}
 &\langle TOP \rightarrow \text{Musharrafs } X_1 ? \quad \# \quad X \rightarrow \text{Musharraf 's } X_1 ? \rangle \\
 &\langle TOP \rightarrow \text{Musharrafs letzter } X_1 \quad \# \quad X \rightarrow \text{Musharraf 's last } X_1 \rangle
 \end{aligned}$$



Rules are also extracted which contain a mixture of decorated and undecorated non-terminals:

$$\langle TOP \rightarrow \text{Musharrafs } X_1 \text{ PUNC}_2 \quad \# \quad X \rightarrow \text{Musharraf 's } X_1 X_2 \rangle$$

$$\langle TOP \rightarrow X_1 \text{ Akt } \text{PUNC}_2 \quad \# \quad X \rightarrow X_1 \text{ Act } X_2 \rangle$$

Rules with adjacent source non-terminals, unlexicalized rules, and rules with more than two non-terminals are also extracted:

$$\langle TOP \rightarrow NE_1 X_2 \text{ PUNC}_3 \quad \# \quad X \rightarrow X_1 X_2 X_3 \rangle$$

The number of rules extracted by the mixed-syntax extraction heuristics is larger than those from the hierarchical phrase-based heuristics. For instance, 103 mixed-syntax rules were extracted from this aligned sentence, compared to 32 hierarchical phrase-based rules. In contrast, 51 were extracted for the tree-to-string grammar.

## 4.4 Experiments

We use the Moses implementation of the hierarchical phrase-based decoder as described in Hoang et al. (2009). For the tree-to-string model, we use the SCFG-based approximation described in Section 4.2.11. The decoder implements the CKY+ algorithm with cube pruning which is kept identical for all experiments for fair comparison.

The definition of maximum rule span as described in Section 4.2.8 is used throughout unless otherwise mentioned; all non-terminals can cover a maximum of 7 source words.

The translation models were trained on the New Commentary 2009 corpus<sup>1</sup>, tuning on a held-out set. Table 4.2 gives more details on the datasets used. The *nc\_test2007* dataset was used for testing.

The training corpus was cleaned and tokenized using standard techniques as implemented in the Moses toolkit (Koehn et al., 2007) and aligned using GIZA++ (Och and Ney, 2003). Minimum error rate training (Och, 2003) was used for tuning the weights in the log-linear model. The target side of the training data was also used to create a trigram language model which was used for each experiment. All experiments use truecased data and results are reported in case-sensitive BLEU scores (Papineni et al., 2002). A standard hierarchical model was used as a baseline.

<sup>1</sup><http://www.statmt.org/wmt09/>

		German	English
Train	Sentences	82,306	
	Words	2,034,373	1,965,325
Tune	Sentences	2000	
Test	Sentences	1026	

Table 4.2: Training, tuning, and test conditions

The Bitpar parser<sup>2</sup> was used to parse the German portion of the training, tuning and test data. 2042 sentences in the training corpus failed to parse and were discarded from the training for both hierarchical phrase-based and syntactic models to ensure that they train on identical amounts of data. Similarly, 991 out of 1026 sentences were parsable in the test set. To compare like-for-like, the baseline hierarchical phrase-based system translates the same 991 sentences, but evaluated over 1026 sentences. For an alternative baseline comparison, we also trained and tested a hierarchical phrase-based model which used all the available data. The result was used as the baseline for the shallow parsing experiments in which every sentence is parseable.

The mixed-syntax model is not dependent on the tree structure of the source syntactic information, only that the syntactic constituents identify continuous spans. A shallow parser (also called partial parser or chunker) (Abney, 1991) satisfies this requirement by recognizing and labelling simple syntactic structures without necessarily building a coherent syntactic tree for the entire sentence. In the shallow parsing experiments, chunk tags from the Treetagger chunker (Schmidt and Schulte im Walde, 2000) were used instead of the full parse tree.

In addition to the mixed-syntax model described previously, we also investigate another approach to combining hierarchical phrase-based and syntactic models, first described by Liu et al. (2009b). This approach jointly decodes with a hierarchical phrase-based and the tree-to-string model, producing a combined hypergraph of translation derivations. Translations can be extracted from the hypergraph that uses translation rules from both models. Liu et al. (2009b) found that this improved translation for large scale Chinese-English task by 1.5 BLEU over a hierarchical phrase-based baseline.

<sup>2</sup><http://www.ims.uni-stuttgart.de/tcl/SOFTWARE/BitPar.html>

### 4.4.1 Results

Using the naïve tree-to-string model constrained with syntactic non-terminals significantly decreases translation quality, Table 4.3, line (2). This confirms the results from Mi et al. (2008); Huang (2008); Liu et al. (2009b).

Using the hierarchical phrase-based jointly with the tree-to-string model resulted in little improvement, line (3) which contradicts Liu et al. (2009b). However, a difference in the implementation of the decoding algorithm may explain this contradiction. In Liu et al. (2009b), hypotheses created from rules from each translation model are added to the chart separately.

When jointly decoding with Moses, translation rules from the two models form one priority queue for cube pruning. The cube pruning limit is unchanged regardless of the number of models. This speeds up decoding but creates competition between the two models within the priority queue. It was noticed that the MERT tuning adjusted the weights so that translation rules from the tree-to-string model are rarely used during joint decoding, suggesting that the broader coverage of the hierarchical phrase-based model was preferred over the precision of the tree-to-string model.

While joint decoding shows minor change (+0.2 BLEU), the mixed-syntax model improves translation quality by +0.8 BLEU over the hierarchical phrase-based baseline, (line 4).

	Model	% BLEU
1	Hierarchical	15.9
2	Tree-to-string	14.9
3	Joint	16.1
4	Mixed-syntax	16.7

Table 4.3: German–English results for hierarchical and syntactic models, in %BLEU

The maximum number of non-terminals per rule was varied to find the optimal constraint for this parameter. Reducing the maximum number of non-terminals per rule reduces translation quality but increasing it has little effect on the mixed-syntax model, Table 4.4.

# non-terms	2	3	4	5
% BLEU	16.5	16.8	16.7	16.8

Table 4.4: Effect on %BLEU of varying maximum number of non-terminals

#### 4.4.2 Rule Span Width During Decoding

Figure 4.5 shows the source span width that translation rules cover when translating the test dataset for the hierarchical phrase-based model and the mixed-syntax model. The mixed-syntax model makes use of proportionally more unigram translation rules than the hierarchical model. This is surprising as the mixed-syntax grammar has rules with three non-terminals which can each cover seven source words, while translation rules in the hierarchical model have a maximum of two non-terminals.

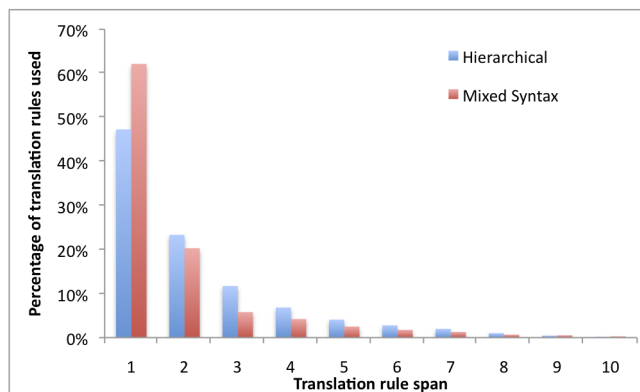


Figure 4.5: Source span lengths

However, analyzing the usage of the glue rules gives us a better indication of how the mixed-syntax model is able to improve translation. All models use the glue rule (Chiang, 2005), which allows the decoder to stop building hierarchical structures and instead concatenate partial translations without reordering. For the sake of efficiency, the decoder imposes a maximum span limit above which the glue rule must be used. The use of the glue rule, therefore, indicates one of two issues during decoding. It can signal that the artificial span limit has been reached. It can also indicate the failure of the translation model to explain the translation as the model does not have the required translation rules. Therefore, a decrease in the use of glue rules is evidence of the better explanatory power of the mixed-syntax model.

As can be seen in Figure 4.6, there is significantly less usage of the glue rules when

decoding with the mixed-syntax model.

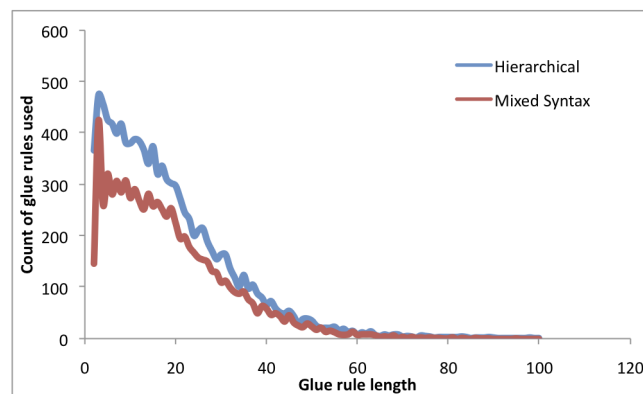


Figure 4.6: Length and count of glue rules used decoding test set

### 4.4.3 Example Decoding Derivation

An example of an input sentence, and the best translation found by the hierarchical phrase-based and mixed-syntax model can be seen in Table 4.5. Both translations are lexically identical but the output of the hierarchical model needs to be reordered to be grammatically correct, which the mixed-syntax model has achieved.

<u>Input</u> laut János Veres wäre dies im ersten Quartal 2008 möglich .
<u>Hierarchical phrase-based model output</u> according to János Veres this in the first quarter of 2008 would <b>be possible</b> .
<u>Mixed-syntax model output</u> according to János Veres this would <b>be possible</b> in the first quarter of 2008 .

Table 4.5: Example input and output decoding with each model

Figure 4.7 is the parsed input sentence to a mixed-syntax model. Contrast the derivations produced by the hierarchical grammar, Figure 4.8, with that produced with the mixed-syntax model, Figure 4.9. The rules used by the mixed-syntax model during the translation of the sentence is shown in Figure 4.10.

The mixed-syntax derivation use several rules which are partially decorated. Crucially, the last rule in the list above reorders the *PP* phrase and the non-syntactic phrase

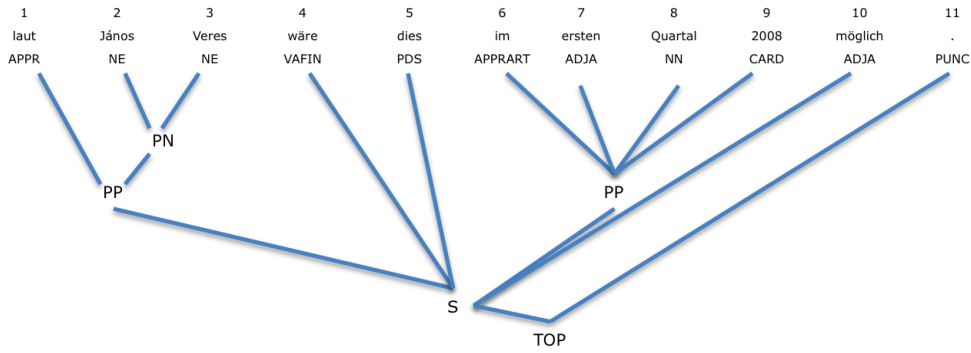


Figure 4.7: Example input parse tree

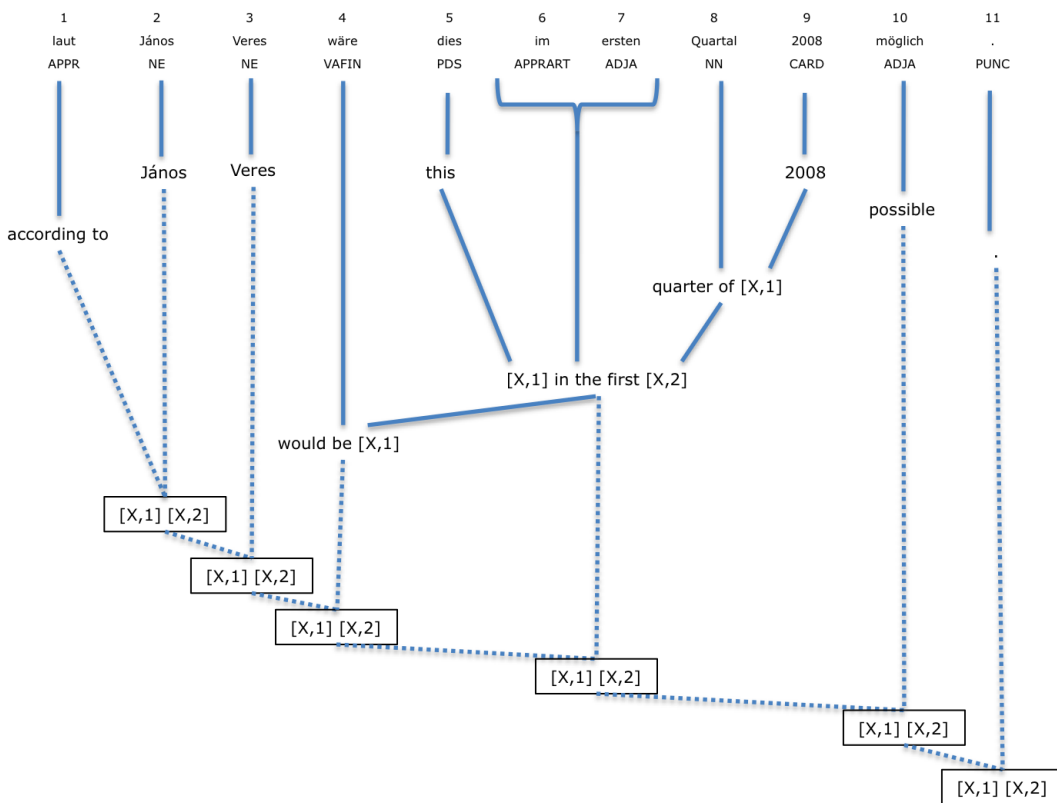


Figure 4.8: Derivation with Hierarchical model

$X$  to generate the grammatically correct output. The other non-lexicalized rules monotonically concatenate the output. This can be performed by the glue rule, but nevertheless, the use of translation rules based on empirical evidence allows the decoder to make a better comparison with hypotheses that have reordered the subphrases. Overall, the derivation rely less on the glue rules than the hierarchical model.

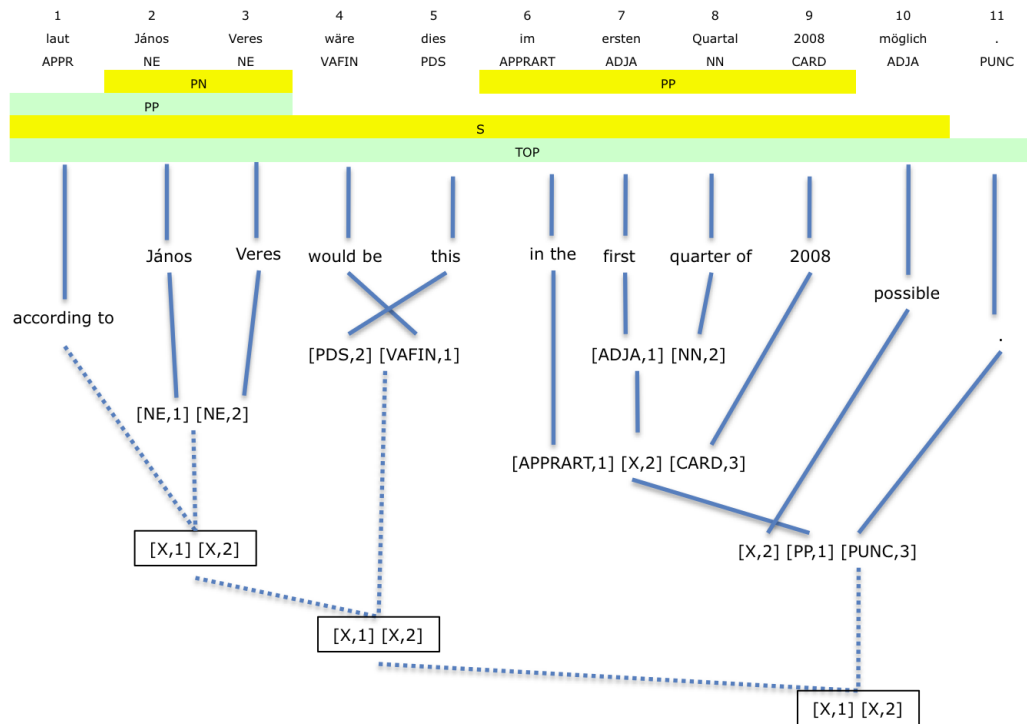


Figure 4.9: Derivation with mixed-syntax model

$$\begin{aligned}
 &\langle PN \rightarrow NE_1 NE_2 \quad \# \quad X \rightarrow X_1 X_2 \rangle \\
 &\langle X \rightarrow VAFIN_1 PDS_2 \quad \# \quad X \rightarrow X_1 X_2 \rangle \\
 &\langle X \rightarrow ADJA_1 NN_2 \quad \# \quad X \rightarrow X_1 X_2 \rangle \\
 &\langle X \rightarrow APPRART_1 X_2 CARD_3 \quad \# \quad X \rightarrow X_1 X_2 X_3 \rangle \\
 &\langle X \rightarrow PP_1 X_2 PUNC_3 \quad \# \quad X \rightarrow X_2 X_1 X_3 \rangle
 \end{aligned}$$

Figure 4.10: Rules used by the mixed-syntax model to translate sentence

## 4.5 Using Shallow Parsers

Parse trees of the source language provide useful information that we have exploited with the mixed-syntax model. However, parsers are an expensive resource as they usually need manually annotated training treebanks. Parse errors are also problematic and more frequent when sentences not in the same domain as the parser training corpus. The brittleness of many parsers also causes some sentences to be unparseable. For

example, our original test corpus of 1026 sentences contained 35 unparsable sentences. Thus, high quality parsers are unavailable for many source languages of interest.

Parse forests can be used to mitigate the parse errors, allowing the decoder to choose from many alternative parses, (Mi et al., 2008).

However, the mixed-syntax translation model is not dependent on the linguistic information being in a tree structure, only that the labels identify continuous spans. Shallow parsers (Abney, 1991) do just that. They offer high accuracy, are not so brittle to out-of-domain data and identify chunk phrases similar to parser-based syntactic phrases that may be useful for translation.

We create a mixed-syntax model for the same German-English language pair, replacing the use of parse constituents in the previous sections with chunk phrases.

#### 4.5.1 Experiments with Shallow Parsers

We use the same data as described earlier in this chapter to train, tune and test our approach. The same alignment information was used to ensure consistency. The Tree-tagger chunker (Schmidt and Schulte im Walde, 2000) was used to tag the source (German) side of the corpus. The chunker successfully processed all sentences in the training and test dataset so no sentences were excluded. The increase in training data, as well as the ability to translate all sentences in the test set, accounts for the higher hierarchical baseline than the previous experiments with parsed data. We use noun, verb and prepositional chunks, as well as part-of-speech tags, emitted by the shallow parser.

Results are shown in Table 4.6. Using chunk tags, we see a gain of 0.5 BLEU, showing that gains can be obtained with simpler tools than full syntactic parsers.

	Model	% BLEU
1	Hierarchical	16.3
2	Mixed-syntax, using chunk tags	16.8

Table 4.6: German–English results for hierarchical and mixed-syntax models using chunk tags, in %BLEU

The use of glue rules by the mixed-syntax model was reduced, Figure 4.11, similar to the trend with saw in Section 4.4.2. This shows once again that the mixed-syntax



model can explain more of the translation with the use of empirically informed translation rules.

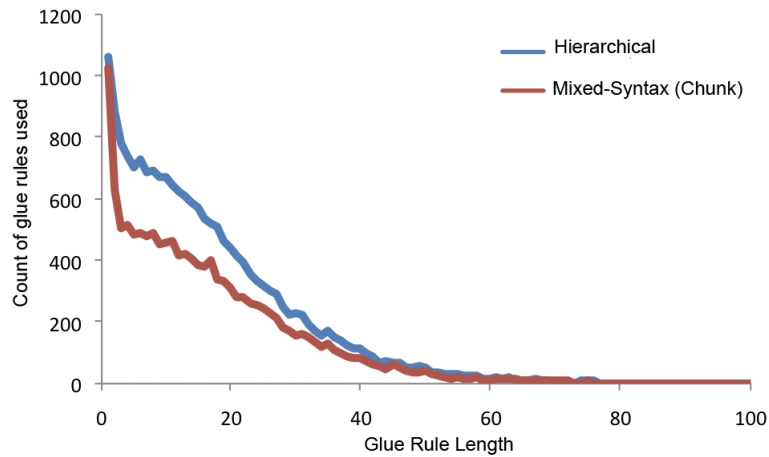


Figure 4.11: Chunk - Length and count of glue rules used decoding test set

The same example sentence in Table 4.5 is shown with chunk tags in Figure 4.12.

1	2	3	4	5	6	7	8	9	10	11
laut	János	Veres	wäre	dies	im	ersten	Quartal	2008	möglich	.
APPR	NE	NN	VAFIN	PDS	APPRART	ADJA	NN	CARD	ADJA	PUNC
PC			VC	NC	PC					

Figure 4.12: Chunked sentence

The mixed-syntax model with chunk tags produced the derivation tree shown in Figure 4.13. The derivation makes use of an unlexicalized rule local reordering. In this example, it uses the same number of glue rules as the hierarchical derivation but the output is grammatically correct by not translating the last source word.

according to János Veres this would **be** in the first quarter of 2008 .

Table 4.7: Example best output by mixed-syntax model with chunk tags

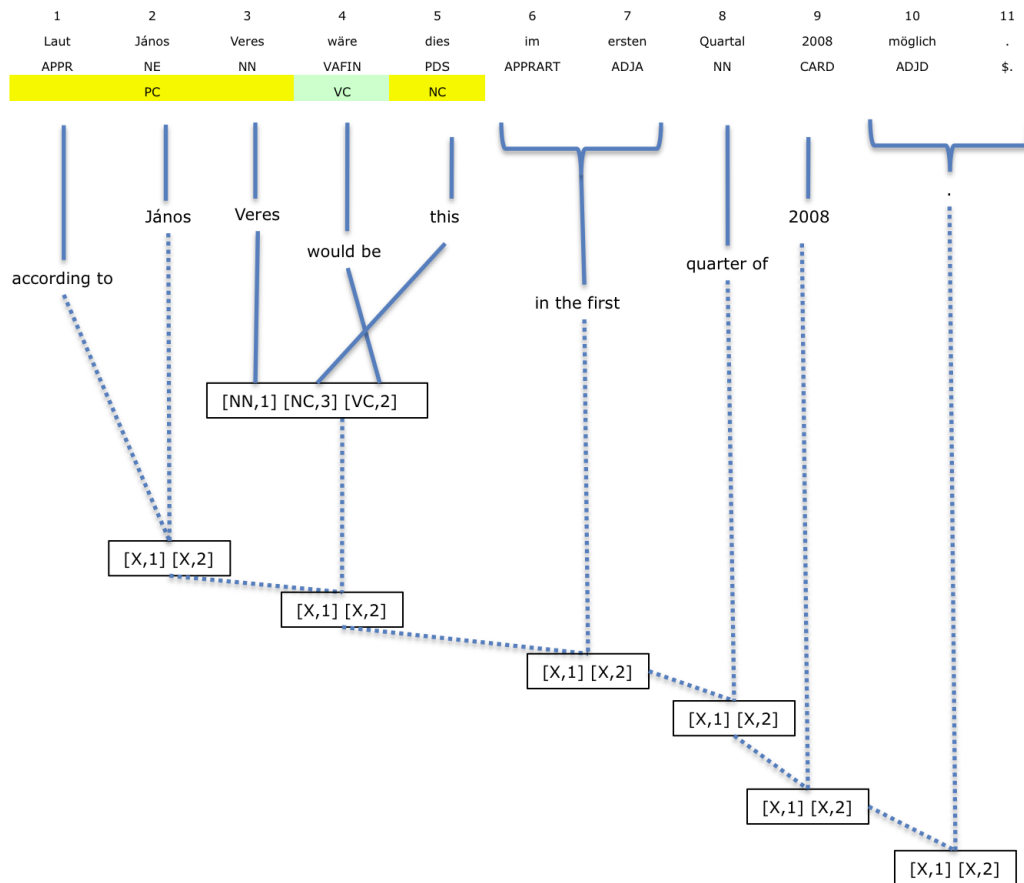


Figure 4.13: Translated chunked sentence

## 4.6 English to German

We experimented with the reverse language direction to see if the mixed-syntax model still increased translation quality. The results were positive but less pronounced, Table 4.8.

	Model	% BLEU
1	Hierarchical	10.2
2	Mixed-syntax	10.6

Table 4.8: English–German results in %BLEU

## 4.7 Large Training Corpora

As with previous chapters, the mixed-syntax model was developed and tested on a small dataset. In this section, we attempt to scale it to more training data.

The Europarl<sup>3</sup> corpora was used as the training data for a large German-English model. The training corpus contains 1,540,549 parallel sentences but once cleaned and filtered of unparseable sentences this was reduced to 1,446,222 sentences. Note that this is the same dataset as the large data experiments in Chapter 2, with more sentences discarded due to unparseable sentences. The language model was trained on the superset of the parallel corpus of 1,843,035 sentences.

In-domain, hold-out data was used for MERT tuning and tested on in and out-of-domain test sets. Table 4.9 gives more details on the datasets used. Again, unparseable sentences are not used.

		German	English	Corpus ID
Train	Sentences	1,446,224		Europarl v5
	Words	37,420,876	39,464,626	
Tune	Sentences	1910		dev2006
Test (out-of-domain)	Sentences	1042		nc_test2007 v2
Test (in-domain)	Sentences	1920		devtest2006

Table 4.9: Training, tuning, and test conditions

The translation performances, as measure by BLEU, are shown in Table 4.10. The results for the mixed-syntax model relative to the hierarchical phrase-based model are disappointing, showing reduced performance for both in and out-of-domain test sets.

	Model	In-Domain	Out-of-domain
1	Hierarchical	22.1	16.5
2	Mixed-syntax	21.6	16.3

Table 4.10: German–English results, trained on Europarl corpus, in %BLEU

Parse errors may be a cause of this degradation. For example, the word *Gewiss* is only seen in the training as an adverb, but is also seen in the test data as a noun.

<sup>3</sup><http://www.statmt.org/europarl/>

Methods based on parse forest Huang et al. (2006a); Liu et al. (2006, 2007); Mi et al. (2008); Mi and Huang (2008) may ameliorate this problem.

However, a major reason for the poor performance of the mixed-syntax model when trained with large training corpora is the increased number of rules available for each source span. This causes multiple derivations that generates the same target string, reducing in the diversity of the n-best list, Figure 4.14. Each sentence generates 5.5 on average in 100-best list using the mixed-syntax model, trained on the small training corpora of Section 4.4,. However, for larger training data this falls to 2.4, compared to 3.4 when using the hierarchical phrase-based model. In turn, this affects the performance of the tuning algorithm. The increased number of rules also causes search

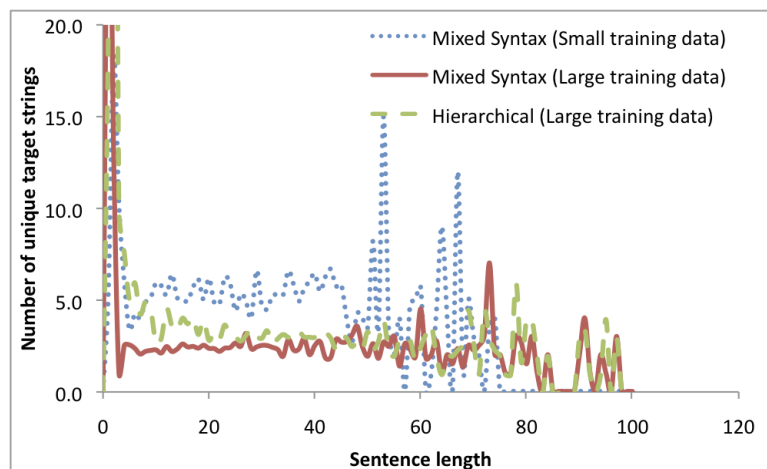


Figure 4.14: Number of unique target string per sentence in 100-best list during tuning

errors during decoding as the fixed cube-pruning pop limit means that the number of target strings evaluated is reduced.

## 4.8 Improved Mixed-Syntax Extraction

The mixed-syntax model offers better translation when trained with a small corpus. However, when trained with large corpora the model produces more derivations which result in the same target sentence. This spurious ambiguity decreases translation quality as it reduces the number of target sentences seen during decoding. Spurious ambiguity also reduces the number of unique translations in the n-best list during tuning, diminishing the accuracy of the tuning algorithm.

In this section, we apply a range of heuristics to contain spurious ambiguity in the

mixed-syntax model. This reduces the size of the grammar extracted from the training data, reducing spurious ambiguity, but does not reduce the ability of the translation model to translate.

We also alter the training and decoding algorithm so that longer-range dependencies can be modelled while maintaining tractability.

These changes turn the translation performance of the mixed-syntax model from under-performing the hierarchical phrase-based baseline for large training corpora, to outperforming the baseline by 0.6 BLEU (1.3 BLEU out-of-domain).

### 4.8.1 Issues with Mixed-Syntax Extraction

The mixed-syntax model extraction algorithm is based on the original hierarchical phrase-based extraction algorithm, described in Section 4.3.1. In the hierarchical phrase-based extraction, initial fully-lexicalized translation rules are extracted from the training data. Translation rules containing subphrases are created by replacing subphrases in existing translation rules with a non-terminal.

The mixed-syntax extraction follows this recursive algorithm but labels the non-terminals with the syntactic category from a source language parser or shallow chunker, where such labels exist for a particular source span. The filtering constraints is also modified to allow rules with three non-terminal, consecutive non-terminals and unlexicalized, as describe in Section 4.3.3.

However, the mixed-syntax extraction algorithm give rise to four main issues which we tackle in this section.

Firstly, the parser often labels a span in the source sentence with multiple labels. Consider the example in Figure 4.15 of partial aligned sentence and the corresponding word alignment and parse information. The source range on the word *mir* is has three constituent labels; NN, NP and PN. The multiple labels give rise to three translation rules extracted from the same span which are identical apart from the label of the first non-terminal.

$$\begin{aligned} \langle X \rightarrow PN_1 \text{ liegt daran , } \# \quad X \rightarrow X_1 \text{ keen that} \rangle \\ \langle X \rightarrow NP_1 \text{ liegt daran , } \# \quad X \rightarrow X_1 \text{ keen that} \rangle \\ \langle X \rightarrow NN_1 \text{ liegt daran , } \# \quad X \rightarrow X_1 \text{ keen that} \rangle \end{aligned}$$

We believe that the functionality of such rules largely overlap. The application of each rule create the same output sentence, wasting the limited resources of the decoder.

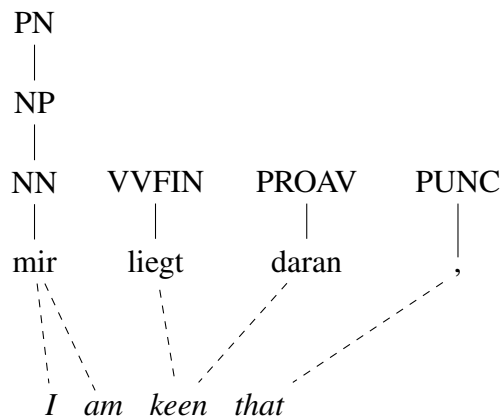


Figure 4.15: Example partial sentence, alignment information, and parse structure

This causes degraded translation performance as the number of derivations created during decoding is usually limited by cube pruning limits (Huang and Chiang, 2007) and beam thresholds.

To counter this affect, therefore, only the top most label in the parse tree for each source range is used during extraction. In the above example, the word *mir* is thus only labelled with the category *PN*, and hence, only the first rule above is extracted.

Secondly, the algorithm extract rules with multiple decorated and undecorated non-terminals but little lexical evidence. This also occur in the extraction of hierarchical grammar but is prevalent in the mixed-syntax grammar due to the relaxation of the filtering constraints, allowing increased number of non-terminals, consecutive non-terminals and unlexicalized rules. To exclude those rules with little empirical evidence, translation rules with undecorated non-terminals must also contain at least the same number of source words. For example, the following rule is prohibited as it contains an undecorated non-terminal but no source words.

$$\langle X \rightarrow PN_1 X_2 PUNC_3 \quad \# \quad X \rightarrow X_1 X_2 X_3 \rangle$$

Thirdly, the mixed-syntax extraction heuristic produces translation rules with non-terminals in the first and last position of the source phrase. For undecorated non-terminals in these positions, it allow flexibility in the span to which they can be applied during decoding. However, it can also cause such rules to be misapplied to the incorrect source range.

Such rules are permitted in the hierarchical phrase-based extraction heuristic. Since the mixed-syntax model is able to make use of decorated non-terminals, we attempt to reduce the opportunity for translation rules to be misapplied by anchoring them more

firmly to the source range with lexical and decorated non-terminals. Therefore, the extraction heuristic is refined to allow only decorated non-terminals in the first and last source position, not undecorated non-terminals. Translation rules such as the one below are now excluded from the grammar.

$$\langle X \rightarrow \text{mir } X_1 \quad \# \quad X \rightarrow \text{I am } X_1 \rangle$$

Lastly, we model long-range dependency by allowing decorated non-terminals to cover an unlimited number of words. Decorated non-terminals can only be specific application, they can only be applied where they match the label of the source span. Therefore, widening the span limit for decorated non-terminal does not increase spurious ambiguity or decrease decoding speed. Undecorated non-terminals are still constrained by the span limit.

### 4.8.2 New Mixed-Syntax Extraction Algorithm

The new mixed-syntax extraction algorithm extract translation rules from parallel sentences in three stages. The first stage identifies initial, fully lexicalized phrases. This is based on the phrase-based model extraction heuristics and is identical to that used in the hierarchical extraction of Section 4.3.1, however, there is no size limit for initial phrases. The source syntactic label of the entire span, if any, is also record.

The second stage creates a lattice from each source word and initial phrases. Each vertex in the lattice correspond to a word position in the source sentence. Each arc correspond to a symbol in the source, either a word or a non-terminal. Arcs connect the start and end position of each symbol. The outline of the algorithm is shown in Figure 4.16.

The result is a lattice consisting of a set of labelled arcs,  $A_{n,m}$ , each spanning the range  $n$  to  $m$ . The label of each arc is a word from the source sentence or a constituent label from the source parse.

The third stage extract rules by iteratively traversing the lattice, Figure 4.17. The two predicates, ‘*is a valid intermediate rule*’ and ‘*is a valid final rule*’, replaces the filtering constrains of the original extraction algorithm.

A rule *is a valid intermediate rule* if there is a possibility a final rule can be created by appending a source edge to it. Intermediate rules have the following constraints:

1. Contains less than 5 source terminal symbols
2. No undecorated source symbols are adjacent on the source side

---

**Require:** source and target sentence,  $s_{1:|s|}$  and  $t_{1:|t|}$ , source labels  $\{V_{i,j}\}^*$  where  $0 \leq i < |s|$  and  $i + 1 < j \leq |s|$

1<sup>st</sup> stage. Identify fully lexicalized rules

$F \leftarrow$  all fully lexicalized rules

2<sup>nd</sup> stage. Create lattice

**for all**  $n, m$  where  $0 \leq n < |s|$  and  $n + 1 < m \leq |s|$  **do**

$A_{n,m} \leftarrow \{\}$

**end for**

**for all** words  $s_n \in s_{1:|s|}$  source sentence **do**

Add an arc with label  $s_n$  to set of arcs  $A_{n,n+1}$

**end for**

**for all** initial phrase  $r \in F$ , with source left-hand-side  $L$ , with source range  $[n, m]$  **do**

Add an arc with label  $L$  to set of arcs  $A_{n,m}$

**end for**

---

Figure 4.16: Identifying and Creating Lattice for Rule Extraction

3. Contains a maximum of 3 non-terminals
4. Undecorated non-terminals span a maximum of 7 source words

The ‘*is a valid final rule*’ predicate ensure rules that are added to the translation grammar have the following additional constraints:

1. First and last source symbols are not undecorated non-terminals
2. Entire rule span an initial phrase
3. Number of undecorated non-terminals is less than number of terminals on the source side
4. Is not composed of a unary non-terminal on the source side

The example sentence of Figure 4.18 result in 1133 translation rules when extracted with the new heuristic, compared to 1965 rules using the old heuristics.

The old heuristics extracted rules such as those below, containing undecorated non-terminals in the first or last source position, or unlexicalized rules with undecorated



---

```

output grammar  $G \leftarrow \{\}$ 
for all  $0 \leq i < |s|$  do
    Create empty rule  $r$  starting and ending at  $i$ 
    Create translation rules from previous rule  $r$ 
end for
return  $G$ 

```

---

```

Function Create translation rules from previous rule  $r_{in}$ 
start  $\leftarrow$  first source position of rule  $r_{in}$ 
end  $\leftarrow$  last source position of rule  $r_{in}$ 
for all  $end < i \leq |s|$  and  $L \in A_{end,i}$  do
     $r_{out} \leftarrow$  append label  $L$  to  $r_{in}$ , spanning start to  $i$ 
    if  $r_{out}$  is valid intermediate rule then
        Create translation rules from previous rule  $r_{out}$ 
    end if
    if  $r_{out}$  is valid final rule then
        for all non-terminals  $B \in A_{start,i}$  do
             $r_{valid} \leftarrow r_{out}$ , with source left-hand-side  $B$ 
            Add  $G \leftarrow r_{valid}$ 
        end for
    end if
end for

```

---

Figure 4.17: Creating Mixed-Syntax Translation Rules

non-terminals.

$$\langle X \rightarrow \text{liegt daran } X_1 \quad \# \quad X \rightarrow \text{keen } X_1 \rangle$$

$$\langle X \rightarrow NN_1 X_2 \quad \# \quad X \rightarrow X_1 X_2 \rangle$$

In contrast, the new heuristics does not extract these rules but have extracted rules from large spans which model long-range patterns.

$$\langle X \rightarrow CNP_1 NP_2 \quad \# \quad X \rightarrow X_1 X_2 \rangle$$

$$\langle TOP \rightarrow S_1 . \quad \# \quad X \rightarrow X_1 . \rangle$$



### 4.8.3 Results

Table 4.11 shows the number of translation rules extracted from the aligned training corpora.

	Extraction Algorithm	Number of rules
1	Hierarchical	500m
2	Mixed-syntax (original)	2,664m
3	Mixed-syntax (new)	1,435m
4	Mixed-syntax (new, allow edge undecorated non-terminals)	2,104m

Table 4.11: Number of translation rules extracted for each model

Re-evaluating the mixed-syntax model with the new grammar shows markedly improved results in both in and out-of-domain test sets, outperforming the hierarchical phrase-based baseline, Table 4.12.

	Model	In-Domain	Out-of-domain
1	Hierarchical	22.1	16.5
2	Mixed-syntax (old)	21.6	16.3
3	Mixed-syntax (new)	22.7	<b>17.8</b>
4	Mixed-syntax (new, allow edge undecorated non-terminals)	<b>22.9</b>	17.7

Table 4.12: German–English results, using new extraction algorithm, in %BLEU

Varying the maximum span limit for decorated non-terminal during decoding, a small but positive gain in translation quality can be seen when the decoder is allowed to apply translation rules to longer range spans, Table 4.13.

## 4.9 Conclusion

We have presented in this chapter a novel model which combines the generality of the hierarchical phrase-based approach with the specificity of a tree-to-string model. The model is able to take advantage of linguistic information in the input parse and training corpus, without being rigidly constrained by syntax.

Max. span for decorated non-terminals	In-Domain
3	22.6
7	22.6
10	22.6
15	<b>22.7</b>
20	<b>22.7</b>
unlimited	<b>22.7</b>

Table 4.13: Effect of varying max. span limit on translation quality, in %BLEU

Using the specificity of syntactic information allows the model to expand the rule forms that can be extracted and used during decoding, while maintaining tractability. The expanded grammar enables the translation model to better explain more of the translation, without relying on the empirically uninformed glue rules, leading to improved translation.

# Chapter 5

## Conclusion and Future Work

This thesis has considered some uses of linguistic information in current statistical machine translation models.

Chapter 2 described the *factored phrase-based model* which extends the phrase-based model to incorporate linguistic information as additional factors in the representation of words. We also show how the factored translation model can be decomposed into multiple steps where each step outputs a subset of the target factors, conditioned on a subset of the available source or target factors.

We see how the factored model aids in the disambiguation of source words and improves grammaticality with sequence models over target factors. We also see the factored decomposition can be configured to improve coverage of previously out-of-vocabulary words. We show that the factored model improves translation over a standard phrase-based mode by as much as 0.9 BLEU for small German-English training corpora, and 0.2 BLEU for larger corpora.

The factored model is a general framework which allows any word-level information to be integrated into a phrase-based model. It has also been used as the basis of other research (Avramidis and Koehn, 2008; Cettolo et al., 2008; Yeniterzi and Oflazer, 2010) which focuses on specific uses of linguistic information for particular language pairs.

In Chapter 3, we extended the factored model to the *factored template model*. This model generalizes translation with part-of-speech tags to improve reordering. We see that the factored template model performs poorly on its own but when combined with a standard factored phrase-based model, performance improves by as much as 1.1 BLEU on a small French-English system. We also see that when using the combined model, the linguistically motivated template reordering model is overwhelmingly preferred

over the distance-based reordering model.

Both the factored model and factored template model, however, are wedded to word-level information which limits their usefulness for improving long-range dependency. In Chapter 4, we extended the word-level information to using information on continuous spans. We also transition from the phrase-based model to models based on synchronous context-free grammar which also offer a natural fit for multi-word linguistic annotation. We described a novel tree-to-string model, the *mixed-syntax model*, which combines the specificity of syntactic models and the generality of the hierarchical phrase-based model. This model uses source language syntactic information to inform translation.

We show that the model is able to explain translation better, leading to a 0.8 BLEU improvement over the baseline hierarchical phrase-based model for a small German-English task. Also, the mixed-syntax model requires only labels on continuous source spans, it is not dependent on a tree structure, therefore, other types of syntactic information can be integrated into the model. We experimented with a shallow parser and see a gain of 0.5 BLEU for the same dataset.

Experimenting with large training corpora shows the difficulty of improving over the linguistically uninformed baseline by the factored model and factored template model. The word-level integration of linguistic information into these models restrict the influence of such information to a small window. The availability of more training data diminishes the advantage of using word-level information in more linguistically motivated models.

Applying large training data to the mixed-syntax showed that translation performance actually decrease, relative to the linguistically unmotivated baseline, due to spurious ambiguity caused by the extraction of translation rules with overlapping functionality. We therefore changed the grammar filtering heuristic to rebalance our desire to take advantage of the mixed-syntax model to extract useful translation, with the need to contain spurious ambiguity. This substantially reduced the size of the grammar but increased translation quality of 0.6 BLEU (1.3 out-of-domain) over the baseline.

The core finding of this thesis is that attempting to rigidly model translation as a linguistic transfer results in degraded performance. However, by combining the principles of empirically-motivated statistical models with linguistically-motivated models, we are able to achieve better translation performance.

## 5.1 Future Work

The introduction of phrase-based models into statistical machine translation has brought with them a range of techniques that have greatly enhanced the field. The log-linear model, discriminative training, phrase-extraction from word alignments, beam search are some of the innovations which were started or popularized by these models. Phrase-based models still have an important place for their simplicity, efficiency and low resource consumption.

However, formally linguistic approaches based on formalisms such as a synchronous context-free-grammar, tree transduction, and synchronous tree-substitution grammar discussed in Chapter 4 are undoubtedly the future of general purpose, state-of-the-art machine translation. More powerful formalisms allow for more linguistic constraints. In future work, we would like to explore the range of formalisms available to assess their strengths and weaknesses, and their suitability for machine translation.

With the mixed-syntax model, we integrated source language syntactic information into the hierarchical phrase-based model to create a model that has the advantage of both syntactic and non-syntactic approaches. We would like to extend this in a number of ways.

Firstly, the mixed-syntax model have hitherto only made use of source syntax information but does not enforce grammaticality on the output. Incorporating target syntactic information into this model would be the next logical step.

Secondly, we have only used one source of syntactic information at a time in the mixed-syntax model. However, we saw that using parse tree labels and shallow chunk tags independently improves translation. This contrasts with the factored model of Chapter 2 where multiple linguistic information such as POS tags, morphology and stems were often used simultaneously. We would like to study models which can incorporate such heterogenous data in a syntactic model.

Thirdly, we have so far relied on linguistic information from external tools. This brings with it a number of disadvantages, such as reliance of language-dependent tools, their reliability and appropriateness for machine translation. It would be interesting to develop annotations that can be used with the mixed-syntax model but are specifically tuned for the task of translation.

Lastly, our experimentation with different formalisms, models and heuristics has shown there are many similarities between the phrase-based, hierarchical and syntax models. In fact, systems based on phrase-based, hierarchical or syntax models can be

viewed as identical, apart from the translation model. The translation model sits at the core of any translation system, its purpose is to propose translations which other models in the system can then rank. Therefore, it is important to ensure that the translation model is able to learn from the training data and it has good inductive bias. That is, given a previously unseen input sentence, a good translation model should propose more good translations and less bad ones.

This is the simple, underlying reason for the choices we make when deciding which formalism, model, heuristic, rule set, to choose. Whether and how we use linguistic information, as with the other choices, is a pragmatic decision based on our domain knowledge of how it will improve the bias of the translation model. In future work, we would like to quantify the affect that these decisions have on the translation model and how this ultimately affect translation quality.



# Bibliography

- Abney, S. (1991). Parsing by chunks. In *Robert Berwick, Steven Abney, and Carol Tenny: Principle-Based Parsing*. Kluwer Academic Publishers.
- Al-Onaizan, Y., Cuřín, J., Jahr, M., Knight, K., Lafferty, J. D., Melamed, I. D., Och, F.-J., Purdy, D., Smith, N. A., and Yarowsky, D. (1999). Statistical machine translation. Technical report, John Hopkins University Summer Workshop <http://www.clsp.jhu.edu/ws99/projects/mt/>.
- Ambati, V. and Lavie, A. (2008). Improving syntax driven translation models by restructuring divergent and non-isomorphic parse tree structures. In *AMTA*.
- Avramidis, E. and Koehn, P. (2008). Enriching morphologically poor languages for statistical machine translation. In *Proceedings of ACL-08: HLT*, pages 763–770, Columbus, Ohio. Association for Computational Linguistics.
- Bilmes, J. and Kirchhoff, K. (2003). Factored language models and generalized parallel backoff. In *Proceedings of HLT/NAACL 2003*.
- Birch, A., Blunsom, P., and Osborne, M. (2009). A Quantitative Analysis of Reordering Phenomena. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 197–205, Athens, Greece. Association for Computational Linguistics.
- Blunsom, P. and Osborne, M. (2008). Probabilistic inference for machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 215–223, Honolulu, Hawaii. Association for Computational Linguistics.
- Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4).

- Brown, P. F., Della-Pietra, S. A., Della-Pietra, V. J., and Mercer, R. L. (1993). The mathematics of statistical machine translation. *Computational Linguistics*, 19(2):263–313.
- Carme Armentano-Oller, C., Corbí-Bellot, A. M., Forcada, M. L., Ginestí-Rosell, M., Boney, B., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Ramírez-Sánchez, G., and Sánchez-Martínez, F. (2005). An open-source shallow-transfer machine translation toolbox: consequences of its release and availability. In *OSMaTran: Open-Source Machine Translation, A workshop at Machine Translation Summit X*, pages 23–30.
- Cettolo, M., Marcello, F., Daniele, P., and Nicola, B. (2008). Shallow-syntax phrase-based translation: Joint versus factored string-to-chunk models. In *AMTA*.
- Chappelier, J.-C., Rajman, M., and Ch-Lausanne (1998). A generalized cyk algorithm for parsing stochastic cfg.
- Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, Michigan. Association for Computational Linguistics.
- Chiang, D. (2006). An introduction to synchronous grammars.
- Chiang, D. (2007). Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- Chiang, D. (2010). Learning to translate with source and target syntax. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1443–1452, Uppsala, Sweden. Association for Computational Linguistics.
- Chiang, D., Knight, K., and Wang, W. (2009). 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 218–226, Boulder, Colorado. Association for Computational Linguistics.
- Chiang, D., Marton, Y., and Resnik, P. (2008). Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Honolulu, Hawaii. Association for Computational Linguistics.

- Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585.
- DeNero, J., Pauls, A., and Klein, D. (2009). Asynchronous binarization for synchronous grammars. In *ACL-IJCNLP '09: Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 141–144, Morristown, NJ, USA. Association for Computational Linguistics.
- Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In Matsumoto, Y., editor, *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pages 205–208.
- Foster, G., Kuhn, R., and Johnson, H. (2006). Phrasetable smoothing for statistical machine translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 53–61, Sydney, Australia. Association for Computational Linguistics.
- Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia. Association for Computational Linguistics.
- Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What’s in a translation rule? In *Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL)*.
- Green, S., Galley, M., and Manning, C. D. (2010). Improved models of distortion cost for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 867–875, Los Angeles, California. Association for Computational Linguistics.
- Hoang, H. and Koehn, P. (2008). Design of the moses decoder for statistical machine translation. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 58–65, Columbus, Ohio. Association for Computational Linguistics.

- Hoang, H. and Koehn, P. (2009). Improving mid-range re-ordering using templates of factors. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 372–379, Athens, Greece. Association for Computational Linguistics.
- Hoang, H. and Koehn, P. (2010). Improved translation with source syntax labels. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 409–417, Uppsala, Sweden. Association for Computational Linguistics.
- Hoang, H., Koehn, P., and Lopez, A. (2009). A Unified Framework for Phrase-Based, Hierarchical, and Syntax-Based Statistical Machine Translation. In *Proc. of the International Workshop on Spoken Language Translation*, pages 152–159, Tokyo, Japan.
- Huang, L. (2007). Binarization, synchronous binarization, and target-side binarization. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 33–40, Rochester, New York. Association for Computational Linguistics.
- Huang, L. (2008). *Dynamic Programming in Semiring and Hypergraph Frameworks*. PhD thesis, University of Pennsylvania.
- Huang, L. and Chiang, D. (2007). Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic. Association for Computational Linguistics.
- Huang, L., Knight, K., and Joshi, A. (2006a). Statistical syntax-directed translation with extended domain of locality. In *5th Conference of the Association for Machine Translation in the Americas (AMTA)*, Boston, Massachusetts.
- Huang, L., Knight, K., and Joshi, A. (2006b). A syntax-directed translator with extended domain of locality. In *Proceedings of the Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, pages 1–8, New York City, New York. Association for Computational Linguistics.
- Huang, L., Zhang, H., Gildea, D., , and Knight, K. (2009). Binarization of synchronous context-free grammars. *Computational Linguistics*.

- Hutchins, W. J. (2000). *Early years in machine translation: memoirs and biographies of pioneers*. Amsterdam.
- Hutchins, W. J. and Somers, H. L. (1992). *An Introduction to Machine Translation*. Academic Press, London.
- Jelinek, F. (1969). Fast sequential decoding algorithm using a stack. *IBM J. Res. Dev.*, 13(6):675–685.
- Jelinek, F. (1998). *Statistical Methods for Speech Recognition*. The MIT Press.
- Kay, M. (1980). Algorithm schemata and data structures in syntactic processing. Technical Report Technical Report CSL-80-12, Xerox PARC, Palo Alto, CA.
- Koehn, P. (2002). Europarl: A multilingual corpus for evaluation of machine translation. Unpublished, <http://www.isi.edu/~koehn/europarl/>.
- Koehn, P. (2004). Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *AMTA*.
- Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the Tenth Machine Translation Summit (MT Summit X)*, Phuket, Thailand.
- Koehn, P., Federico, M., Shen, W., Bertoldi, N., Bojar, O., Callison-Burch, C., Cowan, B., Dyer, C., Hoang, H., Zens, R., Constantin, A., Moran, C. C., and Herbst, E. (2006). Open source toolkit for statistical machine translation. Technical report, Johns Hopkins University.
- Koehn, P. and Hoang, H. (2007). Factored translation models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876.
- Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

- Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase based translation. In *Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL)*.
- Lavie, A. (2008). Stat-xfer: a general search-based syntax-driven framework for machine translation. In *CICLing'08: Proceedings of the 9th international conference on Computational linguistics and intelligent text processing*, pages 362–375, Berlin, Heidelberg. Springer-Verlag.
- Li, Z., Callison-Burch, C., Dyer, C., Khudanpur, S., Schwartz, L., Thornton, W., Weese, J., and Zaidan, O. (2009). Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 135–139, Athens, Greece. Association for Computational Linguistics.
- Liberato, F., Mohit, B., and Hwa, R. (2010). Improving phrase-based translation with prototypes of short phrases. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 301–304, Los Angeles, California. Association for Computational Linguistics.
- Liu, Y., Huang, Y., Liu, Q., and Lin, S. (2007). Forest-to-string statistical translation rules. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 704–711, Prague, Czech Republic. Association for Computational Linguistics.
- Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-string alignment template for statistical machine translation. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 609–616, Morristown, NJ, USA. Association for Computational Linguistics.
- Liu, Y., Lü, Y., and Liu, Q. (2009a). Improving tree-to-tree translation with packed forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 558–566, Suntec, Singapore. Association for Computational Linguistics.

- Liu, Y., Mi, H., Feng, Y., and Liu, Q. (2009b). Joint decoding with multiple translation models. In *In Proceedings of ACL/IJCNLP 2009*, pages 576–584, Singapore.
- Manning, C. D. and Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA.
- Marcu, D., Wang, W., Echihiabi, A., and Knight, K. (2006). Spmt: Statistical machine translation with syntactified target language phrases. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 44–52, Sydney, Australia. Association for Computational Linguistics.
- Marton, Y. and Resnik, P. (2008). Soft syntactic constraints for hierarchical phrasal-based translation. In *Proceedings of ACL-08: HLT*, pages 1003–1011, Columbus, Ohio. Association for Computational Linguistics.
- Mi, H. and Huang, L. (2008). Forest-based translation rule extraction. In *EMNLP*, Honolulu, Hawaii.
- Mi, H., Huang, L., and Liu, Q. (2008). Forest-based translation. In *Proceedings of ACL-08: HLT*, pages 192–199, Columbus, Ohio. Association for Computational Linguistics.
- Moore, R. C. and Quirk, C. (2007). Faster beam-search decoding for phrasal statistical machine translation. In *Proceedings of MT Summit XI*.
- Ney, H. (1992). A comparative study of two search strategies for connected word recognition: Dynamic programming and heuristic search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14:586–595.
- Ney, H. and Ortmanns, S. (1997). Progress in dynamic programming search for lvsr. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 287–294, Sta. Barbara, CA.
- Nießen, S. and Ney, H. (2000). Improving smt quality with morpho-syntactic analysis. In *Proceedings of the 18th conference on Computational linguistics*, pages 1081–1085, Morristown, NJ, USA. Association for Computational Linguistics.
- Nießen, S. and Ney, H. (2004). Statistical machine translation with scarce resources using morpho-syntactic information. *Comput. Linguist.*, 30(2):181–204.

- Och, F. J. (2003). Minimum error rate training for statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association of Computational Linguistics (ACL)*.
- Och, F. J., Gildea, D., Khudanpur, S., Sarkar, A., Yamada, K., Fraser, A., Kumar, S., Shen, L., Smith, D., Eng, K., Jain, V., Jin, Z., and Radev, D. (2004). A smorgasbord of features for statistical machine translation. In *Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL)*.
- Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–52.
- Och, F. J. and Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*.
- Ortiz-Martínez, D., García-Varea, I., and Casacuberta, F. (2006). Generalized stack decoding algorithms for statistical machine translation. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 64–71, New York City. Association for Computational Linguistics.
- Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2001). BLEU: a method for automatic evaluation of machine translation. Technical Report RC22176(W0109-022), IBM Research Report.
- Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*.
- Poutsma, A. (2000). Data-oriented translation. In *Proceedings of the 18th conference on Computational linguistics*, pages 635–641, Morristown, NJ, USA. Association for Computational Linguistics.
- Satta, G. and Peserico, E. (2005). Some computational complexity results for synchronous context-free grammars. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 803–810, Morristown, NJ, USA. Association for Computational Linguistics.



- Schmid, H. (1994). Probabilistic part-of-speech tagger using decision trees. In *International Conference on New methods in Language Processing*.
- Schmidt, H. and Schulte im Walde, S. (2000). Robust German noun chunking with a probabilistic context-free grammar. In *Proceedings of the International Conference on Computational Linguistics (COLING)*.
- Shen, L., Xu, J., Zhang, B., Matsoukas, S., and Weischedel, R. (2009). Effective use of linguistic and contextual information for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 72–80, Singapore. Association for Computational Linguistics.
- Shieber, S. M., Schabes, Y., and Pereira, F. C. N. (1995). Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1–2):3–36. Also available as cmp-lg/9404008.
- Stolcke, A. (2002). SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*, volume 2, pages 901–904, Denver, USA.
- Tillmann, C. (2004). A unigram orientation model for statistical machine translation. In *Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL)*.
- Tomas, J. and Casacuberta, F. (2003). Combining phrase-based and template-based alignment models in statistical translation. In *IbPRIA*.
- Venugopal, A., Zollmann, A., Smith, N. A., and Vogel, S. (2009). Preference grammars: Softening syntactic constraints to improve statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 236–244, Boulder, Colorado. Association for Computational Linguistics.
- Wang, W., Knight, K., and Marcu, D. (2007). Binarizing syntax trees to improve syntax-based machine translation accuracy. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 746–754.
- Wang, W., May, J., Knight, K., and Marcu, D. (2010). Restructuring, relabeling, and realigning for syntax-based machine translation. *Comput. Linguist.*, 36(2):247–277.

- Wu, D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3).
- Xiao, T., Li, M., Zhang, D., Zhu, J., and Zhou, M. (2009). Better synchronous binarization for machine translation. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 362–370, Morristown, NJ, USA. Association for Computational Linguistics.
- Yamada, K. and Knight, K. (2001). A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association of Computational Linguistics (ACL)*.
- Yamada, K. and Knight, K. (2002). A decoder for syntax-based statistical MT. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*.
- Yeniterzi, R. and Oflazer, K. (2010). Syntax-to-morphology mapping in factored phrase-based statistical machine translation from english to turkish. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 454–464, Uppsala, Sweden. Association for Computational Linguistics.
- Zens, R., Och, F. J., and Ney, H. (2002). Phrase-based statistical machine translation. In *Proceedings of the German Conference on Artificial Intelligence (KI 2002)*.
- Zhang, H., Huang, L., Gildea, D., and Knight, K. (2006). Synchronous binarization for machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 256–263, New York City, USA. Association for Computational Linguistics.
- Zhang, H., Zhang, M., Li, H., Aw, A., and Tan, C. L. (2009). Forest-based tree sequence to string translation model. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1, ACL-IJCNLP '09*, pages 172–180, Morristown, NJ, USA. Association for Computational Linguistics.
- Zhang, M., Jiang, H., Aw, A., Li, H., Tan, C. L., and Li, S. (2008a). A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of ACL-08: HLT*, pages 559–567, Columbus, Ohio. Association for Computational Linguistics.

- Zhang, M., Jiang, H., Li, H., Aw, A., and Li, S. (2008b). Grammar comparison study for translational equivalence modeling and statistical machine translation. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 1097–1104, Morristown, NJ, USA. Association for Computational Linguistics.
- Zollmann, A. and Venugopal, A. (2006). Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 138–141, New York City. Association for Computational Linguistics.