Diplomarbeit im Fach Informatik
Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik 6
Prof. Dr.-Ing. H. Ney

Extending Statistical Machine Translation Using Syntax

3. März 2010

vorgelegt von:
Stephan Peitz
Matrikelnummer 265372

Gutachter:
Prof. Dr.-Ing. H. Ney
Prof. Dr. rer. nat. T. Siedl

Betreuer:
Dipl. Inform. D. Vilar Torres
Dipl. Inform. D. Stein

Erklärung

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet habe. Alle Textauszüge und Grafiken, die sinngemäß oder wörtlich aus veröffentlichten Schriften entnommen wurden, sind durch Referenzen gekennzeichnet.

Aachen, 3. März 2010

Stephan Peitz

Abstract

In statistical machine translation, the best translation for a given sentence is searched. To get a more syntactic well-formed translation, we extend in this work our hierarchical phrase-based translation system with syntax information.

The first method is similar to the *Syntax Augmented Machine Translation System* [Zollmann & Venugopal 06]. The second approach is based on the *Preference Grammars* [Venugopal & Zollmann$^+$ 09].

In addition, we present a new method to create meta syntax information.

The first approach is tested on IWSLT Chinese-English, the second approach on IWSLT Chinese-English, QUAERO German-English and NIST Chinese-English. Further, we start first experiments with the meta syntax information on NIST Chinese-English using the second approach.

Kurzfassung

In der statistisch maschinellen Übersetzung wird die beste Übersetzung für einen gegebenen Satz gesucht. Mit dem Ziel eine syntaktisch korrekte Übersetzung zu erhalten, erweitern wir unser hierarchisch-basiertes Übersetzungssystem mit Information über die Syntax der Zielsprache.

Der erste Ansatz ist mit dem *Syntax Augmented Machine Translation System* [Zollmann & Venugopal 06] zu vergleichen. Der zweite Ansatz basiert auf *Preference Grammars* [Venugopal & Zollmann$^+$ 09].

Zusätzlich präsentieren wir eine neue Methode um Meta-Syntax Informationen zu erzeugen.

Der erste Ansatz wird auf IWSLT Chinesisch-Englisch, der zweite Ansatz auf IWSLT Chinesisch-Englisch, QUAERO Deutsch-Englisch und NIST Chinesisch-Englisch gestestet. Des Weiteren beschreiben wir erste Experimente, die die Meta-Syntax Informationen nutzen. Die Experimente werden mit dem zweiten Ansatz auf NIST Chinesisch-Englisch durchgeführt.

Acknowledgements

I would like to thank all the people who have supported me in the production of this work.

In particular, I express my gratitude towards Prof. Dr.-Ing. H. Ney for giving me the opportunity to write my diploma thesis at the Chair for Computer Science 6 at the RWTH Aachen University.

Further, special thanks go to my supervisors David Vilar Torres and Daniel Stein for their unrelenting support and for always being available for questions, discussions, advice, constructive criticism and jokes even in hard times.

I would also like to thank Prof. Dr. rer. nat. T. Seidl for co-evaluating my thesis.

Further, I would like to thank Markus Freitag for implementing a fast optimization method.

My thanks also go to all workers at the Chair for Computer Science 6 for the friendly atmosphere in particular at the coffee machine.

I thank the administrators of the RBI cluster for their immediate help in case of technical problems and for keeping the computers up and running reliably for the great number of time consuming and computationally demanding experiments I ran to evaluate my work.

Finally, I would like to thank my parents for their emotional and financial support throughout the past years.

Contents

1 Introduction

The challenge of statistical machine translation is to find the best translation with methods from the statistical decision theory. In this work, a translation system is used which derives the translation with the highest probability using a stochastic context-free grammar. To guide the decoding process further, deep syntactic information of the target language is included to the grammar. We hope to get a translation which is more fluent, structured and grammatically correct.

In this thesis, we present two methods to extend our statistical machine translation system with syntax information. For both approaches, we extract the deep syntactic information of the target language before the actual translation.

The first approach is similar to *Syntax Augmented Machine Translation System* [Zollmann & Venugopal 06], although the necessary modification of the decoding process is not described in this paper. Therefore, we develop an own method. The second approach is based on *Preference Grammars* [Venugopal & Zollmann$^+$ 09]. Furthermore, we present a new method to create meta syntax information.

All methods are compared with our standard translation system on different corpora and language pairs. Using the syntax information, some results show an improvement in the translation quality.

1.1 State-of-the-art

The following two methods are state-of-the-art in syntactic incorporation to the hierarchical phrase-based translation approach:

- The Syntax Augmented Machine Translation System includes the syntax information to the grammar directly [Zollmann & Venugopal 06]. This approach has the disadvantage that the decoding process must be extended. Further, it is quite expensive in time and memory.
- The concept of Preference Grammars [Venugopal & Zollmann$^+$ 09] uses the syntax information in an additional translation model. The rules in the grammar are weighted by considering the syntax information. In the decoding process, the model tries to produce a syntactic well-formed translation using

the weights of the rules. The hierarchical phrase-based system can extended easily by this additional model.

- In [Shen & Xu$^+$ 08], a method was presented to use information of dependency trees which reveal long range relations between words. For a given sentence, each word has a parent word which it depends on. Compared with approaches using syntax trees, dependency relations directly model semantic structure to get fluent and structured translation.

The following approaches also works with syntax information, but employ different decoder:

- Yamada and Knight developed a Decoder for Syntax-based Statistical Machine Translation [Yamada & Knight 02] to incorporate phase-based translation approach. Before the decoding process, an additional model is computed. In order to do that, each target sentence is parsed. On the resulting parse trees, they do reorderings and insertions to get the correct source sentences. In the decoding process, the direction is reverse. The given source sentence is parsed and the most plausible parse tree in the target language is search based on the additional model.
- In [Charniak & Knight$^+$ 03] a syntax-based language model was introduced and combined with the syntax-based translation model described above. The language model is a forst of syntax trees. In the decoding process, the parse tree produced by the syntax-based translation model is estimate with the parse trees from the language model.

1.2 Outline

This work is structured as follows: First, we introduce some basics which are state-of-the-art in statistical machine translation in Chapter 2. In Chapter 3 we describe our hierarchical phrase-based translation system in detail. Then, we present the approaches to extend this translation system with syntax information in Chapter 4. In Chapter 5, we show some experiments which compare the extension with the standard translation system. This work is concluded with a summary and an outlook (Chapter 6).

2 Statistical Machine Translation

The goal of machine translation is to produce an automatic translation of a given source sentence in a target language. In the statistical approach, the main idea is to learn translation rules automatically from bilingual training data, to compute the associated probabilities and to build up a translation model without any human interaction. With this kind of translation models, many possible translations are generated for a given sentence. The choice of the best translation is based on statistical decision theory. In the statistical approach, the standard method to choose a translation is the Bayes decision rule.

2.1 Bayes Decision Rule

The Bayes decision rule tells us to choose the target sentence with the highest probability to minimize the sentence error rate. More formally, for a given source sentence $f_1^J = f_1 \dots f_j \dots f_J$ which is to be translated into a target sentence $e_1^I = e_1 \dots e_i \dots e_I$, we choose the translation $\hat{e}_1^{\hat{I}}$ which maximizes the posterior probability $Pr(e_1^I|f_1^J)$:

$$\hat{e}_1^{\hat{I}} = \arg\max_{e_1^I} \left\{ Pr(e_1^I|f_1^J) \right\} \tag{2.1}$$

It is one of the main challenges in statistical machine translation to properly define a probability distribution $Pr(e_1^I|f_1^J)$ and to estimate it from examples of bilingual training data. One possibility is to rewrite the probability distribution by using Bayes theorem:

$$\begin{aligned}
\hat{e}_1^{\hat{I}} &= \arg\max_{e_1^I} \left\{ Pr(e_1^I|f_1^J) \right\} && (2.2)\\
&= \arg\max_{e_1^I} \left\{ \frac{Pr(e_1^I) Pr(f_1^J|e_1^I)}{Pr(f_1^J)} \right\} && (2.3)\\
&= \arg\max_{e_1^I} \left\{ Pr(e_1^I) Pr(f_1^J|e_1^I) \right\} && (2.4)
\end{aligned}$$

The probability distribution is decomposed into a translation model $Pr(f_1^J|e_1^I)$, which models the correspondence between source and target sentence, and a language model $Pr(e_1^I)$, which tries to produce a well formed target sentence.

2.2 Log-linear Model

An alternative to the decomposition is to directly model $Pr(e_1^I|f_1^J)$ [Och & Ney 02]. The *log-linear model* considers different models $h_m(f_1^J, e_1^I)$ called features. Each feature is multiplied with a corresponding scaling factor λ_m.

$$Pr(e_1^I|f_1^J) = \frac{\exp\left(\sum_{m=1}^M \lambda_m h_m(f_1^J, e_1^I)\right)}{\sum_{\tilde{e}_1^I} \exp\left(\sum_{m=1}^M \lambda_m h_m(f_1^J, \tilde{e}_1^I)\right)} \tag{2.5}$$

Using a log-linear model is more flexible, because it is easier to extend with additional features. These features could be probability distributions like the language model and the translation model or simple heuristics. With the log-linear model, the Bayes decision rule is rewritten:

$$\begin{aligned}
\hat{e}_1^{\hat{I}} &= \arg\max_{e_1^I} \left\{ Pr(e_1^I|f_1^J) \right\} && (2.6)\\
&= \arg\max_{e_1^I} \left\{ \frac{\exp\left(\sum_{m=1}^M \lambda_m h_m(f_1^J, e_1^I)\right)}{\sum_{\tilde{e}_1^I} \exp\left(\sum_{m=1}^M \lambda_m h_m(f_1^J, \tilde{e}_1^I)\right)} \right\} && (2.7)\\
&= \arg\max_{e_1^I} \left\{ \exp\left(\sum_{m=1}^M \lambda_m h_m(f_1^J, e_1^J) \right) \right\} && (2.8)\\
&= \arg\max_{e_1^I} \left\{ \sum_{m=1}^M \lambda_m h_m(f_1^J, e_1^J) \right\} && (2.9)
\end{aligned}$$

To learn the scaling factors λ_m of the log-linear model, we use in our work the minimum-error-rate-training (MERT) [Och 03]. This optimization method tries to find parameters which minimize an error rate.

2.3 Phrase-based Translation

In this section, we describe the *phrase-based translation* approach [Zens & Och$^+$ 02]. This is a standard approach in statistical machine translation and the starting point

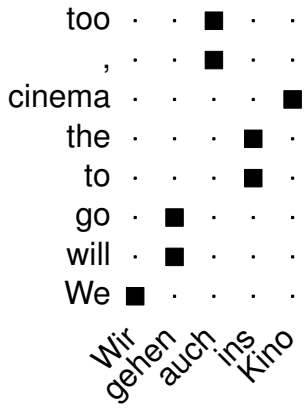

Figure 2.1. Alignment of a sentence

for the work in this diploma theses, because this approach is extended in Chapter 3.

The first approaches to SMT translated each word by itself and did not take contextual information into account. The phrase-based translation approach considers the local context using *bilingual phrases*. A bilingual phrase is a pair formed by a sequence of words of a source sentence and its translation and are extracted from a word aligned corpus.

An *alignment* is a mapping from several words of a source sentence to several words of the target sentence. For given e_1^I, f_1^J and an alignment $\mathcal{A} \subseteq \{1 \dots J\} \times \{1 \dots I\}$, the word f_j is aligned to e_i iff

$$(j, i) \in \mathcal{A}.$$

As example can be seen in Figure 2.1. An alignment with reasonable quality can be automatically computed with e.g. GIZA++ [Och & Ney 00] from the training data. The resulting alignment based on the IBM1-models [Brown & Cocke$^+$ 90] and describe the correspondence between single words of a source sentence to single words of a target sentence. To get alignments as defined above, we create alignments for both translation directions with GIZA++ and merge them with a heuristic [Och & Ney 03]. With the definition of alignments we can extract a set of phrases $\mathcal{P}$:

$$\begin{aligned}\mathcal{P}(f_1^J, e_1^I, \mathcal{A}) = \big\{(f_{j_1}^{j_2}, e_{i_1}^{i_2}) \,|j_1, j_2, i_1, i_2 \text{ s.t.} \\ \forall (j,i) \in A : j_1 \leq j \leq j_2 \Leftrightarrow i_1 \leq i \leq i_2 \\ \wedge \exists (j,i) \in A : (j_1 \leq j \leq j_2 \wedge i_1 \leq i \leq i_2)\big\}.\end{aligned} \tag{2.10}$$

source phrase	target phrase
Wir	We
Wir gehen	We will go
Wir gehen ins	We will go to the
gehen	will
gehen	will go
gehen	go
gehen auch ins Kino	will go to the cinema , too
auch	,
auch	, too
auch	too
auch ins Kino	to the cinema , too
ins	to
ins	to the
ins	the
ins Kino	to the cinema
Kino	cinema

Figure 2.2. List of extracted phrases

An example set of phrases which are extracted from the alignment in Figure 2.1 can be seen in Figure 2.2.

We define the phrase-based translation model for a given sentence pair (f_1^J, e_1^I) which is segmented into K phrase pairs $(\tilde{f}_1^K, \tilde{e}_1^K)$. K is a hidden variable in the log-linear model.

$$h_{phrasesT2S}(f_1^J, e_1^I, K) = \prod_{k=1}^{K} p(\tilde{f}_k|\tilde{e}_k) \tag{2.11}$$

$$h_{phrasesS2T}(f_1^J, e_1^I, K) = \prod_{k=1}^{K} p(\tilde{e}_k|\tilde{f}_k) \tag{2.12}$$

Two such models are estimated for both directions and use the phrase translation probabilities $p(\tilde{e}|\tilde{f})$ and $p(\tilde{f}|\tilde{e})$.

A further feature is the *word-based lexicon model* which is based on the IBM1-model [Brown & Cocke$^+$ 90]. This feature is another estimation of the probability for a source phrase given a target phrase and vice versa using the word translation probabilities.

Language Model

The language model estimates the probability of the translated sentence in the target language. For that purpose, a general language model considers the history e_1^{i-1} of a word e_i.

$$Pr(e_1^I) = \prod_{i=1}^{I} p(e_i|e_1^{i-1}) \tag{2.13}$$

However, in practice, a standard n-gram language model is used, which does not consider the complete history.

$$h_{LM}(f_1^J, e_1^I, s_1^K) = \prod_{i=1}^{I} p(e_i|e_{i-n+1}^{i-1}) \tag{2.14}$$

For example, we consider a 3-gram and the target sentence "we will go to the cinema". "cinema" does not depend on "we". But "cinema" depends on "to the".

Word and Phrase Penalty

The word penalty equals the number of words and the phrase penalty equals the number of phrases used for the translation.

$$h_{WP}(f_1^J, e_1^I, s_1^K) = I \tag{2.15}$$
$$h_{PP}(f_1^J, e_1^I, s_1^K) = K \tag{2.16}$$

3 Hierarchical Phrase-Based Translation

In this chapter we describe our hierarchical phrase-based system which is an extension of the phrase-based approach described in Section 2.3. Phrase-based systems have limitations considering long range dependencies in a sentence. Furthermore, reorderings of phrases are estimated by an additional model (cf. [Och & Ney 04]). The hierarchical phrase-based translation approach [Chiang 05] uses hierarchical phrases to handle long range dependencies and to include reordering information in the translation model. The main idea is to replace parts of a larger phrase with a "gap" which can then be filled with other phrases. The resulting hierarchical phrase represents the dependency between the words before the gap and the words after the gap. Furthermore, the reordering of phrases is integrated in the translation process and considers the global context.

The translation example from [Chiang 05] in Figure 3.1 illustrates the limitations of the standard phrase reordering and the advantage of hierarchical phrases which make a correct reordering possible and are shown in Figure 3.2.

The hierarchical system uses a synchronous context-free grammar (SCFG) as translation model. After a formal definition of SCFG in Section 3.1, we describe how these rules are learned automatically from bilingual training data in Section 3.2. Then, we present a method for generating translations with a synchronous context-free grammar in Section 3.4.

3.1 Synchronous Context-free Grammar

SCFGs are based on context-free grammars (CFG).

Definition 1 (CFG) *Let $\mathcal{NT}$ be a set of non-terminals, $\mathcal{T}$ a set of terminals, $\mathcal{R}$ a set of rules and S a start symbol, $S \in \mathcal{NT}$. $\mathcal{R}$ consists of rules of the form $X \rightarrow \alpha$ where $X \in \mathcal{NT}$ and $\alpha \in (\mathcal{T} \cup \mathcal{NT})^*$. A CFG is a 4-tuple $(\mathcal{NT}, \mathcal{T}, \mathcal{R}, S)$.*

The rules of a CFG have a non-terminal on the left hand side and a string of non-terminals and terminals on the right hand side.

source sentence: Aozhu shi yu Bei Han you bangjiao de shashu guojia zhiyi
phrase-based translation
$[\text{Aozhu}]_0$ $[\text{shi}]_1$ $[\text{yu}]_2$ $[\text{Beihan}]_3$ $[\text{you}]_4$ $[\text{bangjiao}]_5$ $[\text{de shashu guojia zhiyi}]_6$ $[\text{Australia}]_0$ $[\text{has}]_4$ $[\text{dipl. rels.}]_5$ $[\text{with}]_2$ $[\text{N. Korea}]_3$ $[\text{is}]_1$ $[\text{one of the few countries}]_6$
hierarchical phrase-based translation
[Aozhu] [shi] X [Australia] [is] X [Aozhu] [shi] $[X^{\sim 1}$ zhiyi] [Australia] [is] [one of $X^{\sim 1}$] [Aozhu] [shi] $[[X^{\sim 1}$ de $X^{\sim 2}]$ zhiyi] [Australia] [is] [one of [the $X^{\sim 1}$ that $X^{\sim 2}$]] [Aozhu] [shi] $[[[\text{yu } X'^{\sim 1} \text{ yout } X'^{\sim 2}]$ de $X^{\sim 2}]$ zhiyi] [Australia] [is] [one of [the $X^{\sim 2}$ that [have $X'^{\sim 2}$ with $X'^{\sim 1}$]]] [Aozhu] [shi] $[[[\text{yu [Beihan]}'^{\sim 1} \text{ you [bangjiao]}'^{\sim 2}]$ de [shashu guojia]$^{\sim 2}]$ zhiyi] [Australia] [is] [one of [the [few countries]$^{\sim 2}$ that [have [dipl. rels.]$'^{\sim 2}$ with [N. Korea]$'^{\sim 1}$]]]

Figure 3.1. Example for hierarchical phrase-based and phrase-based translation from Mandarin to English: $X^{\sim i}$ are the gaps of the hierarchical phrases. A gap $X^{\sim i}$ has been replaced by a phrase $[...]^{\sim i}$. For the phrase-based translation we denote with $[..]_i$ the connection between the source and target phrase.

source phrase	target phrase
yu $X^{\sim 1}$ you $X^{\sim 2}$	have $X^{\sim 2}$ with $X^{\sim 1}$
$X^{\sim 1}$ de $X^{\sim 2}$	the $X^{\sim 1}$ that $X^{\sim 2}$
$X^{\sim 1}$ zhiyi	one of $X^{\sim 1}$

Figure 3.2. Hierarchical phrases

$$X \rightarrow \langle \text{yu } X^{\sim 1} \text{ you } X^{\sim 2}, \text{have } X^{\sim 2} \text{ with } X^{\sim 1} \rangle$$
$$X \rightarrow \langle X^{\sim 1} \text{ de } X^{\sim 2}, \text{the } X^{\sim 2} \text{ that } X^{\sim 1} \rangle$$

Figure 3.3. Example of rules

With this grammar, we can describe a language. For translation we need a grammar which generates a language pair. Therefore, we introduce synchronous context-free grammars.

Definition 2 (Synchronous CFG) *Let $\mathcal{T}_f$ be a set of terminals of the source language and $\mathcal{T}_e$ a set of terminals of the target language. $\mathcal{R}$ consists of rules of the form $X \rightarrow \langle \alpha, \beta, \sim \rangle$ where $X \in \mathcal{NT}$, $\alpha \in (\mathcal{T}_f \cup \mathcal{NT})^*$ and $\beta \in (\mathcal{T}_e \cup \mathcal{NT})^*$. $\sim$ is a one-to-one correspondence between the non-terminals of α and β.*

Figure 3.3 shows some rules of a SCFG. These rules make the reordering in Figure 3.1 possible. The correspondence between the non-terminals is denoted implicitly with $^{\sim 1}$ and $^{\sim 2}$.

A translation can be derived from a given source sentence with rules from a synchronous CFG. A derivation d is an application of rules of the grammar. A derivation begins with the start symbol S and at each step, the corresponding non-terminals on the right hand side of the rules are rewritten by applying rules of the grammar.

3.2 Rule Extraction

In this section, we describe the extraction of rules for a SCFG which is designed to parse every given source sentence. Therefore, we use only a generic non-terminal X and a start symbol S. To build such a SCFG, we start with the extraction of a set of bilingual phrases $\mathcal{P}$ from a bilingual training corpus based on the definition of phrases. Each phrase of $\mathcal{P}$ is transformed into a *lexical rule* in the synchronous CFG. Those rules have a non-terminal on the left hand side and only terminals on the right hand side.

Definition 3 (Lexical Rule) *Let $(\tilde{f}, \tilde{e})$ be a phrase in $\mathcal{P}$ and $\mathcal{NT} = \{X, S\}$. Then $X \rightarrow \langle (\tilde{f}, \tilde{e}) \rangle$ is a lexical rule in $\mathcal{R}$.*

$$r_8 : X \rightarrow \langle \text{Ins Kino zu gehen}, \text{going to the cinema} \rangle$$
$$r_{38} : X \rightarrow \langle \text{meiner Meinung nach }, \text{In my opinion} \rangle$$
$$r_{51} : X \rightarrow \langle \text{eine gute Idee}, \text{a good idea} \rangle$$

Figure 3.4. Lexical rules

$$r_{20} : X \rightarrow \langle \text{Ins Kino } X^{\sim 1} \text{ ist } X^{\sim 2}, X^{\sim 2}, X^{\sim 1} \text{ to the cinema is} \rangle$$
$$r_{40} : X \rightarrow \langle \text{meiner Meinung } X^{\sim 1}, X^{\sim 1} \text{ my opinion} \rangle$$
$$r_{91} : X \rightarrow \langle X^{\sim 1} \text{ ist } X^{\sim 2}, X^{\sim 2} X^{\sim 1} \text{ is} \rangle$$

Figure 3.5. Hierarchical rules

Figure 3.4 shows a subset of lexical rules which are generated by using the phraseset in Figure A.2 on page 50.

Next, we extract the *hierarchical phrases* using $\mathcal{P}$. Whenever we find a phrase containing a sub-phrase also in $\mathcal{P}$, we replace this sub-phrase by a non-terminal. With the hierarchical phrases we can define the *hierarchical rules* in the synchronous CFG.

Definition 4 (Hierarchical Rule) *Let $(\tilde{f}, \tilde{e})$ be a sub-phrase contained in a phrase $(\alpha_1 \tilde{f} \alpha_2, \beta_1 \tilde{e} \beta_2)$. Then $X \rightarrow \langle \alpha_1 X^{\sim 1} \alpha_2, \beta_1 X^{\sim 1} \beta_2 \rangle$ is a hierarchical rule in $\mathcal{R}$ with one non-terminal on the right hand side.*

Further gaps can be generated iterative if the hierarchical phrases contains further sub-phrases. In contrast to lexical rules, hierarchical rules have non-terminals on the right hand side. These non-terminals are replaced with a lexical rule or another hierarchical rule in a derivation. Hence, translations are possible which consider long range dependencies and reordering of phrases.

Figure 3.5 shows a subset of hierarchical rules which are generated by using the phraseset in Figure A.2 on page 50.

In our hierarchical phrase-based system, we limit the number of non-terminals in the set $\mathcal{NT}$ to two. The non-terminal X is on the left hand side of each rule, which is generated as described in the extraction process above. X replaces the sub-phrases on the right hand side of a hierarchical rule. The non-terminal S is the

$$S \rightarrow \langle S^{\sim 1} X^{\sim 2}, S^{\sim 1} X^{\sim 2} \rangle$$
$$S \rightarrow \langle X^{\sim 1}, X^{\sim 1} \rangle$$

Figure 3.6. Glue Rules

$$\begin{aligned} & S \\ \Rightarrow & \langle X^{\sim 1}, X^{\sim 1} \rangle \\ \Rightarrow & \langle \text{Wir } X^{\sim 1}, \text{We } X^{\sim 1} \rangle \\ \Rightarrow & \langle \text{Wir } X^{\sim 1} \text{ ins Kino}, \text{We } X^{\sim 1} \text{ to the cinema} \rangle \\ \Rightarrow & \langle \text{Wir gehen ins Kino}, \text{We will go to the cinema} \rangle \end{aligned}$$

Figure 3.7. Example of a possible derivation

start symbol of the synchronous CFG. Further rules must be added which map the start symbol to other non-terminals.These rules are called *glue rules* and are shown in Figure 3.6. We also restrict the maximum number of non-terminals on the right hand side to two which is a trade-off between the complexity while decoding and the possibility to handle long range dependencies and reorderings. In Figure 3.7 we see a possible derivation of our hierarchical phrase-based system.

Furthermore, we use a *weighted* SCFG. In our hierarchical phrase-based system each rule in the SCFG get a probability as weight. The probability for a lexical rule equals to the phrase translation probability. The probability for a hierarchical rule is estimated by a relative frequency.

3.3 Features

In our hierarchical phrase-based system, we use as standard features the phrase-based translation models, the IBM1 models, the language models, the word penalty and the phrase penalty. Further, three binary features are included. This features are applied on each rule r_i used in a derivation d:

$$h_m(d) \quad := \quad \sum_{i}^{|d|} h_m(r_i) \tag{3.1}$$

The feature $h_{isHier}(r_i)$ equals one if the given rule r_i is hierarchical. In this case, the feature $h_{isPaste}(r_i)$ checks if r_i has a non-terminal at the first or last position of the right hand side on both languages. The third feature $h_{glueRules}(r_i)$ is true if the given rule is a glue rule.

3.4 Decoding

To generate the best translation for a given source sentence, we consider the generation of a translation as probabilistic parsing procedure. We use the *CYK+* algorithm [Chappelier & Rajman 98] to build up a parse tree of the source sentence which induces a parse tree with the highest probability in the target language. The probability is a product of the probabilities of the applied rules. The induced parse tree is used to get the best translation.

Including the language model in the decoding process produces a forest of parse trees. In our work, we use the *Cube Prune* algorithm to search in the resulting forest of parse trees for the best derivation. The algorithm is beyond the scope of this work and we refer to [Chiang 05]. In this work, we will focus on the CYK+ algorithm. In Section 4.2.2, we extend the CYK+ algorithm to parse with an augmented grammar.

3.4.1 Parsing with CYK+

The CYK+ algorithm [Chappelier & Rajman 98] is based on the CYK algorithm which decides if an input string is generated by a given CFG. For the CYK algorithm, the given CFG must be in Chomsky normal form (CNF). Therefore, either one terminal or two non-terminals are allowed on the right hand side of each rule. But the extraction process of a hierarchical system does not produce such a CFG. The lexical rules may have more than one terminal on the right hand side and the hierarchical rules consists of terminals and non-terminals. Transformating a CFG in CNF produces many rules which must be considered while decoding. This causes a high runtime.

Therefore, we use the CYK+ algorithm which handles a subclass of CFG called *non partially lexicalized*-CFG (nplCFG). In such a grammar, terminals only appear in lexical rules. The terminals in the hierarchical rules are transformed to the *lexical non-terminals*.

Before we go into detail, we define a CYK+ chart based on the definition of a standard CYK chart and give an overview about the algorithm.

For a given input sentence f_1^J, the *CYK+ chart* is a table with $\frac{J(J+1)}{2}$ cells. Each cell (i, j) spans the substring f_j^{j+i-1} where j is the starting position and $j + i - 1$ the length. A cell (i, j) contains two lists of items:

- the type-1 list contains non-terminals Y that parse a substring f_j^{j+i-1}. We write this as a derivation $Y \overset{*}{\Rightarrow} f_j^{j+i-1}$
- the type-2 list represent partial parses of f_j^{j+i-1} for which there is a rule $Y \rightarrow \alpha_1\alpha_2$ with $\alpha_1 \overset{*}{\Rightarrow} f_j^{j+i-1}$ and a non-empty string of non-terminals α_2. This will be denoted by $\alpha_1\bullet$

The parsing step of CYK+ is also divided into two procedures: the standard cell filling procedure which is similar to the parsing step of the CYK algorithm and the self-filling procedure. The input sentence is generated by the grammar if the start symbol of the grammar is found in the type-1 list of cell $(J, 1)$. Algorithm 1 gives an overview of the CYK+ algorithm.

Algorithm 1: CYK+

```
Data: a nplCFG and an input sentence f_1^J
Result: parse tree of f_1^J
conversion in nplCFG;
initialization step;
for i ← 1 to J do
    for j ← 1 to J do
        standard cell filling procedure;
        self-filling procedure;
```

Conversion in nplCFG

Before applying this algorithm the CFG must be transformed to a nplCFG which consists of *non partially lexicalized*-rules. To transform a CFG to a nplCFG, every terminal in a hierarchical rule must be replaced by a new lexical non-terminal (Algorithm 2).

We have only either terminals or non-terminals on the right hand side of each rule and the CYK+ algorithm can be applied.

Initialization Step

With the initialization step the type-1 lists are filled. For each word or phrase of the input sentence, we search for a suitable lexical rule. The left hand side non-terminal

Algorithm 2: Conversion in nplCFG

Data: a CFG with a set of rules $\mathcal{R}$
Result: a nplCFG
for each *hierarchical rule* $Y \rightarrow \alpha \in \mathcal{R}$ **do**
  for each *terminal* $a \in \alpha$ **do**
    if *lexical non-terminal* L_a *does not exists* **then**
      create new lexical non-terminal L_a;
    replace a by L_a in α;
    add new rule $L_a \rightarrow a$ to $\mathcal{R}$;

of the corresponding rule is added to the type-1 lists (Algorithm 3).

Algorithm 3: Initialization Step

Data: input sentence f_1^J, set of rules $\mathcal{R}$
Result: initialized chart
for each *partial string* f_j^{j+i-1} *in* f_1^J **do**
  for each *lexical rule* $Y \rightarrow f_j^{j+i-1} \in \mathcal{R}$ **do**
    add Y to the type-1 list of the cell (i, j)

Standard Cell Filling Procedure

The real parsing is done in the standard cell filling procedure. A set of non-terminals for a given cell (i, j) is generated by the recombination of type-2 items of cell (k, j) with type-1 items of cell $(i-k, j+k)$ (Algorithm 4). A recombination is possible if the right hand side of a rule $Y \rightarrow \alpha_1 Z \alpha_2$ matches with a partial parse in the type-2 list of cell (k, j) and a parse of a substring in the type-1 list of cell $(i-k, j+k)$. Then, the non-terminal of the left hand side of the found rule parses the substring f_j^{j+i-1} and is added to the type-1 list. If α_2 is not empty, but a string of non-terminals, the type-2 item $\alpha_1 Z \bullet$ represents the partial parse of f_j^{j+i-1}.

Self-filling Procedure

The self-filling procedure is necessary to deal with rules of the form $Y \rightarrow Z$ where $Y, Z \in \mathcal{NT}$. If Z parses f_j^{j+i-1} then Y also parses this substring. Further, type-2 lists are kept up to date (Algorithm 5). This procedure also completes the ini-

Algorithm 4: Standard Cell Filling Procedure

```
Data: a cell (i, j)
Result: a filled cell (i, j)
for k ← 1 to i do
    type1-list ← (i − k, j + k).type1-list;
    type2-list ← (k, j).type2-list;
    for each α1• in type2-list do
        for each Z in type1-list do
            for each Y → α1Zα2 ∈ R do
                if α2 empty then
                    add Y to (i, j).type1-list;
                else
                    add α1Z• to (i, j).type2-list;
```

tialization step, because the type-2 lists are filled considering the initialized type-1 lists.

Algorithm 5: Self-filling Procedure

```
Data: type1-list of cell (i, j)
Result: filled type1-list and type2-list of cell (i, j)
for each Z in type1-list do
    for each Y → Zα ∈ R do
        if α empty then
            add Y to type1-list
        else
            add Z• to type2-list
```

Example

To illustrate the CYK+ algorithm, we introduce a simple monolingual grammar in Figure 3.8. A CFG is given which is not in CNF.

In Figure 3.9 we transform the CFG in a nplCFG. Therefore, we replace the terminal b in the rule $X \rightarrow Xb$ with the lexical non-terminal L_b.

Now, a input sentence *acb* is given. Figure 3.10 shows the parsing of this sentence.

$$
\begin{aligned}
S &\rightarrow XX \\
X &\rightarrow a \\
X &\rightarrow Xb \\
X &\rightarrow c
\end{aligned}
$$

Figure 3.8. Simple monolingual grammar

$$
\begin{aligned}
S &\rightarrow XX \\
X &\rightarrow a \\
X &\rightarrow Xb \\
X &\rightarrow c
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
S &\rightarrow XX \\
X &\rightarrow a \\
X &\rightarrow XL_b \\
X &\rightarrow c \\
L_b &\rightarrow b
\end{aligned}
$$

Figure 3.9. Normalization

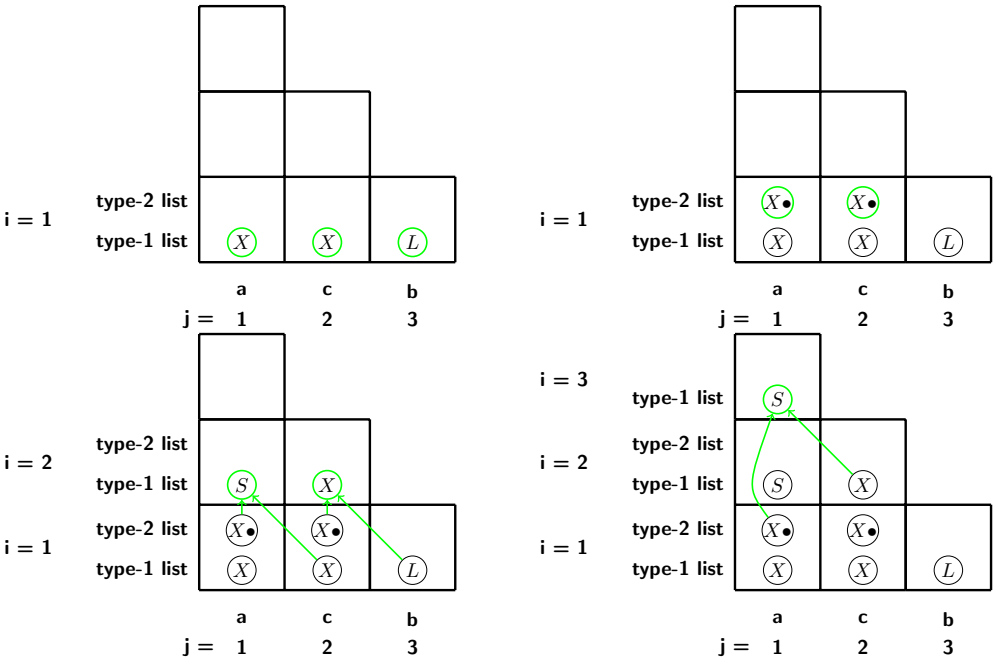

Figure 3.10. Example of a parsing with CYK+

In the initialization step, the type-1 lists of the cells $(1,1)$ and $(1,2)$ are filled with the non-terminal X, because the lexical rules $X \rightarrow a$ and $X \rightarrow c$ are in the grammar. The lexical non-terminal L_b is added to the type-1 list of the cell $(1,3)$ because the lexical rules $L_b \rightarrow b$ is in the grammar.

Then, the first self-filling procedure can be applied. With rule $S \rightarrow XX$ the type-2 lists of cell $(1,1)$ and $(1,2)$ are filled with the partial parses $X\bullet$.

In the first standard cell filling procedure, the type-2 element $X\bullet$ of cell $(1,1)$ is recombined with the type-1 element X in cell $(1,2)$, because a rule $S \rightarrow XX$ exists. Further, the non-terminal S is added to type-1 list of cell $(2,1)$. X is added to $(2,2)$, because it is also a recombination of $X\bullet$ and L_b using rule $X \rightarrow XL_b$.

The final parsing step generates the start symbol S in the cell $(3,1)$. So a parse is found for our simple example.

Considering a bilingual grammar, a probabilistic parsing and a source sentence which can be parsed with the CYK+ algorithm, the resulting parse tree induces a parse tree with the highest probability in the target language. We use this parse tree to get the best translation.

4 Extending Statistical Machine Translation Using Syntax

In this chapter, we describe the extension of our hierarchical phrase-based system using additional information extracted from deep syntactic parse trees in the decoding process.

Our standard hierarchical phrase-based system uses generic non-terminals in the grammar to generate translations. To further guide the decoding process and to get a more structured translation, we include information about the syntax of the target language. This additional information is extracted from syntax trees which are parsed from sentences of the target language. Because parsing is expensive, we opted not to parse the resulting translations in decoding process. To approximate a parsing in a decoding process, each sentence of the target language is parsed as a preprocessing step before the rule extraction. In the extraction process, we substitute the generic non-terminals of each phrase with a suitable syntactic category. The modified extraction expands the set of non-terminals. Therefore, the grammar can not generate every sentence. To get still a translation, the CKY+ algorithm must be extended to find a parse for every input sentence. This approach is similar to [Zollmann & Venugopal 06] and is described in Section 4.2. But the necessary modification of the parsing is not described in this paper. Therefore, we develop an own method.

Using this approach increases the runtime and memory complexity of the translation process compared with our standard hierarchical system. To avoid this, we introduce a second approach to include syntax information.

The idea is to use the syntax information in an additional model. The advantage is that the CYK+ algorithm does not need to be modified. Compared with our standard hierarchical system, the runtime of the decoding process increases slightly and the memory complexity is equal. This approach is described in Section 4.3 and is based on [Venugopal & Zollmann$^+$ 09]. In addition, we present two methods to use the syntax information.

In Section 4.4, we present a method to generate non-terminals without requiring a deep syntactic parser. Rather, each phrase is assigned a label based on an automatically generated cluster of phrase pairs.

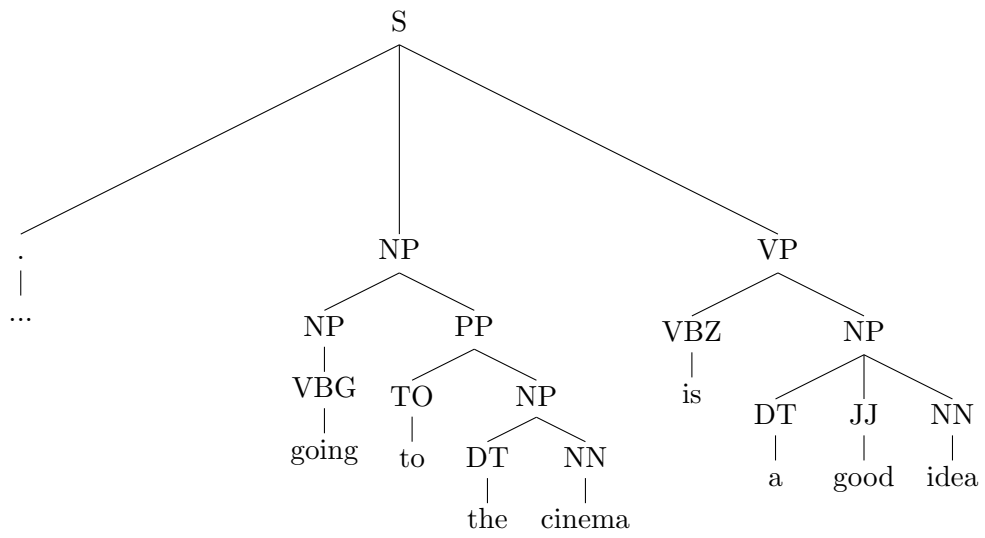

Figure 4.1. Syntax tree for sentence "[...] going to the cinema is a good idea"

4.1 Deep Syntactic Analysis

Before starting an extraction process with deep syntactic parse trees, we have to extract syntax information from the target language of the bilingual training data. In our work we use the Stanford Parser from [Klein & Manning 08] which produces *syntax trees*.

A syntax tree is a tree with an arbitrary number of children. The root node is labeled with *clause level*-labels which define the type of sentence e.g. a simple declarative clause or a direct question. The inner nodes are marked with *phrase level*-labels. In the context of deep syntactic analysis, phrases are not defined as before. On linguistic grounds, a phrase is a group of words defined as an unit in the syntax of a sentence. For example, phrases can be classified as verb phrases or noun phrases. The leaves of the syntax tree are labeled with the words of the sentence and the corresponding *word level*-labels. This labels are the well-known part-of-speech tags. Table B.1 on page 52 gives an overview about the different labels used in this work. For more details about the meaning of the labels, we refer to [Santorini 90] and [Ann Bies & MacIntyre 95].

As preprocessing step before the rule extraction, each sentence of the target language must be parsed to get such a syntax tree. Figure 4.1 shows an example of a syntax tree for a given target sentence (Complete syntax tree in Figure B.1).

$$r_8 : NP \rightarrow \langle \text{Ins Kino zu gehen}, \text{going to the cinema} \rangle$$
$$r_{44} : PP \rightarrow \langle \text{meiner Meinung nach}, \text{In my opinion} \rangle$$
$$r_{59} : NP \rightarrow \langle \text{eine gute Idee}, \text{a good idea} \rangle$$

Figure 4.2. Lexical rules extended with syntax

4.2 First Approach: Extending the Non-Terminal Set

In the first approach, the set of non-terminals $\mathcal{NT}$ is extended by the labels of the syntax trees. In order to do that, the extraction of our hierarchical system must be modified.

4.2.1 Rule Extraction

In the extraction process, we have to extend the extraction of the lexical rules. Each produced phrase from a sentence is labeled with syntactic categories from the corresponding syntax tree. This method is introduced in [Vilar & Stein$^+$ 08]. In order to label a phrase $(\tilde{f}, \tilde{e})$ of a input sentence (f_1^J, e_1^I), we check whether this sequence corresponds to the yield of some node in the syntax tree. If we do not find such a node, we compute the minimum number of words which must be deleted or added to the phrase to find a suitable node. To determine this number, we search in the tree for the highest node that covers all the words of the phrase and possibly more. If the phrase is transformed to a lexical rule, the generic non-terminal on the right hand side is substituted with the found label. In Figure 4.2 we see the example of the previous chapter extended with labels of the syntax tree in Figure 4.1.

The extraction of the hierarchical rules uses the non-terminals on the left hand side of the lexical rules to replace the sub-phrases. Figure 4.3 shows an example of hierarchical rules extended with syntactic categories.

The probabilities of the rules are reestimated for the new grammar. Because the set of non-terminals is extended, the same rule with different non-terminals is stored in the new grammar. For example, the rule

$$X \rightarrow \langle \text{Ins Kino } X^{\sim 1} \text{ ist } X^{\sim 2}, X^{\sim 2}, X^{\sim 1} \text{ to the cinema is} \rangle$$

which is seen two times in the standard grammar is augmented to two rules:

$$r_{18} : S \rightarrow \langle \text{Ins Kino } NP^{\sim 1} \text{ ist } PP^{\sim 2}, PP^{\sim 2} NP^{\sim 1} \text{ to the cinema is} \rangle$$
$$r_{46} : PP \rightarrow \langle \text{meiner Meinung } IN^{\sim 1}, IN^{\sim 1} \text{ my opinion} \rangle$$
$$r_{114} : S \rightarrow \langle NP^{\sim 1} \text{ ist } NP^{\sim 2}, NP^{\sim 2} NP^{\sim 1} \text{ is} \rangle$$

Figure 4.3. Hierarchical rules extended with syntax

$$S \rightarrow \langle \text{Ins Kino } VBG^{\sim 1} \text{ ist } PP^{\sim 2}, PP^{\sim 2}, VBG^{\sim 1} \text{ to the cinema is} \rangle$$
$$S \rightarrow \langle \text{Ins Kino } VBG^{\sim 1} \text{ ist } NP^{\sim 2}, NP^{\sim 2}, VBG^{\sim 1} \text{ to the cinema is} \rangle$$

4.2.2 Parsing with CYK*

The extraction with syntax information defines a new grammar with an extended set of non-terminals. Not all input sentences can be parsed using CYK+ and for these we get no translation. This can happen if the non-terminals of the right hand side of a rule can not be rewritten by other rules. Therefore, we modify CYK+ to allow substitution of non-matching non-terminals. Nevertheless, a correct parse should be preferred. In order to do that, the substitution of non-matching non-terminals is penalized. This additional costs are added by a binary feature h_{syntax}.

The modification allows the algorithm to be more flexible if a parsing is not possible. The conversion algorithm and the initizalitation step are the same as in CYK+ algorithm. We have to adapt the self-filling procedure as well as the standard cell filling procedure.

New Standard Cell Filling Procedure

In the new standard cell filling procedure, we allow to recombine the non-terminals in the type-2 list of cell (k, j) with every non-terminal in $\mathcal{NT}$ instead of considering the non-terminals in the type-1 list of cell $(i-k, j+k)$ only. A recombination causes an additional cost if the non-terminal is not in the type-1 list (Algorithm 6).

New Self-filling Procedure

In the self-filling procedure, every non-terminal in $\mathcal{NT}$ is considered instead of testing only the non-terminals in the type1-list of the cell. A non-terminal is added

Algorithm 6: New Standard Cell Filling Procedure

```
Data: a cell (i, j)
Result: a filled cell (i, j)
for k ← 1 to i do
    type1-list ← (i − k, j + k).type1-list;
    type2-list ← (k, j).type2-list;
    for each α1• in type2-list do
        for each Z in NT do
            for each Y → α1 Z α2 ∈ R do
                if Z ∈ type1-list then
                    if α2 empty then
                        add Y to (i, j).type1-list;
                    else
                        add α1 Z• to (i, j).type2-list;
                else
                    if α2 empty then
                        add Y to (i, j).type1-list with additional costs;
                    else
                        add α1 Z• to (i, j).type2-list with additional costs;
```

$$
\begin{array}{ll}
S \rightarrow XX & S \rightarrow BA \\
X \rightarrow a & A \rightarrow a \\
X \rightarrow XL_b & \Rightarrow \quad B \rightarrow BL_b \\
X \rightarrow c & C \rightarrow c \\
L_b \rightarrow b & L_b \rightarrow b
\end{array}
$$

Figure 4.4. Extended monolingual grammar

with additional costs if it is not in the origin type-1 list of a cell (Algorithm 7).

Example

To illustrate the CYK* algorithm, we extend our simple monolingual grammar of Figure 4.4 with further non-terminals $\{A, B, C\}$. The input sentence *acb* can not parsed with the new grammar using CYK+.

Algorithm 7: New Self-filling Procedure

```
Data: type1-list of cell (i, j)
Result: filled type1-list and type2-list of cell (i, j)
for each Z in NT do
    for each Y → Zα ∈ R do
        if Z ∈ type1-list then
            if α empty then
                add Y to type1-list;
            else
                add Z• to type2-list;
        else
            if α empty then
                add Y to type1-list with additional costs;
            else
                add Z• to type2-list with additional costs;
```

Figure 4.5 shows a parsing of the sentence *acb* with CYK*. Items in the chart cells are marked with orange if they are added with costs. Further, only these items which are needed for the parse tree are shown.

In the first new self-filling step, $B\bullet$ is added to the type-2 lists of cell $(1, 1)$ and $(1, 2)$. This is allowed because the lexical rule $S \rightarrow BA$ is in the grammar. Furthermore, this operation causes cost, but $B\bullet$ is needed for the recombination. The type-2 item $B\bullet$ in cell $(1, 1)$ can be recombined with the type-1 item C in cell $(1, 2)$ because C is substituted by $A \in \mathcal{NT}$. After a further new self-filling step and a further new standard cell filling step, a parse is found.

4.2.3 Problems Using this Approach

We tested this approach on a small corpus and observed a high runtime and memory complexity. This is due to several reasons. Using further non-terminals, the grammar is augmented and more rules must considered while parsing. The CYK* algorithm has to check more conditions and possible recombinations. Further, the produced parse tree grows and needs more memory. If we add the language model to the decoding process, the Cube Prune algorithm has to search for the best derivation in a greater forest of parse trees. Runtime and memory usage of sentences with different length from the IWSLT corpus (Section 5.2) are shown in Table 4.1 and 4.2.

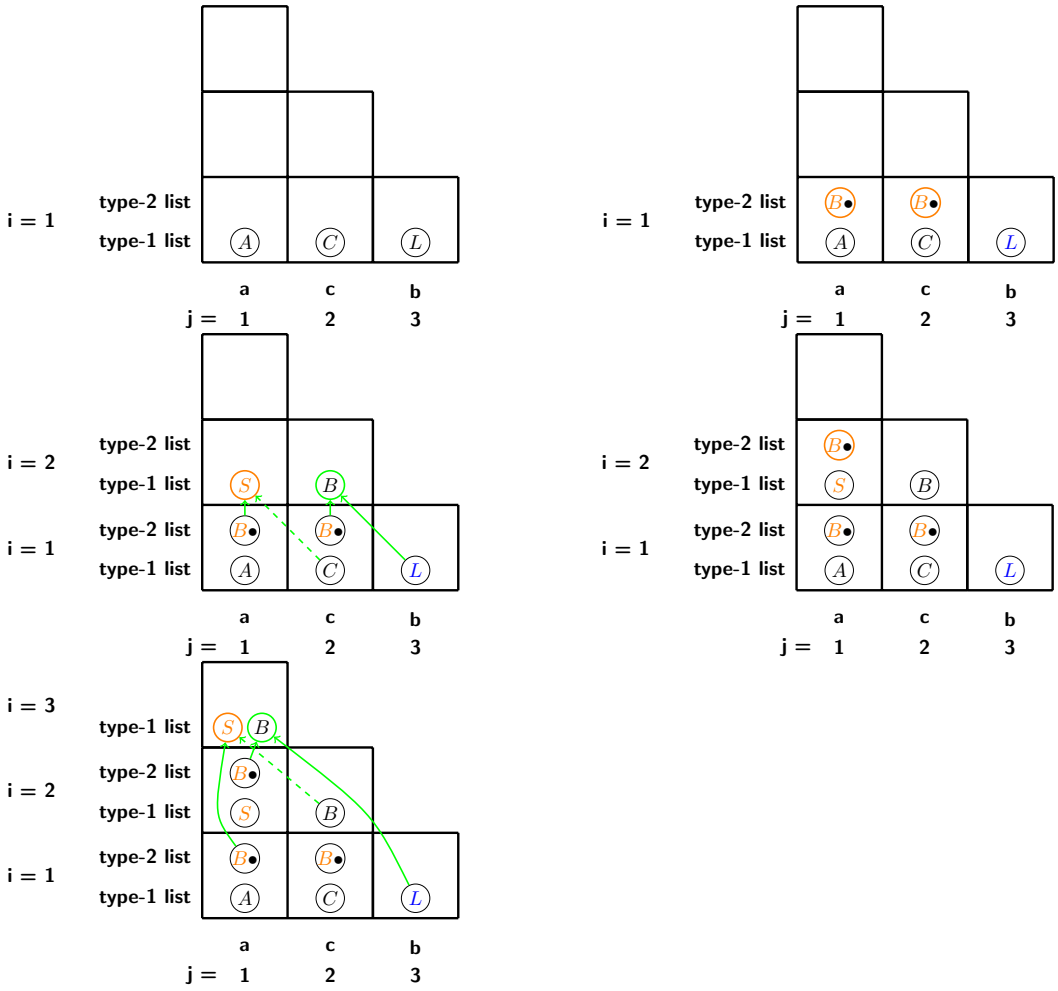

Figure 4.5. Example of a parsing with CYK*

	runtime [s]		
sentence length	baseline	syntax	
	parsing + Cube Prune	parsing	Cube Prune
8	0.4	1.5	8.5
13	1	6.5	41.5

Table 4.1. Runtime

	memory usage [mb]		
sentence length	baseline	syntax	
	parsing + Cube Prune	parsing	Cube Prune
8	22	96	347
13	44	390	2 150

Table 4.2. Memory usage

Nevertheless, we present an experiment on IWSLT with this approach in Chapter 5.

4.3 Second Approach: Soft Syntactic Labels

The second approach to extend the hierarchical phrase-based translation system uses an additional model. The parsing is not restricted by an augmented set of non-terminals. We use the same grammar with the set of generic non-terminals $\mathcal{NT} = \{S, X\}$ as in our standard hierarchical phrase-based system. Therefore, we can work with the CYK+ algorithm again. The syntax information is stored at each rule additionally. For the computation of an additional feature, the generic non-terminals are replaced with labels from the syntax trees. This set of labels is denoted as $\mathcal{H} = \{NP, PP, NN, DT \ldots\}$. Before introducing two additional features which use the syntax information different, we describe the extraction of *Soft Syntactic Labels*.

4.3.1 Rule Extraction

The extraction is similar to the method described in Section 4.2.1.

Instead of adding the new non-terminals from the syntax tree to $\mathcal{NT}$, we add them to the set of labels $\mathcal{H}$. For each rule r_i, we define a probability distribution $p(\mathbf{h}|r_i)$ over a vector of labels $\mathbf{h} = (h_0, h_1, \ldots, h_{|\mathbf{h}|})$ which replaces in the additional model the generic non-terminal on the left hand side (h_0) as well as each generic non-terminal on the right hand side ($h_1, \ldots, h_{|\mathbf{h}|}$) of the rule. Each h_i is a label in $\mathcal{H}$ and the set of all vectors of a rule is denoted as $H(r_i)$. To denote the j-th entry in a vector $\mathbf{h}$, we write $\mathbf{h}[j]$.

Figures 4.6 and 4.7 show the difference between the extraction of the first approach and the second approach. As we can see, the second approach uses the rule $X \rightarrow \langle \text{Ins Kino } X^{\sim 1} \text{ ist } X^{\sim 2}, X^{\sim 2}, X^{\sim 1} \text{ to the cinema is} \rangle$ from our standard hierarchical system. Further, the rule is extended with a distribution over (S, PP, VBG) and (S, NP, VBG).

$$r_{24}: \quad S \rightarrow \langle \text{Ins Kino } VBG^{\sim 1} \text{ ist } PP^{\sim 2}, PP^{\sim 2}, VBG^{\sim 1} \text{ to the cinema is} \rangle$$
$$r_{25}: \quad S \rightarrow \langle \text{Ins Kino } VBG^{\sim 1} \text{ ist } NP^{\sim 2}, NP^{\sim 2}, VBG^{\sim 1} \text{ to the cinema is} \rangle$$

Figure 4.6. Hierarchical rules using extended set of non-terminals

$$r_{18}: \; X \rightarrow \langle \text{Ins Kino } X^{\sim 1} \text{ ist } X^{\sim 2}, X^{\sim 2}, X^{\sim 1} \text{ to the cinema is} \rangle$$
$$\left\{ \begin{array}{rcl} p((S, PP, VBG)|r_{18}) & = & 0.5 \\ p((S, NP, VBG)|r_{18}) & = & 0.5 \end{array} \right\}$$

Figure 4.7. Hierarchical rule with probability distribution

Because the second approach uses the same grammar as the standard hierarchical system, the probabilities of the rules do not equal the probabilities of rules in the grammar which is built up with the extraction of the first approach. The probability that a rule was seen with certain labels is stored in the additional information.

We described in the introduction of this section, the CYK+ algorithm remains unaffected, but to guide the translation process we introduce two additional features.

4.3.2 Simple Penalization

With the feature $h_{simpleP}$, we try to approximate the penalization of the first approach. We define a rule r and a set of rules $\{r_i\}_1^{|\mathbf{h}|}$ applied in a derivation to be consistent if

$$\exists \mathbf{h} \in H(r) : \forall i \in \{1, \ldots, |h|\} \; \exists \mathbf{h}' \in H(r_i) \; \mathbf{h}'[0] = h_i.$$

$h_{simpleP}$ equals one if a rule and set of rules are not consistent.

From the subset of a grammar in Figure 4.8, two different translation for the source sentence "diese Zweideutigkeit ..." can be derived.

The first translation is derived using the rules r_0 and r_1 and translate the source sentence with "this ambiguity ...". The used rules are consistent, because the label DT on the right hand side of rule r_0 matches the label of the left hand side of r_1.

However, the second translation "such ambiguity ..." is derived with the rules r_0 and r_2, which are not consistent. Not any label on the right hand side of the rule

$$
\begin{array}{lll}
r_0: & X & \rightarrow \langle X \text{ Zweideutigkeit}, \ X \text{ ambiguity} \rangle \\
 & & \left\{ \begin{array}{c} (NP, DT) \\ (PP, PP) \\ (NP, NP) \end{array} \right\} \\
r_1: & X & \rightarrow \langle \text{diese}, \ \text{this} \rangle \\
 & & \{ \ (DT) \ \} \\
r_2: & X & \rightarrow \langle \text{diese}, \ \text{such} \rangle \\
 & & \left\{ \begin{array}{c} (JJ) \\ (PDT) \end{array} \right\}
\end{array}
$$

Figure 4.8. Example simple penalization

r_0 matches a label on the left hand side of r_2. Therefore, the second translation get additional cost.

4.3.3 Tree Well-formedness Probability Model

The hard matching constraints of the simple penalization can be relaxed by using the probability distribution of each rule. We formulate a feature which measures the compatibility between two rules. The idea is that rules with high mutual match should get a high probability.

For simplification of explanation, the following definitions only hold for one non-terminal on the right hand side of each rule and we assume that in derivation d rules are applied. Later, we give a more complex definition for several non-terminals.

In a bottom-up manner, we computed for a rule r_i applied in the derivation d a probability distribution $p(h, r_i)$ over all labels h in $\mathcal{H}$ and a feature $h_{syntaxPM}(r_i)$. Both are calculated using the distribution which was computed for partial derivations $r_{i+1}^{|d|}$ and the probability distribution $p(\mathbf{h}, r_i)$ of the rule r_i in the grammar. Figure 4.9 illustrates this method. A distribution $p(\cdot, d_1)$ over the non-terminals B, Y, D for the partial derivation d_1 was computed. With this distribution and the probability distribution of the rule $X \rightarrow aXb$, the distribution over the labels A and C and the feature for the derivation d is estimated.

The feature in our log-linear model is the multiplication of $h_{syntaxPM}(r_i|r_{i+1}^{|d|})$:

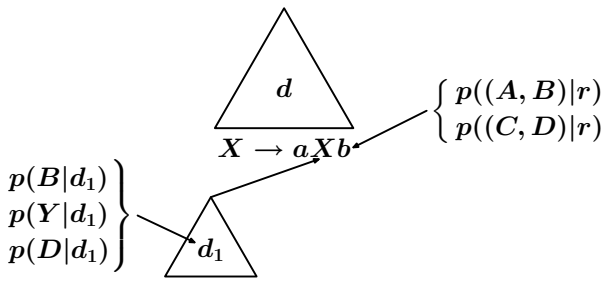

Figure 4.9. Illustration of the Tree Well-formedness Probability Model

$$h_{syntaxPM}(d) = \prod_{i=1}^{|d|} h_{syntaxPM}(r_i|r_{i+1}^{|d|}) \tag{4.1}$$

For a lexical rule $r_{|d|}$ applied in the derivation, the distribution for the computation of the feature equals to distribution of the rule in the grammar. Further, we define the feature $h_{syntaxPM}(r_{|d|})$ to be one, because no previous derivation is given.

$$\forall h_0 \in \mathcal{H} : p(h_0|r_{|d|}) = p((h_0)|r_i) \tag{4.2}$$

$$h_{syntaxPM}(r_{|d|}) = 1 \tag{4.3}$$

Given a hierarchical rule r_i applied in the derivation, the distribution $p(\cdot, r_i)$ is calculated by using the probability distribution of the rule $p(\mathbf{h}|r_i)$ and the distribution $p(\cdot|r_{i+1}^{|d|})$. We note that vectors (h_0, h_1) are added where the first entry equals the given label h_0.

$$\tilde{p}(h_0|r_i) = \sum_{(h_0,h_1)\in H(r_i)} p((h_0, h_1)|r_i) \cdot p(h_1|r_{i+1}^{|d|}) \tag{4.4}$$

$$\forall h_0 \in \mathcal{H} : p(h_0|r_i) = \frac{\tilde{p}(h_0|r_i)}{\sum_{h_0' \in \mathcal{H}} \tilde{p}(h_0'|r_i)} \tag{4.5}$$

Renormalizing keeps the estimated probabilities in the same scale as the probabilities of the rules. The feature is computed similar to the distribution, but it is added for all vectors $\mathbf{h}$ in $H(r_i)$:

$$
\begin{array}{lll}
r_0: & X & \rightarrow \langle X \text{ Zweideutigkeit},\ X \text{ ambiguity} \rangle \\
 & & \left\{ \begin{array}{rcl} p((NP,DT)|r_0) & = & 0.5 \\ p((PP,PP)|r_0) & = & 0.3 \\ p((NP,NP)|r_0) & = & 0.2 \end{array} \right\} \\
r_1: & X & \rightarrow \langle \text{diese},\ \text{this} \rangle \\
 & & \left\{ \begin{array}{rcl} p((DT)|r_1) & = & 1 \end{array} \right\} \\
r_2: & X & \rightarrow \langle \text{diese},\ \text{such} \rangle \\
 & & \left\{ \begin{array}{rcl} p((JJ)|r_2) & = & 0.7 \\ p((PDT)|r_2) & = & 0.3 \end{array} \right\}
\end{array}
$$

Figure 4.10. Grammar with probability distribution at each rule

	diese Zweideutigkeit ...	$h_{syntaxPM}$
d_1	this ambiguity ...	$0.5 \cdot 1 + 0.3 \cdot 0 + 0.2 \cdot 0 = 0.5$
d_2	such ambiguity ...	$0.5 \cdot 0 + 0.3 \cdot 0 + 0.2 \cdot 0 = 0$

Table 4.3. Example for computing the additional feature using the Tree Well-formedness Probability Model

$$
h_{syntaxPM}(r_i|r_{i+1}^{|d|}) = \sum_{\mathbf{h} \in H(r_i)} p(\mathbf{h}|r_i) \cdot p(h_1|r_{i+1}^{|d|}) \tag{4.6}
$$

In Figure 4.10 we extend our example of Figure 4.8 with probabilities, so the feature for each derivation can be calculated (see Table 4.3).

To considering more than one non-terminal on right hand side of a rule, we introduce three auxiliary functions:

- $r(i,d)$ returns the applied rule at the position i in a derivation d
- $a(i,d)$ returns for a given position i in the derivation d a tuple of positions in the derivation where the non-terminals of the right hand side of $r(i,d)$
- $s(t,d)$ returns analogous to $a(i,d)$ a tuple of partial derivations for a tuple t of positions in the derivation d

We denote $s(t,d)[j]$ for the jth derivation of a tupel of partial derivation.

Further, we assume that a derivation d use N rules of the grammar.

Now, we can rewrite the definition of $h_{syntaxPM}(d)$ for several non-terminals:

$$h_{syntaxPM}(d) = \prod_{i=1}^{|N|} h_{syntaxPM}(i|s(a(i,d),d) \quad (4.7)$$

If $r(i,d)$ is a lexical rule, we define:

$$\forall h_0 \in \mathcal{H} : p(h_0|s(i,d)) = p((h_0)|r(i,d)) \quad (4.8)$$
$$h_{syntaxPM}(i) = 1 \quad (4.9)$$

If $r(i,d)$ is a hierarchical rule, we define:

$$\tilde{p}(h_0|s(i,d)) = \sum_{\mathbf{h}=(h_0,h_1,\ldots,h_{|\mathbf{h}|})\in H(r(i,d))} p(\mathbf{h}|r(i,d)) \prod_{j=1}^{|\mathbf{h}|} p(h_j|s(a(i,d),d)[j]) \quad (4.10)$$

$$\forall h_0 \in \mathcal{H} : p(h_0|s(i,d)) = \frac{\tilde{p}(h_0|s(i,d))}{\sum_{h_0'\in\mathcal{H}} \tilde{p}(h_0'|s(i,d))} \quad (4.11)$$

$$h_{syntaxPM}(i|s(a(i,d),d))) = \sum_{\mathbf{h}\in H(r(i,d))} p(\mathbf{h}|r(i,d)) \prod_{j=1}^{|\mathbf{h}|} p(h_j|s(a(i,d),d)[j]) \quad (4.12)$$

In pratice, we combine the Simple Penalization and the Tree Well-Formedness Probability Model. In our hierarchical phrase-based translation system, we use the logarithm of probabilities. If we computed with the Tree Well-Formedness Probability Model a probability equals zero, the logarithm will be infinity. Therefore, the feature $h_{simpleP}$ equals one whenever $h_{syntaxPM} = 0$.

4.4 Additional Approach: Poor Man's Syntax

In this section, we present a new approach to generate non-terminals extending our hierarchical system. Until now, we had to use additional data to generate non-terminals and could not vary the number of non-terminals. Further, the Stanford

Parser is trained on the Wall-Street-Journal and does not take the given training corpus into account.

The idea is to generate non-terminals representing groups of similar phrases. Theses groups are found automatically by clustering all possible non-hierarchical phrases, which are extracted from the training data. Then, we label the phrases with the assigned cluster.

To do that efficiently a hierarchical clustering is used. First, we cluster each word of the training corpus with *mkcls* [Och 00] to a wordgroup. Then, the words in the phrases are substituted with the corresponding classes. The resulting phrases are clustered by *cluto* [Karypis 07] and the original phrases are labelled with the resulting clusters. In Section 5 we present our first result for different number of clusters using the Tree Well-formedness Probability Model.

5 Experiments

In this chapter we present the results of several experiments using the proposed methods. First, we introduce two evaluation measures to evaluate the quality of translations. After that, we describe the used corpora. The experiments were done on different language pairs and corpora sizes. Then, we analyze the effect of including information of deep syntactic parse trees in the decoding process. The results are presented in Section 5.4.

5.1 Evaluation Measures

In machine translation, the automatic evaluation of the generated translations is an important challenge. Using human evaluation is extensive, but it is also time-consuming. To evaluate a huge amount of translations produced by translation systems, we need a fast, automatic and language independent evaluation method. This method should also correlate highly with human evaluation. To evaluate the quality of a translation, the translation of each source sentence is compared with a human generated reference translation. The following two metrics are state-of-the-art in machine translation.

5.1.1 Translation Edit Rate

The *Translation Edit Rate* (TER) is an error metric and is defined as the minimum number of edit operations needed to change a translation so that it equals to the closest the reference translation [Snover & Dorr$^+$ 06]. Similar to the word error rate, the edit operations include insertion, deletion and substitution of single words. In addition, a shift moves a contiguous sequence of words within the translation. Each edit operation has equal cost. This measure is normalized by the average length of the references. More formally:

$$\text{TER} = \frac{\text{number of edit operations}}{\text{average number of reference words}} \tag{5.1}$$

In practice, the TER is computed on document level by collecting the edit counts over the whole data set.

5.1.2 Bilingual Evaluation Understudy

The *Bilingual Evaluation Understudy* (BLEU) is an evaluation metric introduced by [Papineni & Roukos$^+$ 02]. BLEU is a precision measure and uses a *modified n-gram precision* combined with a *brevity penalty*. To compute the modified n-gram precision $p_n(e_1^I, e'^{I'}_1)$ for a given translation e_1^I and reference translation $e'^{I'}_1$, we count the number of occurrences of a n-gram w_1^n in the translation present in the reference translation. The minimum of both counts is divided by the total number of n-grams in the translation. The counts are denoted by $C(w_1^n|e_1^I)$. If no n-gram is found in the translation, the n-gram precision equals zero. Further, we add p_n for 1 to N n-grams and divide the sum by N to get a geometric mean.

The brevity penalty BP is necessary to penalize too short translations.

$$\text{BLEU} = BP(I, \hat{I}) \cdot \exp\left(\frac{1}{N}\sum_{n=1}^{N} p_n(e_1^I, e'^{I'}_1)\right) \tag{5.2}$$

with

$$BP(I, \hat{I}) = \begin{cases} 1 & \text{if } I \geq I' \\ \exp(1 - I'/I) & \text{if } I < I' \end{cases} \tag{5.3}$$

$$p_n(e_1^I, e'^{I'}_1) = \frac{\sum_{w_1^n} \min\left\{C(w_1^n|e_1^I), C(w_1^n|e'^{I'}_1)\right\}}{\sum_{w_1^n} C(w_1^n|e_1^I)} \tag{5.4}$$

BLEU is defined to take more than one reference translation into account. We compute BLEU on document level by collecting the n-gram counts over all translations.

5.2 Data Sets

We tested our methods on three different corpora with different sizes and language pairs.

The first corpora is from the *International Workshop on Spoken Language Translation* (IWSLT) and consists of sentence from the travel domain. The training data

		Chinese	English
train:	Sentences	42 942	
	Running Words	380 259	420 431
	Vocabulary	11 760	9 933
	Singletons	4637	3937
dev:	Sentences	500	
	Running Words	3 578	62 520
	Vocabulary	950	3 878
	OOVs	75	28 177
test:	Sentences	506	
	Running Words	3 837	63 525
	Vocabulary	938	4 099
	OOVs	73	28 913
eval:	Sentences	489	
	Running Words	3 256	25 019
	Vocabulary	885	1 528
	OOVs	60	6 822

Table 5.1. Corpus statistics for IWSLT Chinese-English

set (train) of the language pair Chinese-English was taken from the IWSLT 2007 evaluation and is a small corpus with roughly 43K training sentences. We took the development set (dev) from the IWSLT 2004 evaluation and the test set from IWSLT 2006 evaluation (test) and an additional evaluation set from IWSLT 2007 evaluation (eval) (statistics shown in Table 5.1).

A second corpus is the QUAERO corpus. The data are mainly taken from Europarl parallel corpus which is extracted from the proceedings of the European Parliament. It includes versions in 11 European languages, but we choose the language pair German-English . With about 1.5 million training sentences (train), QUAERO is a medium size corpus. We use the development (dev) and test set (test) published for the *ACL 2009 Workshop on Statistical Machine Translation* (WMT09) (statistics shown in Table 5.2).

A medium size subset of the corpus from the *National Institute of Standards and Technology* (NIST) was chosen to test our methods on an well-known corpus with the language pair Chinese-English. The data consists of different news text and was collected by the *Linguistic Data Consortium*. The development set (dev) was taken from the NIST 2006 evaluation and the test set (test) is from the NIST 2008 evaluation (statistics shown Table 5.3).

		German	English
train:	Sentences	1 521 715	
	Running Words	41 009 835	41 695 098
	Vocabulary	177 031	119 140
	Singletons	66 985	45 575
dev:	Sentences	2 121	
	Running Words	56 029	45 211
	Vocabulary	9 454	10 325
	OOVs	1 121	6 131
test:	Sentences	2 007	
	Running Words	53 654	43 797
	Vocabulary	9 375	9 999
	OOVs	1 341	5 881

Table 5.2. Corpus statistics for QUAERO German-English

		Chinese	English
train:	Sentences	1 165 478	
	Running Words	30 545 919	31 351 263
	Vocabulary	69 804	180 921
	Singletons	15 782	82 502
dev:	Sentences	1 664	
	Running Words	42 930	194 885
	Vocabulary	6 387	9 673
	OOVs	1 897	6 935
test:	Sentences	1 357	
	Running Words	36 114	149 057
	Vocabulary	6 418	17 877
	OOVs	1 449	43 595

Table 5.3. Corpus statistics for NIST Chinese-English

5.3 Experimental Setup

The scaling factors of our log-linear model are computed using MERT on the development sets (dev) of the corpora. We test our system blind on the test sets (test). For IWSLT we got an additional test set (eval).

In the results, we denote the first approach with *syntax*, the second approach us-

			test		eval	
	rules	NTs	BLEU [%]	TER [%]	BLEU [%]	TER [%]
baseline	791 799	2	55.4	33.6	31.3	50.5
syntax	1 625 149	68	57.6	32.8	35.0	48.3

Table 5.4. Results on IWSLT Chinese-English using first approach

source	你 认识 她 多久 了 ?
baseline	do you know how long is she ?
syntax	how long do you know her ?
reference (1 of 16)	how long have you known her ?

Table 5.5. Example for hierarchical phrase-based translation

ing the Simple Penalization with *syntax simple penalty*, the second approach using the Tree Well-formedness Probability Model *syntax probability model*, and the second approach using Poor Man's Syntax with *syntax poor man* and the number of clusters.

For each experiment we present examples from the translation to investigate the impact of the additional syntax information.

5.4 Results

5.4.1 First approach: Extending the Non-Terminal Set

Table 5.4 shows the results of the experiment on IWSLT. We get an improvement of 2% in BLEU on test and 4% in BLEU on eval. Also, the TER decreases. In this experiment the grammar is extended with 68 non-terminals. We also can observe that the number of rules of the resulting grammar is two times higher than the number of rules of the baseline.

In table 5.5, we see an example of a better translation with syntax. The difference between the translation of the single words is minor, but the different reordering gives the translation sense. Further, "her" is used instead of "she". It seems that the system is able to learn dependencies between phrases and the correct grammatical sub-phrase. In this case, after the phrase "how long do you", a verb phrase follows which consists of the verb "know" and of the personal pronoun "her".

source	能 给 我 一 张 桌子 吗 ?
baseline	can i have a table ?
syntax	could you give me a table ?
reference (1 of 16)	can i have a table ?

Table 5.6. Example for hierarchical phrase-based translation

		test		eval	
	NTs	BLEU [%]	TER [%]	BLEU [%]	TER [%]
syntax	68	57.6	32.8	35.0	48.3
baseline	2	55.4	33.6	31.3	50.5
+ syntax simple penalty	2/68	56.6	33.1	31.9	50.5
+ syntax probability model		56.7	33.1	33.3	49.3

Table 5.7. Results on IWSLT Chinese-English using second approach

The translation in Table 5.6 is an example for producing alternative translation. The sentence "could you give me a table ?" still makes sense, but none of the 16 reference translations equals such a translation.

Considering this experiment, we can reason that the first approach can improve the translation quality. Nevertheless, the runtime and memory complexity are high and further experiments on other corpora are not possible.

5.4.2 Second Approach: Soft Syntactic Labels

With the second approach, we had the possibility to test on larger corpus like QUAERO and NIST. The memory complexity is the same as running the translation process without the additional models. The runtime is slightly higher, because additional computations must done.

IWSLT experiments

However, we first test the second approach on IWSLT to compare with the first approach. The results in table 5.7 shows that the translation using Simple Penalization or the Tree Well-formedness Probability Model is superior than the baseline in this experiment. But the first approach still reaches the best results.

systems	# of different translations
baseline, syntax	314
baseline, syntax simple penalty	194
baseline, syntax probability model	253
syntax, syntax simple penalty	303
syntax, syntax probability model	316
syntax simple penalty, syntax probability model	243

Table 5.8. Number of the different translations on IWSLT eval set

source	我 计划 明天 早上 离开 .
baseline	i am planning to tomorrow morning .
syntax	i plan to leave tomorrow morning .
syntax simple penalty	i am planning to leave tomorrow morning .
syntax probability model	i am leaving tomorrow morning .
reference 1	i plan to leave tomorrow in the morning .
reference 2	i'm planning to leave tomorrow morning .

Table 5.9. Translation example from IWSLT test set

We count the number of different translation in Table 5.8 and observe that the each method induces different translations.

In Table 5.9 three different translations for the same source sentence are listed. If syntax information is involved in the translation, the verb "leave" is used. The non-terminals of the rule $X \rightarrow \langle X^{\sim 1}$ 明早 $X^{\sim 2}$, i am $X^{\sim 1}$ to $X^{\sim 2}$ tomorrow morning$\rangle$ is substituted with (VBG, VP). Therefore, the decoder prefers to replace $X^{\sim 1}$ with a gerund and $X^{\sim 2}$ with a verb phrase.

QUAERO experiments

The next experiments were made on the QUAERO corpus. It seems that the extension does not have any effect. The results in Table 5.10 shows no change in Bleu as well as in Ter.

Nevertheless, we compare the translations in detail. In Table 5.11 both translations with syntax information have a higher Ter than the baseline translation, although the baseline translation does not include the translation of the verb "konzentriert sich". It seems the additional models learned that a verb is needed and therefore

		dev		test	
	NTs	BLEU [%]	TER [%]	BLEU [%]	TER [%]
baseline	2	24.4	59.0	26.1	56.4
+ syntax simple penalty	2/70	24.5	59.1	26.2	56.4
+ syntax probability model		24.8	58.6	26.1	56.4

Table 5.10. Results on QUAERO German-English using second approach

		TER [%]
source	während sich die Welt auf den Irak , Nordkorea und eine mögliche Auseinandersetzung mit dem Iran über Atomwaffen konzentriert , ist der Kosovo von der Bild Fläche verschwunden .	-
baseline	while the world on Iraq , North Korea , and a possible conflict with Iran on nuclear weapons , is the Kosovo disappeared .	35.7
syntax simple penalty	while the world on Iraq , North Korea , and a possible conflict with Iran on nuclear weapons concentrated , is the Kosovo disappeared .	39.3
syntax probability model	while the world on Iraq , North Korea and a possible conflict with Iran on nuclear weapons concentrated , is the Kosovo disappeared .	42.9
reference	with the world focused on Iraq , North Korea , and a possible clash with Iran over nuclear weapons , Kosovo has fallen off the radar screen .	-

Table 5.11. Translation example from QUAERO development set

"konzentriert sich" had to be translated with "concentrated".

Table 5.12 shows a further translation example. Using the additional model, "as a Turk" is identified as prepositional phrase and therefore a comma must follow. A suitable translation for "empfinde" is found.

source	als Türke empfinde ich beide Seiten dieser Debatte unmittelbar .
baseline	as Turk I find both sides in this debate .
syntax simple penalty	as a Turk , I feel both sides in this debate immediately .
syntax probability model	as a Turk , I feel both sides in this debate immediately .
reference	as a Turk , I feel both sides of this debate directly .

Table 5.12. Translation example from QUAERO test set

systems	# of different translation
baseline, syntax simple penalty	1477
baseline, syntax probability model	1737
syntax simple penalty, syntax probability model	1565

Table 5.13. Number of different translations on QUAERO test set

Because there are better translations as well as worse translations, all in all the results of this experiment do not show improvements. We observe that more than half of the translations are different (Table 5.13).

NIST experiments

With testing on NIST, we consider again the language pair Chinese-English. The results of this experiment are shown in Table 5.14. It seems using the Simple Penalization reduces the translation quality. With the Tree Well-formed Probability Model, we get slight improvement in Bleu. Ter is unchanged.

In Table 5.15 three different translations are listed. Both translations made by the extend hierarchical system consider that "the united nations" is 3rd person singular with the label *VBZ*. Therefore, "has" instead of "have" was used.

Further, we can also observe that syntax information helps to learn dependencies between a modal auxiliary verb and the following main verb. In Table 5.16, both method using syntax information consider the correct order "need to resolve through dialogue".

		dev		test	
	NTs	BLEU [%]	TER [%]	BLEU [%]	TER [%]
baseline	2	27.6	66.2	22.2	69.3
+ syntax simple penalty	2/68	26.9	66.5	22.1	70.1
+ syntax probability model	2/68	28.4	65.6	22.6	69.2

Table 5.14. Results on NIST Chinese-English using second approach

source	联合国 因 德黑兰 未能 冻结 铀 浓缩 作业 , 已 对 伊朗 实施 two 套 制裁 .
baseline	the united nations because tehran failed to freeze uranium enrichment operations , have sanctions against iran carry out two sets .
syntax simple penalty	the united nations because tehran will not freeze uranium enrichment operations , has implemented two sets of sanctions against iran .
syntax probability model	the united nations because tehran failed to freeze uranium enrichment operations , has implemented two sets of sanctions against iran .
reference (1 of 4)	the un has already imposed two sets of sanctions on iran because tehran failed to freeze its uranium enrichment operation .

Table 5.15. Translation example from NIST test set

In Table 5.17, we check the number of different translations and observe again that each method produces different translations.

5.4.3 Additional Approach: Poor Man's Syntax

We tested our new approach to generate non-terminals on NIST with the Tree Well-formed Probability Model. We present two experiments with different numbers of clusters. The results of the experiment are shown in Table 5.18. Using 10 clusters, we observe an improvement in BLEU only on the development set. On both sets the TER decreases. With 20 and 30 clusters all evaluation measures are worse.

In Table 5.19, we present a translation example using Poor Man's Syntax and

source	我们 需要 通过 对话 来 解决 事端 而 不是 暴力 .
baseline	we need through dialogue to resolve the incidents and not violence .
syntax simple penalty	we need to resolve incidents through dialogue and not violence .
syntax probability model	we need to resolve incidents through dialogue and not violence .
reference (3 of 4)	we need to resolve this issue through dialogue , not violence .

Table 5.16. Translation example from NIST development set

systems	# of different translations
baseline, syntax simple penalty	1230
baseline, syntax probability model	1276
syntax simple penalty, syntax probability model	1266

Table 5.17. Number of different translations on the NIST test set

compare it with the other methods.

Table 5.20 gives an overview about the number of different translation produced by the Poor Man's Syntax. This method generates a huge number of different translations compared to our standard hierarchical system.

		dev		test	
	NTs	BLEU [%]	TER [%]	BLEU [%]	TER [%]
baseline	2	27.6	66.2	22.2	69.3
+ syntax poor man 10	2/10	28.0	65.8	22.3	68.7
+ syntax poor man 20	2/20	27.4	66.2	22.1	69.7
+ syntax poor man 30	2/30	27.0	67.0	21.8	70.2

Table 5.18. Results on NIST Chinese-English using Poor Man's Syntax

		TER [%]
source	阿 巴 斯 敦促 欧盟 继续 对 加 沙 的 巴勒斯坦 民众 提供 人道主义 援助 .	-
baseline	abbas urged the eu to continue to provide humanitarian aid to gaza with the palestinian people .	31.3
+ syntax simple penalty	abbas urged the european union to continue the palestinian people providing humanitarian aid to the gaza .	37.5
+ syntax probability model	abbas urged the european union to continue to provide humanitarian aid for gaza the palestinian people .	43.8
+ syntax poor man 10	abbas urged the eu to continue its gaza humanitarian assistance to the palestinian people .	18.8
+ syntax poor man 20	abbas urged the eu to continue for gaza the palestinian people and providing humanitarian aid .	31.25
+ syntax poor man 30	abbas urged the eu to continue its gaza the palestinian people and providing humanitarian aid .	25.00
reference (3 of 4)	abbas urged the eu to continue its humanitarian aid to the palestinian people in gaza .	-

Table 5.19. Translation example from NIST development set using Poor Man's Syntax

systems	# of different translations
baseline, syntax poor man 10	1178
baseline, syntax poor man 20	1143
baseline, syntax poor man 30	1265
syntax poor man 10 , syntax poor man 20	1152
syntax poor man 10 , syntax poor man 30	1251
syntax poor man 20 , syntax poor man 30	1226

Table 5.20. Number of different translations on the NIST test set using Poor Man's Syntax

6 Conclusion

In this section, we summarize our work presented in this thesis and give an outlook about future work on this topic.

6.1 Summary

In this work we have described two approaches to extended our hierarchical phrase-based system using additional information extracted from deep syntactic parse trees of the target language. In addition, we have presented a new method to generate meta syntax information by clustering phrases.

The deep syntactic parse trees were extracted by a syntax parser from the target language of the training data before the actual translation process. With the syntax information, we extended the grammar which is used as a translation model by our hierarchical phrase-based system. With this modified grammar, the decoding process was guided further and it appears that a more structured and grammatically correct translation was produced, as we assumed.

The first approach produced an augmented grammar by substituting the generic non-terminals with syntactic categories of the syntax trees. For this approach, we had to modify the CYK+ algorithm to obtain a translation for each input sentence. Results of our experiments on IWSLT show an improvement in Bleu and Ter. The disadvantage of this method was the high runtime and memory complexity.

To avoid this higher runtime and memory usage, we described a second approach which did not change the grammar, but used the syntax information in an additional model. Therefore, we could use the CYK+ algorithm as before. Also, we described two different features using the syntax information. The runtime and memory complexity of this approach is comparable to our standard hierarchical system. Therefore, experiments on medium size corpora was possible. The results on QUAERO showed no improvement, but on NIST we got a slight improvement.

In this thesis we also presented a new approach to generate non-terminals by clustering similar phrases. The motivation was to be independent from external parsers of the target language. First experiments shows an improvement in Ter.

In conclusion, it seems that the extension with syntax information can yield a benefit for language pairs that are sufficiently non-monotonic like English-Chinese. This was also observed in [Zollmann & Venugopal$^+$ 08]. That could also be a reason why we got no improvement on QUAERO. Language pairs like German-English have a similar syntactic structure.

6.2 Future Work

There are a number of questions which are beyond the scope of this work. Following topics should be researched further:

In this work we used the Stanford Parser to parse the target language. The parser is trained on a special corpus and does not take the given training data into account. Therefore, the influence of the parser on the translation quality should be investigated.

Also, the approach to generate non-terminals by clustering phrases seems promising. In this work, we did a hierarchical clustering by mapping all words to a word group, and we clustered the phrases afterwards. However, we mapped the words to a small number of word groups to do a fast clustering. Therefore, in further experiments the number of words groups should be increased.

Appendix A

Hierarchical Phrase-based Translation

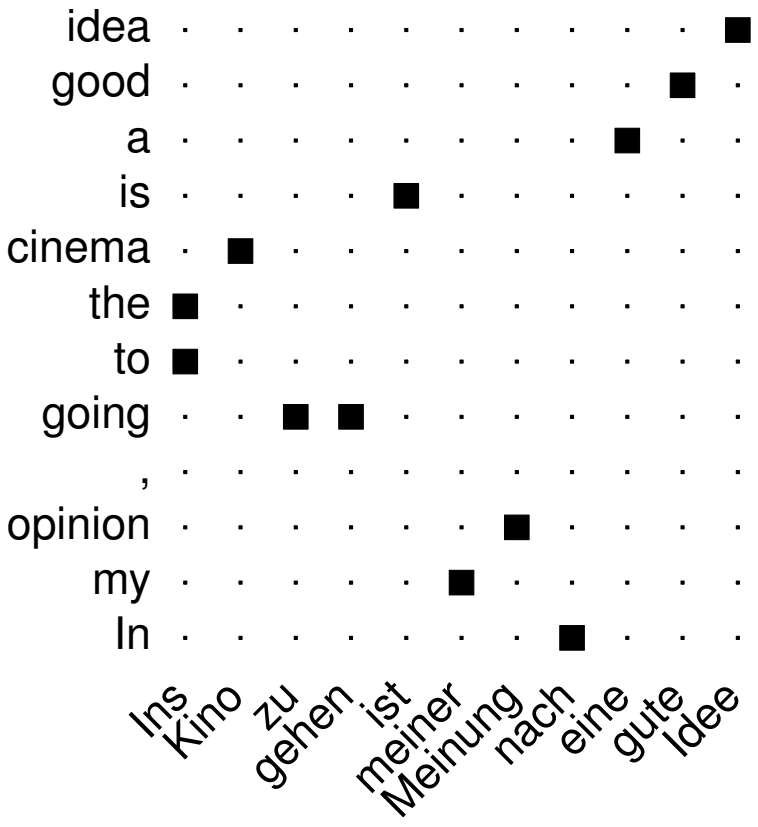

Figure A.1. Alignment of a sentence

source phrase	target phrase
Ins	to
Ins	to the
Ins	the
Ins Kino	to the cinema
Ins Kino zu gehen	, going to the cinema
Ins Kino zu gehen	going to the cinema
Ins Kino zu gehen ist	, going to the cinema is
Ins Kino zu gehen ist	going to the cinema is
Ins Kino zu gehen ist meiner Meinung	my opinion , going to the cinema is
Ins Kino zu gehen ist meiner Meinung nach	In my opinion , going to the cinema is
Ins Kino zu gehen ist meiner Meinung nach eine	In my opinion , going to the cinema is a
Kino	cinema
zu	going
zu gehen	, going
zu gehen	going
gehen	going
ist	is
meiner	my
meiner Meinung	my opinion
meiner Meinung	my opinion ,
meiner Meinung nach	In my opinion
meiner Meinung nach	In my opinion ,
Meinung	opinion
Meinung	opinion ,
nach	In
eine	a
eine gute	a good
eine gute Idee	a good idea
gute	good
gute Idee	good idea
Idee	idea

Figure A.2. List of extracted phrases

Appendix B

Extending Statistical Machine Translation Using Syntax

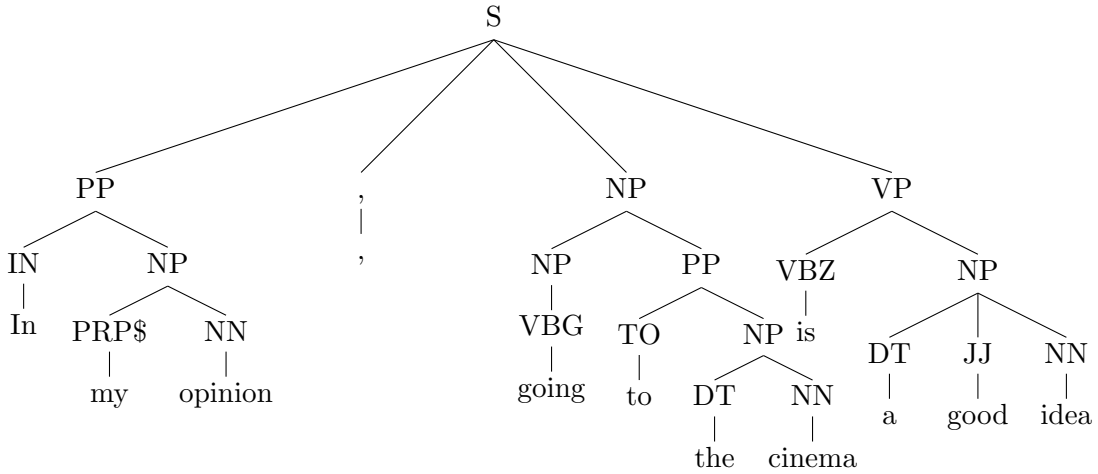

Figure B.1. Complete syntax tree for sentence "In my opinion , going to the cinema is a good idea"

Clause Level	Phrase Level	Word Level
S	ADJP	CC
SBARQ	ADVP	CD
SBARQ	CONJP	DT
SINV	FRAG	EX
SQ	INTJ	FW
	LST	IN
	NAC	JJ
	NP	JJR
	NX	JJS
	PP	LS
	PRN	MD
	PRT	NN
	QP	NNS
	RRC	NNP
	UCP	NNPS
	VP	PDT
	WHADJP	POS
	WHAVP	PRP
	WHNP	PRP$
	WHPP	RB
	X	RBR
		RBS
		RP
		SYM
		TO
		UH
		VB
		VBD
		VBG
		VBN
		VBP
		VBZ
		WDT
		WP
		WP$
		WRB

Table B.1. Labels

List of Figures

List of Tables

Bibliography

[Ann Bies & MacIntyre 95] K. K. Ann Bies, Mark Ferguson, R. MacIntyre. Bracketing guidelines for treebank ii style penn treebank project. Technical report, Department of Computer and Information, University of Pennsylvania, Pennsylvania, USA, 1995.

[Brown & Cocke$^+$ 90] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, P. S. Roossin. A Statistical Approach to Machine Translation. *Computational Linguistics*, Vol. 16, No. 2, pp. 79–85, June 1990.

[Chappelier & Rajman 98] J.-C. Chappelier, M. Rajman. A generalized cyk algorithm for parsing stochastic cfg. In *TAPD'98 Workshop*, pp. 133–137, Paris, France, April 1998.

[Charniak & Knight$^+$ 03] E. Charniak, K. Knight, K. Yamada. Syntax-based language models for statistical machine translation. In *In MT Summit IX. Intl. Assoc. for Machine Translation*, 2003.

[Chiang 05] D. Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pp. 263–270, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[Karypis 07] G. Karypis. Cluto - software for clustering high-dimensional datasets, 2007. `http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview`.

[Klein & Manning 08] D. Klein, C. D. Manning. Cluto - software for clustering high-dimensional datasets, 2008. `http://nlp.stanford.edu/software/lex-parser.shtml`.

[Och & Ney 00] F. J. Och, H. Ney. Giza++: Training of statistical translation models, 2000. `http://www-i6.Informatik.RWTH-Aachen.DE/~och/software/GIZA++.html`.

[Och & Ney 02] F. J. Och, H. Ney. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 295–302, Philadelphia, PA, July 2002.

[Och & Ney 03] F. J. Och, H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, Vol. 29, No. 1, pp. 19–51, March 2003.

[Och & Ney 04] F. J. Och, H. Ney. The alignment template approach to statistical machine translation. *Computational Linguistics*, Vol. 30, No. 4, pp. 417–449, December 2004.

[Och 00] F. J. Och. mkcls: Training of word classes for language modeling, 2000. `http://www-i6.informatik.rwth-aachen.de/~och/software/mkcls.html`.

[Och 03] F. J. Och. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pp. 160–167, Sapporo, Japan, July 2003. Association for Computational Linguistics.

[Papineni & Roukos$^+$ 02] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 311–318, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[Santorini 90] B. Santorini. Part-of-speech tagging guidelines for the penn treebank project. Technical Report MS-CIS-90-47, Department of Computer and Information, University of Pennsylvania, Pennsylvania, USA, 1990.

[Shen & Xu$^+$ 08] L. Shen, J. Xu, R. Weischedel. A New String-to-Dependency Machine Translation Algorithm with a Target Dependency Language Model. In *46rd Annual Meeting on Association for Computational Linguistics*, pp. 577–585, Columbus, Ohio, June 2008.

[Snover & Dorr$^+$ 06] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, J. Makhoul. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, August 2006.

[Venugopal & Zollmann$^+$ 09] A. Venugopal, A. Zollmann, N. A. Smith, S. Vogel. Preference grammars: softening syntactic constraints to improve statistical machine translation. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 236–244, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[Vilar & Stein$^+$ 08] D. Vilar, D. Stein, H. Ney. Analysing soft syntax features and heuristics for hierarchical phrase based machine translation. In *International Workshop on Spoken Language Translation*, pp. 190–197, Waikiki, Hawaii, Oct. 2008.

[Yamada & Knight 02] K. Yamada, K. Knight. A decoder for syntax-based statistical mt. In *ACL '02: Proceedings of the 40th Annual Meeting on Association*

for Computational Linguistics, pp. 303–310, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[Zens & Och$^+$ 02] R. Zens, F. J. Och, H. Ney. Phrase-Based Statistical Machine Translation. In M. Jarke, J. Koehler, G. Lakemeyer, editors, *25th German Conf. on Artificial Intelligence (KI2002)*, Vol. 2479 of *Lecture Notes in Artificial Intelligence (LNAI)*, pp. 18–32, Aachen, Germany, September 2002. Springer Verlag.

[Zollmann & Venugopal 06] A. Zollmann, A. Venugopal. Syntax augmented machine translation via chart parsing. In *Proceedings of the Workshop on Statistical Machine Translation*, pp. 138–141, New York City, June 2006. Association for Computational Linguistics.

[Zollmann & Venugopal$^+$ 08] A. Zollmann, A. Venugopal, F. Och, J. Ponte. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical mt. In *COLING '08: Proceedings of the 22nd International Conference on Computational Linguistics*, pp. 1145–1152, Morristown, NJ, USA, 2008. Association for Computational Linguistics.