An Empirical Account of Compositionality of Translation through Translation Data

MSc Thesis *(Afstudeerscriptie)*

written by

Dieuwke Hupkes
(born August 1st, 1989 in Wageningen, the Netherlands)

under the supervision of **Dr Khalil Sima'an**, and submitted to the Board of Examiners in partial fulfillment of the requirements for the degree of

MSc in Logic

at the *Universiteit van Amsterdam.*

Date of the public defense:
18 December, 2013

Members of the Thesis Committee:
Dr Willem Zuidema
Dr Jakub Szymanik
Dr Rens Bod
Dr Henk Zeevat

Institute for Logic, Language and Computation

Contents

1 Introduction **1**

2 Background: an Overview of Machine Translation **6**
2.1 A brief History of Machine Translation 7
2.2 IBM models 11
2.3 Phrase-based models 13
2.4 Synchronous Context Free Grammars 14
2.4.1 Formally 15
2.4.2 The computational difficulties of SCFG's 16
2.4.3 Learning SCFG's 17
2.5 Beyond Context Free 19
2.5.1 Synchronous Tree Substitution Grammars 19
2.5.2 Semantic Mappings 20
2.6 Conclusion 20

3 Empirical Research on Transfer Models **21**
3.1 Compositionality of Translation 22
3.1.1 Monolingual Compositionality 23
3.1.2 Bilingual Compositionality, issues 24
3.2 Investigating Compositionality of Translation 25
3.2.1 Skeletons of Compositional Translation trees 26
3.2.2 Labelled Trees 27
3.3 Establishing Translational Equivalence 28
3.3.1 Types of Word-Alignments 29
3.3.2 Obtaining Word Alignments 30
3.3.3 Translation Equivalence through Word-alignments 32
3.4 Compositional Translation Structures 32
3.4.1 Alignment Trees 33
3.5 Empirically Studying Compositionality 36
3.5.1 Linguistic Questions 37
3.5.2 Formal Questions 39
3.6 Dependency Parses 39
3.6.1 Background 40
3.6.2 Formally 41

3.7 Summary 42

4 An Empirical Account of Compositionality 44
4.1 Summary and Objectives 44
4.2 Set-up 47
4.3 Foundations of Empirical Studies 47
4.3.1 Correctness of the Translation Data 48
4.3.2 Correctness of Word Alignments 49
4.3.3 Correctness of Dependency Parses 49
4.4 Translation Structures 49
4.4.1 HATs 49
4.4.2 Motivation for using HATs 50
4.4.3 Representational Aims 52
4.4.4 Representation 53
4.4.5 Generation 54
4.5 Dependency parses 55
4.5.1 Representation 55
4.5.2 Generation 56
4.6 Summary 56

5 A Bilingual Perspective on Dependency Parses 57
5.1 Data 57
5.2 Direct Consistency 58
5.2.1 Experiment 1 59
5.2.2 Results 60
5.2.3 Analysis 61
5.3 Deeper Consistency 62
5.3.1 Experiment 2 64
5.3.2 Results 65
5.3.3 Analysis 65
5.4 Manual Analysis 66
5.4.1 English-German 67
5.4.2 French 70
5.5 Summary 72

6 Discussion and Future Work 74
6.1 Discussion 74
6.1.1 Compositional Translation from an Empirical Perspective 75
6.1.2 Finding Consistency 75
6.1.3 Summary of Results 76
6.1.4 Conclusion 76
6.2 Learning a Compositional Grammar using Dependency Information 77
6.2.1 Comparing structures 77
6.2.2 Learning the grammar 80
6.3 In conclusion 82

Abstract

Both in theoretical and applied research of machine translation it is often assumed that translation between natural languages can be treated in a compositional fashion, but it has proven far from trivial to develop a compositional translation system, or theoretically show it exists. In this thesis, an empirical investigation of compositionality of translation is presented, of which the main purpose is to find empirical evidence for the compositionality of actual translation data in the form of parallel corpora.

All maximally compositional translation structures of sentences in parallel corpora aligned at the word level were studied, to gain information about the system that generated them. In particular, it was studied whether monolingual information from dependency parses could be the basis of this underlying system.

Experiments showed that hardly over fifty percent of the dependency relations were preserved during translation if no modifications in the dependency relations were allowed. Considering deeper versions of dependency parses boosted this score with over thirty percentage points for all datasets. A manual analysis showed that most of the structure deviations were caused by errors in the data or systematic differences between the languages.

The results are encouraging for pursuing development of compositional translation systems based on dependency parses. A proposal for doing so is presented in the discussion of this thesis. Tools to execute this proposal, as well as tools to conduct further empirical research, have been made available.

Chapter 1

Introduction

Language and meaning play an important role in many aspects of our lives. When means of transportation and communication over larger distances became more publicly available, contact with other cultures became more prevalent and being able to understand how meanings are expressed in other languages than ones mother tongue became more important. Translation has something intriguing, as it seems to touch on something that is universal for all human beings, but is yet, even for human beings, a very nontrivial task. We would like to start this thesis with a famous quote of Warren Weaver that we believe has, besides the author of current work, inspired many to pursue a career in machine translation:

> *"When I look at an article in Russian, I say: 'This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode."* (Weaver, 1955)

Evidently, automatic translation is not as easily solved as Weaver thought at the time. Over 60 years later, the state-of-the-art systems are still not able to produce translations of arbitrary pieces of text with a quality comparable to that of a human translation. This was not for the lack of trying: on Google Scholar one can find over 100,000 articles that contain the phrase "Machine Translation", of which almost 10,000 were published in the last two years.

Compositional Translation

One of the many different methods that have been explored over the years is called the transfer method. Contrary to more direct approaches that treat sentences as structureless sequences that can be translated into a sequence of words in another language more or less word for word, the transfer method aims to find structural representations for sentences in the source and target languages and a mapping between them. The translation process then consists of analysing the source sentence into a structure, mapping this structure to a

target side structure, and generating a target sentence from this structure. A graphical representation of this process is shown in Figure 1.1. The depicted pyramid shows that direct (word for word) translation can be seen as an extreme version of the transfer method, in which the distance from the sentences to the intermediate representation is zero, and thus no analysis or generation takes place. On the other end of the spectrum we can find the case in which the intermediate representation is a universal one, independent of source and target language, and the mapping is the identity mapping. Such a universal intermediate representation of language is called an 'interlingua'.

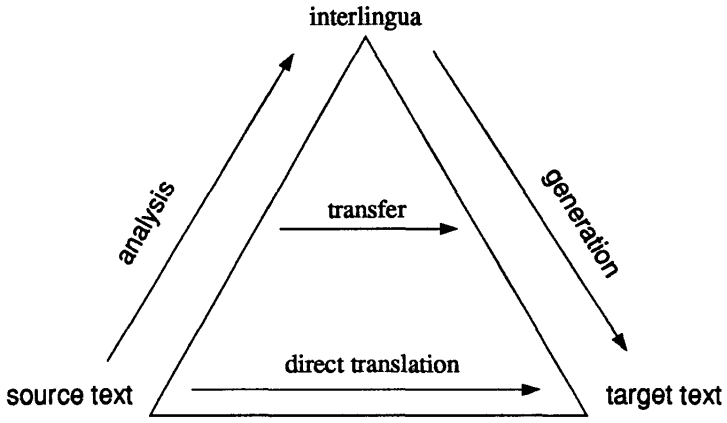


Figure 1.1: Vauquois pyramid

The pyramid also shows that transfer can be employed in many ways, varying the distance from intermediate representation to source and target language (analysis and generation, respectively). In this thesis, we consider the case in which the intermediate representations are interpreted as structural descriptions specifying how to derive the meaning of a sentence from the meaning of its parts.[1] In other words, the underlying system generating these structures can be seen as a semantically motivated syntax of the language that specifies how the sentence was (compositionally) constructed with respect to its meaning. Hereby, it is important to notice that the system describing these structures gives a recursive notion of meaning, and a mapping between source and target structures that utilizes this fact can thus also be seen as compositional.

The type of transfer that matches compositional translation is theoretically interesting, as it addresses translation on a very fundamental level. However, although very many translation models have tried to incorporate transfer as such (e.g., Wu, 1997; Chiang, 2005), and compositional methods of translation are theoretically well studied (e.g., Janssen, 1996), it is not fully understood if translation from natural language to natural language can be treated in such a

[1] Theoretically, we believe this is the only interesting case of transfer, although transferring other types of information might be useful in practice.

compositional fashion. In translation between other domains (logical languages, programming languages), compositional translation is almost trivially a sound approach, as the expressions of such languages are completely unambiguous and the compositional system (also called a grammar) according to which the meaning can be derived is known. Natural language has neither of these qualities, which does not only complicate the construction of a translation system, but also renders the existence of such a system uncertain.

The Adequacy of Compositional Translation

In this thesis, we will investigate the adequacy of compositional translation as a method for translation between natural languages. As this is a very wide and well investigated question, we do not expect to solve the matter in one master thesis. Rather, we will focus on one subproblem. The research questions asked, as well as the strategies for assessing them, are founded in four core observations.

1. The adequacy of compositional translation seems hard to assess by application. The soundness of the approach can only be confirmed - by a machine successfully carrying out translation - and not refuted. Although many researchers have tried, the MT world is far from presenting such a machine.

In theory, on the contrary, the soundness of compositional translation can only be refuted - by finding examples that cannot be translated as such - and not be confirmed. However, it turns out that theoretical examples of non-compositional translations (e.g., idiomatic translations), can often be dealt with elegantly in practice (Janssen, 1996). Neither of these approaches thus seem to be promising with respect to determining whether it is possible to systematically construct structures for two languages and a mapping between them.

2. The availability of huge parallel corpora with texts that are manual translations of each other, together with techniques to align these corpora on the sentence and even word level, provide a rich source of information about translation. Alignments specify, on different levels of granularity, which units are each others translation and can therefore be interpreted as constraints on the compositional structures according to which the sentence was possibly translated. These constraints, and the set of compositional structures they give rise to, can be investigated, resulting in a more suitable strategy of assessing compositional translation: empirical research.

3. Previous empirical studies of translation data can be roughly divided into two categories: formal studies and linguistic studies. The former studies focus on the formal properties of the structures alignments enact. For instance, prior studies have focussed on the coverage of structures that are completely binary (e.g., Søgaard and Kuhn, 2009). While such studies provide insight in the complexity of the transformation phenomena that occur during translation, their

purely bilingual perspective requires arbitrary choices to prefer one compositional structure over the other.

Linguistic studies, on the other hand, study the extent to which conventional *monolingual* linguistic syntax oversteps the constraints set by alignments(e.g., Fox, 2002; Hwa et al., 2002). Under this perspective, informed choices can be made as to which structures should be included in a grammar, but when alignments and monolingual syntax deviate, there is no solution available to solve this. The literature seems to lack a study that combines the strength of both approaches.

4. Given their cognitive motivation and semantic nature, dependency parses seem a suitable monolingual formalism to motivate bilingual choices. Furthermore, investigating dependency parses through translation data will offer a wider perspective on its universality. However, there are no thorough studies that assess the usefulness of dependency parses for translation that take a general perspective. Hwa et al. (2002) investigated whether dependency parses can be projected from English to Chinese, but did not account for unaligned words or phrasal translations. Fox (2002) investigated how well the phrases suggested by dependency parses cohere during translation from English to French, but constituency structures were the main focus of her paper, and her only conclusion related to dependency parses was that they seemed more cohesive than constituency grammars.

Research Questions and Plan

The previous observations resulted in the following research questions:

1. Are the compositional structures suggested by dependency parses universal for language?
2. What are the reasons dependency structures deviate during translation?
3. Can dependency parses be used to construct a bilingual compositional grammar?

We will address these questions empirically, by investigating the coherence between the dependency parse of a sentence (monolingual linguistic perspective) and *all* maximally compositional translation structures that are in agreement with its alignment[2] (bilingual formal perspective). We will investigate whether dependency relations are generally preserved during translation, and what the main causes are for parses to break down during translation. Finally, we will propose a method for combining the information from dependency parses and HATs to learn a bilingual grammar.

[2]As previously defined in Sima'an and Maillette de Buy Wenniger (2013), where they are called HATs

Contributions

The contributions of this thesis are twofold. Firstly, we will investigate the preservation of predicate argument structures across different language pairs. As this yields an empirical view on the universality of such structures, this is interesting for the field of linguistics in general. Furthermore, the results of such research are interesting for machine translation, as they broaden the perspective on the adequacy of compositional translation as a strategy for translating from one natural language to another.

Secondly, we present an open source implementation that can be used for further research in the same direction, but also to enrich translation corpora with recursive translation structures, motivated by linguistic intuitions. The resulting structurally consistent treebank for a parallel corpus might serve as a starting point for a new MT model, and the tools provided to learn and train a grammar from this treebank could prove useful for developing one.

Thesis Outline

This thesis is structured as follows. In Chapter 2, background information about the field of machine translation is provided. The chapter gives a general overview of the developments that led the field to its current state, and discusses techniques and models that are relevant for this thesis in more detail. In the subsequent chapter, empirical research of compositional translation is discussed. The chapter starts with setting out the main assumptions underpinning compositional translation, subsequently discusses how these can be empirically evaluated, and summarises related empirical research. The chapter ends with a background section on dependency grammars. In Chapter 4, the research questions of this thesis will be revisited, and more extensively discussed. Furthermore, Chapter 4 presents the basic algorithms and strategies that are used to answer these questions. The actual experiments, as well as their results, will be discussed in Chapter 5. In Chapter 6, we will look back at the thesis, discuss its results, and propose a method for overcoming the problems that were encountered in previous chapters. Examples from the data and documentation of the implementation can be found in Appendix A and B, respectively.

Chapter 2

Background: an Overview of Machine Translation

In this thesis, we address the universalities of language on an empirical level, through studying translation data. On some level, this thesis is thus related to linguistics in general. The research field to which this thesis is closest, however, is machine translation (MT). Not only do we build on previous empirical research conducted in the MT world, we also use the corpora, techniques and tools from this field, and our results are closely linked to - as well as interesting for - a specific type of MT models: structure based models. This chapter, in which the MT research field is discussed, serves several purposes, which we will set out in the following two paragraphs.

The chapter is divided into two parts. In the first section, an overview of MT is presented. We describe the developments in the field since the very first attempts, exemplifying the apparent cycle in the types of models that were investigated. Besides serving as a background sketch that helps the reader to put the thesis in perspective, this overview is meant to help the reader understand why all current state of the art MT models are structure based, and thus motivates the relevance of gaining a better understanding of such models. We also hope, that an overview of the various approaches tried in MT will help the reader appreciate the difficulty of the field, as well as its current state. This chapter is not meant as a thorough and complete overview of everything that has happened in MT over the years. The field is enormous, and discussing all this elaborately would not serve the purpose of this thesis. For instance, neither decoding nor technical implementation details of the models are discussed at all, as they are irrelevant for an empirical analysis of compositionality. For more elaborate overviews of MT, Hutchins and Somers (1992) (early MT), Somers (1999) (exemplar based MT), Koehn (2008) or Wu (2005) (statistical MT) can be consulted. These works have also all been used as references to write this chapter.

In the second part of this chapter (Sections 2.2 to 2.5), the most important

models in the history of MT are discussed in more detail (in the order they were presented in the overview). These sections provide a better insight in the many approaches that were investigated in MT, and in doing so further emphasize why current MT systems - which will be discussed in the last two of these sections - are in their current state. Furthermore, they will introduce the reader to the techniques used in these models and the ideas on which they are founded. Many of these techniques and ideas are still very important for MT in general, and for this thesis in particular.

As the current chapter provides a summary and literature study but does not present any original work, it might be superfluous for readers with an extensive knowledge of MT. These readers might leap to the end of this chapter, where a summary of the most important points is provided (Section 2.6).

2.1 A brief History of Machine Translation

Machine Translation arose as a research field almost immediately after the emergence of the first computers. In these early days, several different approaches were explored. These approaches can be roughly divided into direct translation models and structure based translation models, which we will discuss in the next two paragraphs.

Direct Translation
In one branch of research, translation was approached as an encoding problem. Such models with a direct approach to translation are now known as the first generation models. Sentences were translated more or less word for word using some contextual information. Figure 2.1 contains an example of the translation of the words 'much' and 'many' into Russian, according to one of the early systems (Dostert, 1955).

1 Is the preceding word *how*? (yes → сколько, no → 2)

2 Is the preceding word *as*? (yes → столько че, no → 3)

3 Is current word *much*? (yes → 5, no → 6)

4 Not to be translated

5 Is preceding word *very* (yes → 4, no → много)

6 Is preceding word a preposition, and following word a noun? (yes → многии, no → много)

Figure 2.1: Translation of much and many from English to Russian in a direct translation system (source Hutchins and Somers, 1992, p.56).

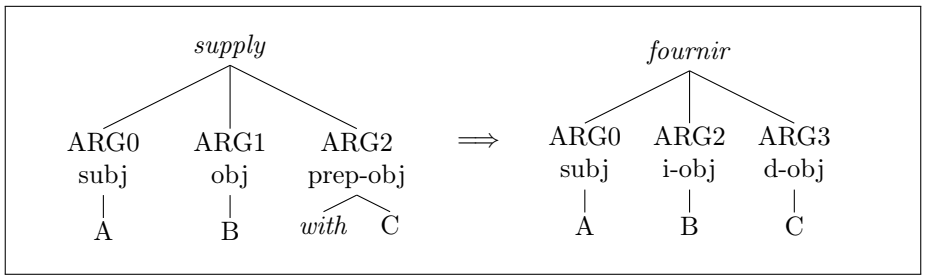


Figure 2.2: A transfer rule that accounts for the translation of 'A supplies B with C' into 'A fournit C à B' (source: Hutchins and Somers, 1992, p.230).

Structure Based Translation

A second line of research followed a more theoretical approach, involving fundamental linguistic research. The models of this strand aimed at translating language through using its underlying compositional structure. The most ambitious approaches aimed at finding representational systems, called interlingua, to express meaning on an abstract level and translate via such abstract meaning representations (thus finding an abstract meaning representation for the source text and generating a target text from this representation). The results of such models were disappointing, and the general consensus in the MT community was that the less ambitious transfer approach, in which different structural representations for source and target language were used, had the best prospects for significant improvements. In such transfer models, an extra stage is added to the process: the source text is first analysed into a structural representation containing information about the meaning and structure of the text and then this representation is mapped to a target side representation from which a target text can be generated. An example of a transfer rule used in the system Ariane (Boitet et al., 1982), is depicted in Figure 2.2.

Although the early systems were sometimes successful in small subdomains of language (e.g., Chandioux (1976) for meteorological forecasts), they failed to scale to bigger domains, as it is very hard to formalize all of language in one system. Driven by exactly this thought, a new line of research came off the ground, that was not primarily based on linguistic knowledge, but on large pairs of texts that were translations of each other.[1] Corpus based models can be roughly divided into exemplar based models and statistical models, although exemplar based models do not necessarily exclude the use of probabilistic techniques.

[1]Such parallel texts were not created for this purpose, but exist by the grace of multilingual societies. Parallel corpora can be created from, e.g., proceedings of governments that are kept in two languages, websites that are translated into multiple languages, and news organisations that publish their articles for a multilingual audience (Koehn, 2008). Techniques to align these text on the sentence level (e.g., Varga et al., 2007) have made such parallel texts very valuable for MT research.

EBMT
The keystone of exemplar based machine translation (EBMT), is the direct use of analogy. Sentences are translated by finding examples in the corpus similar to (fragments of) the sentence, and recombining them into a translation. Although most MT models are in some sense based on this general idea, the match-and-change method as typically used in EBMT is computationally not scalable, as huge databases need to be searched for similar phrases or sentences, and similarity is not easy to establish (or even define). To exploit the systematicities in translation of natural language, more sophisticated methods are required, that are often probabilistic or statistical in nature.

Statistical Word-based Models
The models called 'statistical models' take another perspective. Rather than basing translation on examples found in the corpora, they use the corpora to statistically decide on the parameters of another (mathematical) model. The first working statistical models (Brown et al., 1988, 1990, 1993) were ground-breaking. Although these models were intrinsically word-based, the quality of their translations was an enormous improvement over that of any earlier model. Equally important is, that the models, now known as 'IBM model 1-5', were able to output a translation for any given input sentence (even ungrammatical ones). The statistical framework takes the view that every sentence t is a possible translation of every sentence s. Modelling translation thus consists of modelling the probability $P(t|s)$ that t is a translation of s, and finding the sentence t for which this probability is highest. The probability distribution is statistically learned from the parallel corpus. A more detailed description of the IBM models, as well as information on how they deal with phenomena as reordering, can be found in Section 2.2.

Statistical Phrase-based Models
The statistical IBM models still had the same drawbacks as the first generation of direct translation models: no structure or local context was considered, and a large amount of natural language phenomena could therefore not be accounted for. With the introduction of phrases as basic units in translation models (Wang, 1998; Och et al., 1999) a major leap forward was taken towards a proper treatment of these problems. A phrase translation pair is a pair of contiguous source and target sequences such that the words in the source phrase are aligned only with words in the target phrase, and vice versa (Och and Ney, 2000). Phrases are thus not restricted to linguistic phrases, but can be any arbitrary contiguous sequence of words whose translation constitutes a contiguous sequence in the target sentence. Phrase-based translation models can therefore capture short contiguous idomatic translations, as well as small insertions and deletions and local reordering. E.g., both 'a casa' and 'o casa' are reasonable word for word translations of the English phrase 'the house'. However, 'o casa' is not a grammatical string in Portuguese. The latter observation could easily be captured by a phrase-based model, as 'the house' could be translated as one

unit, but would be much harder to model in a word-based model. Furthermore, a word-based model would never be able to get the correct idiomatic translation of 'bater as botas' (the Portuguese equivalent of 'to kick the bucket'), while a phrase-based model would have little trouble finding this translation, provided this specific idiomatic phrase was present in the training corpus. Phrase based models are discussed in more detail in Section 2.3.

Statistical Structure based Translation Models

Phrase based models, although still considered state of the art, suffer from the fact that no structure beyond the phrase level is taken into account. Approaches that addressed this problem by incorporating syntactic information to, e.g., sophisticate phrase selection of a standard phrase-based system (Koehn et al., 2003) or rerank its output (Och and Ney, 2004) were not very successful, which lead a large part of the MT community to move back to models similar to the earlier transfer based models that ruled the field before the emergence of statistical models. The newer transfer models stayed true to the statistical and corpus based tradition, in which translation is formulated as learning a probability distribution from a parallel corpus. While the rules in the old transfer models were constructed manually, the rules of the new batch of transfer models were based on the patterns found in translation corpora, making the models more scalable and robust (but leading to more limited mappings than before, which we will discuss in more detail in Section 2.4).

Translation structures, however, are yet another hidden component in translation data. They need to be inferred (or learned), which is not a trivial task. Some approaches tried to base the structures involved in translation on monolingual syntactic structures, by first parsing the source and target sentence into such a structure, and then try to establish correspondences between their nodes. This so-called 'parse-match-parse' method has a couple obvious limitations. Fully automated parsers that can parse large amounts of text efficiently are required for both source and target language, which severely limits the number of language pairs that can be treated with this approach. Furthermore, monolingual grammars are not designed for translation purposes, and there is no guarantee that source and target structures are similar enough to find correspondences for every part of them. How suitable monolingual syntax is for translation is a genuine question, that lies at the heart of this thesis, and will thus be discussed extensively in later chapters.

A second approach is to forget about linguistic information, and concentrate on the structures suggested by the translation data. Models using this strategy are based solely on alignments (see 3.3), that describe which source words are translated into which target words, and the restrictions they impose on structural representations of the sentence (more details are provided in the next chapter). As alignments generally give rise to a huge number of structures for every sentence, models using this method are hindered by computational issues. Several different solutions to restricting the structure space have been presented, in some of which formal criteria were used, while in others linguistic information

was incorporated. We will discuss these methods in Section 2.4. In this section we will also pay more attention to the practical side of such models.

Summary

We have given a brief overview of machine translation from the very start until now, that started with the direct first generation models, and ended with the structure based statistical models that are considered in this thesis. Before we will discuss the latter models in detail, we will first provide a more elaborate description of statistical word- and phrase based models (in Section 2.2 and 2.3, respectively). Statistical word- and phrase-based models laid the ground work for statistical transfer models, and a description of their techniques and ideas is thus indispensable to properly understand the rest of this thesis.

2.2 IBM models

In this section, a explanation of the main concepts used in the word-based models presented by Brown et al. (1993) will be provided. These models, which were the first working statistical models, laid the foundation for current state-of-the-art translation models. The IBM-models focussed on learning the probability distributions $P(t|s)$ - the probability that a target sentence t is a translation of a source sentence s - from a parallel corpus. That is, the predefined model for $P(t|s)$ has parameters that can be estimated from the translation data.[2] The hope is, that the learned distribution predicts translations in line with human intuitions. For instance, $P(t|s)$ should be high for (t, s) = ('I grow chilli plants in my backyard', 'ik kweek chili plantjes in mijn achtertuin'), and low for (t, s)= ('I grow chilli plants in my backyard', 'gisteren is mijn portemonnee gestolen').

Modelling P(t|s)

Brown et al. use Bayes' rule to split the translation probability into multiple probability distributions, yielding the following expression:

$$P(t|s) = \frac{P(t)P(s|t)}{P(s)}$$

As $P(s)$ does not depend on t, this results in the following equation (called 'The Fundamental Equation of Machine Translation' by the authors) for the desired translation $\hat{t}$:

$$\hat{t} = \arg\max_t P(t)P(s|t)$$

The equation splits the translation task in two: modelling the translation probability $P(s|t)$, and modelling the language probability $P(t)$. In Brown et al.

[2]For instance, $P(t|s)$ might be dependent on the probability distribution of the different translations of the word 'obvious' in the corpus.

(1993), 5 different models of increasing complexity are presented to model the translation probability. These models are generally referred to with the names 'IBM models 1-5'. In all these models, the probability $P(s|t)$ is modelled by marginalizing over all possible ways in which the words in t could have been generated by the words in s, which is expressed by an alignment function a (more information on which can be found in Section 3.3). Thus: $P(s|t) = \sum_a P(s, a|t)$. $P(s, a|t)$ cannot be computed exactly, and the 5 IBM models differ in the complexity of their approximation. For instance, in IBM model 1, all alignments are assumed to have an equal probability, and the probability $P(s, a|t)$ is the (normalized) product of all the lexical translation probabilities $p(s_j|f_{t(j)})$ indicated by the alignment. The translation probabilities for sentences t with the same words in different orders are thus identical. To address this issue, an alignment probability distribution is added in IBM model 2. In later models, also fertility of the input words is considered, and the word-order differences between source and target language are modelled more sophisticatedly. An example of IBM-style translation is depicted in Figure 2.3.

The parameters of the IBM models (e.g., lexical translation probabilities, fertilities of the words, alignment probabilities) are learnt from a parallel corpus using the expectation maximization algorithm (Dempster et al., 1977), on which a short explanation can be found in Section 3.3. Mathematical details on the exact procedure of parameter estimation for the IBM models can be found in Brown et al. (1993).

Consider the following pair of sentences and a possible alignment (the numbers indicate the alignment: (Le chien e battu per Jean, John (6) does beat (3,4) the (1) dog (2)). The probability $P(s, a|t)$ is computed as follows:

1. Compute the lexical probabilities of the source words being translated into the target words, thus compute: $P(Jean|John) \cdot P(est|beat) \cdot P(battu|beat) \cdots$

2. factor in the fertility probabilities of the source words, thus multiply with: $P(f = 1|John) \cdot \cdot P(f = 1|does) \cdot P(f = 2|beat) \cdots$

3. Factor in the distortion probabilities, that are in this model just depending on source and target position and target length, thus multiply with: $P(1|4, l = 6) \cdot P(2|5, l = 6) \cdot P(3|3, l = 6) \cdots$

The parameters for this IBM model are thus: a set of lexical translation probabilities $P(f|e)$, a set of fertility probabilities $P(n|e)$ and a set of distortion probabilities $P(i|j, l)$ for each target position i, source position j and target length l. In practice, i, j, l and n are maximally 25.

Figure 2.3: An example from (Brown et al., 1990, p.3), that shows the workings of the IBM word-based translation model

Modelling P(t)

The language model, a probability distribution for $P(t)$, is supposed to account for fluency and grammaticality of the target language string. That is, to prevent the model from putting too much probability mass on not well-formed target

strings. In the IBM models, the probability distribution is an n-gram model, whose parameters can be estimated through relative frequency estimation on the target side of the parallel corpus. The set-up in which a separate language model is used to assign probabilities to translations is used by almost every current state-of-the-art MT-model.

2.3 Phrase-based models

Phrase-based models address some of the limitations of word-based models, by using phrases instead of words as basic units in the translation models, which allows the translation model to take into account local context. For instance, a simple phrase based model would translate a sentence through the following sequence of steps. The foreign sentence is first broken up into phrases. These phrases are then translated as a whole, and the probability of the source sentence given the target sentence[3] is defined as the product of the phrasal translation probabilities and a 'distortion' probability based on how far every phrase was moved relatively to the previously translated phrase (Koehn et al., 2003). The probability of the target sentence given the source sentence can then be computed taking into account the language model.

To translate with phrases, a phrase translation table is needed in which probabilities are assigned to the translation of source phrases in target phrases. Phrase-tables can be acquired in different ways (Marcu and Wong, 2002; Och et al., 1999; Koehn et al., 2003; Mylonakis and Sima'an, 2008), details of which are not relevant to this thesis. An aspect that is more relevant to this thesis relates to the 'definition' of a phrase - in other words: when can a subsequence of a sentence pass for a phrase that can be used in translation - which will be discussed in Chapter 3.

Phrase-based translation has some obvious advantages over word-based translation. First of all, short idiomatic translation can be accounted for in an intuitive fashion: directly assigning a probability to 'of course' as the translation of 'natuurlijk' makes intuitively more sense than having two separate entries that assign probabilities to 'of' and 'course' being translations for 'natuurlijk'. Secondly, phrases can use local context, which means they can make informed decisions about the translation of, e.g., the gender of determiners and adjectives, of which an example was given in the overview in Section 2.1. Finally, phrases can capture local reordering phenomena of phrases seen during training, making it easier to prefer 'the Italian woman' over 'the woman Italian' as translation of 'la donna italiana'.

Phrase based models also have certain limitations, of both practical and theoretical nature. Firstly, phrase based models have no clear strategy to account for global reordering. Due to data sparsity, useful phrases are generally not much longer than 3 words, and can thus not account for such reordering phenomena

[3] A quick reminder: the generative model of phrase-based models is largely similar to the word-based IBM models, the translation probability is still inverted due to application of Bayes' rule.

on their own. Naturally, reordering models are included in most phrases based models, but the reordering space for phrases is, although significantly reduced with respect to word-based models, still too large to exhaustively search through all possibilities, and phrasal movement is generally only considered within a window of a couple of phrases. Global reordering of entire constituents that take up several phrases can thus not be captured with such reordering models.

A second issue concerns the partitioning of the sentence into phrases. The probability of this partitioning is rarely considered, and phrases are not allowed to overlap, resulting in poor modelling of agreement phenomena.

Another difficulty with phrase based models arises in the assignment of probabilities to phrase pairs, which is, as mentioned before, not straight forward. Several approaches have been used to learn phrase-translation tables (see Koehn, 2008, p.130).

Finally, phrase-based models have no means to detect systematicities in phrase structures that can help them to generalise beyond what they have seen in the training data. Even if a phrase based system has seen several examples similar to 'la donna italiana' (article adjective noun), it will not infer that adjectives and nouns in translation between Italian and English switch order in general.[4]

2.4 Synchronous Context Free Grammars

To address the global reordering problem, more structure needs to be incorporated, which prompted the revisiting of transfer models. To statistically exploit transfer methods, syntactic formalisms for both source and target side are needed, as well as a method of combining them. The transfer process in statistical based transfer models differs slightly from the previously sketched picture, in which entire source structures were mapped to entire target structures. As a rule, the source and target structures (or sentences) are assumed to be generated simultaneously (bit by bit) by two 'linked' monolingual grammars, that are together called a synchronous grammar. Figure 2.4 depicts an example of the simultaneous generation of a sentence and its translation with a synchronous (context free) grammar. Regarding the complexity of the monolingual grammars, there are several choices that can be made. Some translation models have incorporated relatively simple formalisms as finite state machines (e.g., Alshawi et al., 2000), others relatively heavy formalisms as tree adjoining grammars (e.g., Poutsma (2000) based a translation model on DOP). However, the lion's share of the statistical tree based transfer models uses synchronous context free grammars (SCFG's), and even approaches that are not explicitly concerned with CFG's can often be reformulated as such.

[4] Of course there are many situations in which phrases still can be used to capture this implicitly. If an unknown adjective-noun combination is to be translated (say 'la donna tedesca'), the model does not have to fall back on word for word translation, but can (in this case) combine the phrases 'la donna' and 'tedesca ...'.

2.4.1 Formally

An SCFG is the synchronous extension of a CFG, as introduced by Chomsky (1956):

Definition 1 (Context Free Grammar)**.**
A context free grammar (CFG) is a quadruple $G = (V, \Sigma, R, S)$, where

1. *V is a (finite) set of non-terminals, in the context of natural language often interpreted as syntactic categories.*
2. *Σ is a (finite) set of terminals, corresponding to the lexical items of the language.*
3. *R is a relation from V to $V \cup \Sigma$, to be interpreted as a set of rewrite rules.*
4. *$S \in V$ is the start symbol of the grammar.*

An SCFG (Aho and Ullman, 1969) is a grammar linking two CFG's that share a set of non-terminals, describing how their expressions can be generated simultaneously. Parse trees generated by SCFGs are thus isomorphic on the non-terminal level (i.e., there is a bijective mapping between the non-terminal nodes of the trees). Formally, we have:

Definition 2 (Synchronous Context Free Grammar)**.**
A synchronous context free grammar (SCFG) is a quadruple $G = (V, \Sigma, R, S)$, where

1. *V is a (finite) set of non-terminals, the syntactic categories of both languages.*
2. *Σ is a (finite) set of terminals, constituted by the union of the terminal symbols of the two languages.*
3. *R is a set of rewrite rules of the form $X \rightarrow \langle \gamma, \alpha, \sim \rangle$, $\gamma \in (V \cup \Sigma)^*$, $\alpha \in (V \cup \Sigma)^*$ and $\sim$ a one-to-one and onto correspondence between the non-terminal symbols in α and γ.*
4. *$S \in V$ is the start symbol of the grammar.*

SCFG's implicitly model large scale reordering phenomena and long distance dependencies, as the non-terminal sequences that are generated in a production do not necessarily have the same order. Figure 2.4 includes a small example of this: the noun and adjective swap order during the translation. Such swaps can also occur on larger scales and SCFG can account for quite complicated reorderings.

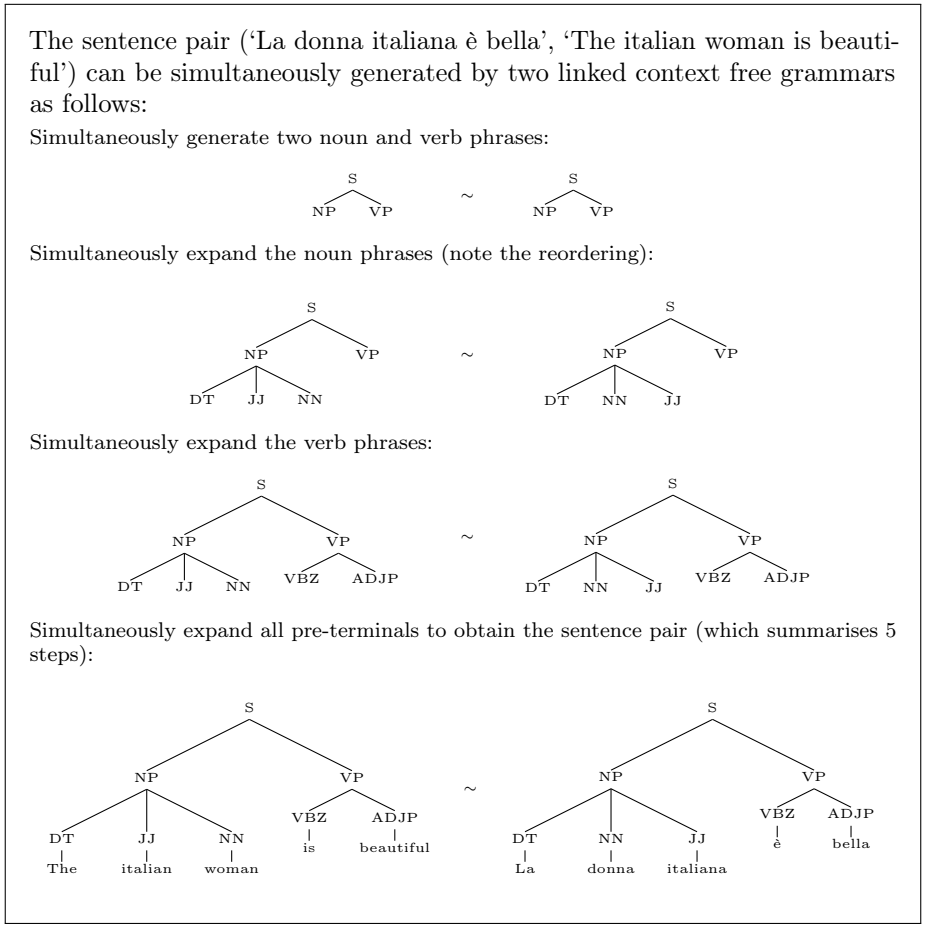


Figure 2.4: The synchronous generation of a sentence and its translation.

2.4.2 The computational difficulties of SCFG's

As most complexer MT models, SCFG's are severely hindered by computational problems. On of the most significant problems, which we will highlight because it has been a motivational factor in many algorithms for learning SCFG's (Zhang et al., 2006, 2008; Huang et al., 2009, e.g.,) relates to the rank of an SCFG. The rank of an SCFG can be defined as the highest number of non-terminals occurring on the right hand side of a rule in the grammar in a single dimension (Gildea et al., 2006), which is also called the rank of a rule. For instance, the toy rule 'NP $\rightarrow$ $\langle$ Det NN Adj, Det Adj NN $\rangle$' has rank 3, as three non-terminals are generated in both source and target language. As with monolingual CFG's, parsing with SCFG's is much more efficient if all the rules are binary (thus the rank of the grammar is 2). Contrary to monolingual CFG's, the rank of an

SCFG cannot always be reduced to two by rewriting the rules. In other words: synchronous CFG's can not always be binarised (Huang et al., 2009), which makes parsing very inefficient.

2.4.3 Learning SCFG's

Learning an SCFG from translation data is a far from trivial task, which is amplified by the fact that bilingual data often do not coincide with monolingual linguistic structures. That is, trees generated by SCFG's are isomorphic on the non-terminal level, which is often not the case if the source and target trees are independently generated by two separate non adjusted parsers. Furthermore, the parts considered constituents by a monolingual CFGs are not necessarily translation admissible parts according to the translation data. This is not to say, that monolingual linguistic syntax is useless for MT. In fact, its usability is precisely what is addressed in this thesis. However, to stay on the topic of synchronous grammars, translation models solely relying on monolingual syntactic structures generally exceed the power of SCFG's. We will briefly discuss such models in Section 2.5.

Rule Extraction

Learning an SCFG consists of determining its rules. In almost all working SCFG models, the grammar rules are induced from a parallel corpus by regarding the data as primary source of information (in contrast to using external, possibly linguistic, knowledge). This approach - introduced by Wu (1995) in the form of an inversion transduction grammar (ITG), that lies at the heart of many later approaches - is based on word-alignments and the constraints on structures (and thus rules) they prescribe. In Chapter 3, we will take a closer look at alignments and the structures they induce.

Purely data-driven models reinforce the computational problems of SCFG's, as the number of rules that can be extracted from a sentence grows exponentially with the length of the sentence (Quirk and Menezes, 2006b). Without considering additional information, there is no a priori reason to prefer one rule over another, yet some serious pruning of the rule space is necessary to make parsing computationally feasible. Furthermore, without linguistic information, a grammar naturally lacks non-terminal labels, which raises a new issue: inventing non-terminal labels. MT models differ in the number and kind of non-terminal labels that they use. In the remainder of this section we will briefly discuss two strategies that have been proposed to address the previously mentioned issues: the formal strategy, and the linguistic strategy. These strategies are not mutually exclusive, some models use them both. The explanation is meant to exemplify endeavours to address these problems, and although several references to models are provided it do not intend to give a complete overview of these. As before, we will not discuss the issue of assigning weights to the rules of the SCFG. Herewith we exclude the strategy of selecting rules by statistically learning which rules have a probability exceeding a certain threshold, which can be considered an

important part of the fields in many contexts. However, in the context of this thesis, knowledge of the different strategies in this area is not relevant, nor will it lead to a better understanding of the rest of this thesis.

Restricting according to Rank

A remedy that is often used to reduce the number of rules is to select only the rules whose rank does not exceed a certain maximum number. In most cases, the rules are restricted to binary (at most two right hand side non-terminals in one dimension), which reduces both the rule space and the parsing complexity (e.g, Wu, 1997; Chiang, 2005; Mylonakis and Sima'an, 2011). This solution is computationally attractive, and easy to implement.

Of course, the assumption that all of language can be captured in binary structures seems rather strong. Wu claimed to be unable to find real-life examples of translations that could not be explained by such trees, but this was later refuted by others (e.g., Galley et al., 2004). However, the coverage of binary transduction grammars is still a hot issue in MT, to which we will revisit later in Chapter 3.

Many of the models using this paradigm to prune the rule space do not really address the non-terminal label issue. Wu's (1995) ITG contains only a single non-terminal label, and his model thus merely learns a high-level reordering preference, without considering further contextual information. An improvement on this was presented by Chiang (2005, 2007). Although also his grammar had no more than one non-terminal label, he allowed the right-hand side of his rules to contain both terminals and non-terminals, such that lexical information could be incorporated. An example of such a rule would be:

$$\mathrm{X} \rightarrow \langle\ \mathrm{X}_1 \text{ de } \mathrm{X}_2, \text{ the } \mathrm{X}_2 \text{ that } \mathrm{X}_1 \rangle$$

which captures the fact that Chinese relative clauses modify noun phrases on the left, whereas English relative clauses modify on the right (Chiang, 2007).[5]

The framework introduced by Chiang combines the strengths of rule-based and phrase-based models, and is referred to with the term 'Hierarchical Phrase Based Translation'. To the knowledge of the author, there are no models that learn syntactic categories without invoking linguistic knowledge, with the exception of models following the previously mentioned probabilistic strategy of weighting rules (Mylonakis and Sima'an, 2010; Blunsom et al., 2008, e.g.,).

Incorporating Linguistic Information

A sensible solution to address the previously mentioned issues with structure-based transfer models is to incorporate linguistic knowledge. Information from

[5] To combine different non-terminals into a sentence, some more rules are needed. Chiang (2007) adds the following two 'glue rules' to his grammar:

$$\mathrm{S} \rightarrow \langle\ \mathrm{S}_1\mathrm{X}_2, \mathrm{S}_1\mathrm{X}_2 \rangle$$
$$\mathrm{S} \rightarrow \langle\ \mathrm{X}_1, \mathrm{X}_1 \rangle$$

monolingual parsers can be used to, e.g., reduce the space of possible node spans, and to induce linguistically motivated terminal labels. Researchers who followed such a strategy include Zollmann and Venugopal (2006), who augmented a standard phrase-based grammar with syntactically motivated non-terminal labels based on constituency grammars, Almaghout et al. (2010) who, following Hassan et al. (2007), labelled phrase pairs with ccg based supertags (Steedman and Baldridge, 2011) and Mylonakis and Sima'an (2011), who learned automatically which source-syntax labels fit best with a phrase-based SCFG and the translation data.

Clearly, the number of language pairs that can be treated as such is very limited, as it requires an automated linguistic parser (or other means of providing linguistic information on a large scale) for (at least one of) source and target language. However, using available syntactic or semantic knowledge can result in robust models that yet do not ignore our intuition of language, especially if high quality parsers are available.

2.5 Beyond Context Free

Formally it is desirable to create grammars that generate isomorphic tree pairs for sentences that are each others translation, but there is no a priori reason for the existence of such structures. In fact, as CFG's have been proved to sometimes be inadequate to model certain natural language phenomena, more powerful transformation methods might be more suitable for the expressive syntactic transformations going on during the translation of natural language. As the necessity of deviating from conventional syntax is smaller, models of this class tend to stay closer to traditional linguistic structures.

2.5.1 Synchronous Tree Substitution Grammars

The class of Synchronous Tree Substitution Grammars (STSG's) is a strict superset of the class of SCFG's, and STSG's are therefore a natural extension to them. Models working with STSG's are, i.a., Poutsma (2000) and Galley et al. (2004, 2006). The core method of the former is to align chunks of parse trees of source and target sentences, and transform them into rules. Poutsma (2000) requires the existence of a parallel corpus aligned on the subtree level. Such datasets were not available and the paper is merely a description of the STSG framework. The model presented by Galley et al. has a somewhat different set-up, learning rules to transform an source-language string into a target language tree. Galley et al. (2006) do provide an implementation, yielding promising results.

An approach that does not explicitly use STSG's, but whose grammar rules do exceed the power of CFG rules, is presented by Melamed et al. (2004). In their generalized multitext grammar (GMTG) they let go of the requirement that constituents need to be contiguous, which allows them to synchronise languages generated by mildly context-sensitive languages. Also Melamed et al. present a

framework with suggestions for further work, rather than an implementation.

2.5.2 Semantic Mappings

The last category of models we will discuss attempts to find mappings between more semantically oriented structures, that specify the predicate-argument structure of the sentence, which is often assumed to be somewhat universal. Such an approach is taken in Menezes and Richardson (2003), in which transfer rules are extracted by aligning pairs of Logical Form structures. Another predicate-argument structure that is often used is the dependency parse (for more information on the dependency parse, see 3.6), from which rules are inferred by either projecting or learning target-side paths. As such rules sometimes create or merge dependents according to the alignment, the dependency structures of source and target side need not be isomorphic, and such models can formally also be seen as STSG's (as made explicit in Eisner, 2003)). Finding a mapping between two dependency trees is not only attractive because dependency trees represent the semantic structure of a sentence more closely than a constituency tree, but also because it is computationally more feasible, because dependency trees contain fewer nodes than constituency trees of the same sentence. Known models differ in the linguistic plausibility of the target side dependency parse. E.g., Eisner (2003) learns mappings between two linguistic dependency trees (his article lacks a working implementation, although it does give a description of algorithms suitable for parsing with his model), while Lin (2004), extracts transfer rules that correspond to linear paths in the source side dependency tree, but not necessarily to linguistic dependency parses on the target side. The models presented in Quirk et al. (2005); Quirk and Menezes (2006b,a) also have clear dependency part, but employ several other strategies as well. They project source dependency trees to target dependency trees, following a set of rules, and extract from the resulting corpus a set of *treelets* - arbitrary connected sub graphs - that are used in translation.

2.6 Conclusion

In this chapter, we sketched the background of this thesis. We presented a brief overview of the history of machine translation, and we discussed some of the relevant models and techniques in more detail. The reader should now be acquainted with the basics of statistical models of translation, know what alignments and phrase pairs are, and appreciate that to adequately model reordering and long distance dependencies, incorporation of information about the structure of the language is essential. For the latter, we have seen that it is unclear how exactly this should be done. In particular, it became clear that the extent to which monolingual syntax is useful for such a process is unknown, and this question hence deserves further investigation.

Chapter 3

Empirical Research on Transfer Models

In the previous chapter we saw that rule-based MT models that incorporate information about the structure of language to model translation constitute an important branch of MT. Most of these models make use of SCFGs, which means they search for bijective mappings between the non-terminal nodes of the syntactic structures of both languages, and hence treat translation in a compositional fashion. Such models are not (yet) very successful, as finding an appropriate SCFG that simultaneously models two different languages is not easy. The search space of such grammars is enormous, and there are formally often no good reasons to prefer one rule over another. Using monolingual syntax to improve such models seems an obvious next step, but it has proved to be hard to adequately incorporate such information. Considering the compositional translation strategy, there are many gaps in our knowledge. We do not know whether the translation of natural language can be treated compositionally at all, how we should find the compositional system according to which it is translated, and to what extent monolingual syntax is helpful in this situation (a graphical picture of this situation is depicted in Figure 3.1). To improve current structure based translation models, an investigation of such questions is essential. In this chapter, we will lay the basis for such an investigation.

The chapter is structured as follows. In Section 3.1, we will link SCFGs to compositional translation, and discuss compositional translation and its difficulties on a rather abstract level, quite distant from actual language and data. In Section 3.2, we will discuss, on an intuitive level, how compositionality can be represented, and therefore identified, by providing some examples of compositional translation structures. In Section 3.3 and 3.4, we will discuss the main ingredients to find compositional translation trees, and how they can be interpreted, which lays the groundwork for an empirical investigation of translation data. In Section 3.5, we will discuss what questions can be addressed by empirical research, and discuss studies that were conducted by others. In

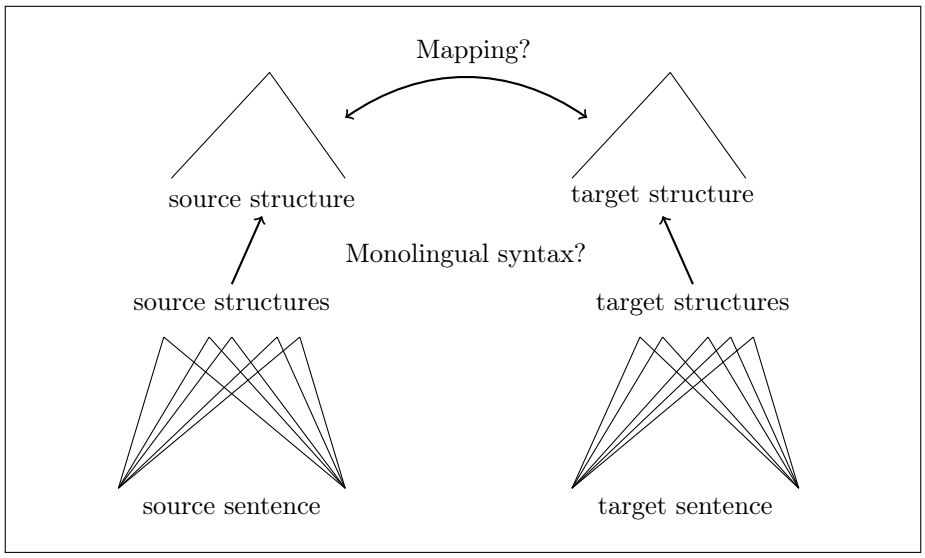


Figure 3.1: A graphical representation of the situation at hand in this thesis.

Section 3.6, we will discuss, very briefly, the remaining gap in this branch of research, and provide the last bit of background information needed for the research conducted in this thesis, by discussing dependency parses. The chapter ends with a short summary in Section 3.7

3.1 Compositionality of Translation

SCFGs assume that the translation of a sentence can be recursively constructed by combining the meanings of smaller units. This property of translation is described in a well known principle, called 'The Principle of Compositionality of Translation':

> **The Principle of Compositionality of Translation**
> Two expressions are each others translation if they are built up from parts which are each other's translation, by means of translation-equivalent rules. (e.g., Janssen, 1998)

Compositional translation is a very common method in translation between artificial languages (Janssen, 1996, 1998). The translation from one logical language into another, or the translation a compiler performs when interpreting a programming language are all defined in a compositional fashion. In some aspects, natural languages are very different from artificial languages, and transferring the methods from the artificial domain to the natural language domain is not straightforward. In this section, we will discuss the various

complications that arise when constructing a compositional translation system for language. These complications can be divided into two different groups. In the first group, there are the monolingual problems, that arise when trying to model a language with a compositional grammar. These issues are discussed in 3.1.1. The second group consists of issues that occur when trying to find a mapping between two monolingual grammars, which are thus of a more bilingual nature. We will discuss these issues in 3.1.2.

3.1.1 Monolingual Compositionality

The existence of a *linked* compositional grammar, prescribes the existence of monolingual compositional grammars. In other words, compositional translation requires two grammars adequately describing the source and target language. Many people would render language intuitively compositional, at least to some extent. We can all perceive a systematicity in the way sentences are constructed (often referred to with the term 'syntax'), and as we do not store the meaning of all possible sentences in our head, it seems reasonable to presume that we rely on this systematicity to derive the meaning of sentences we hear, using the meaning of the words and the methods we know for combining them (invoking our internal grammar). As for the cognitive existence of such grammars, there is no consensus among linguists. On the one hand, there is the Chomskian group of researchers, that advocate the existence of an underlying compositional grammar universal to all human beings (as first claimed in Chomsky, 1956), while others believe no such system exists, and language users rely on some sense of familiarity with what they have heard before in a much less complex fashion (quite recently, e.g., Frank et al., 2012). In this thesis we will not scrutinize cognitive debate, but focus on whether it is possible to create a compositional grammar at all, and the problems that arise when doing so.

In practice, it has shown to be very hard to create a grammar that covers *all* grammatical utterances of a natural language, without generating too many ungrammatical ones. Even for a finite (but reasonably sized) corpus, it is surprisingly difficult to construct a compositional grammar that adequately describes the corpus (Scha, 1990). The bigger the grammar grows, the more phenomena need to be taken into account, as well as how they interact with each other. As an example, idiomatic expressions tend to be problematic, as they behave differently than other expressions. Their meaning cannot be derived from its parts, which suggests they should be included as basic units in the system. However, idiomatic expressions can have an internal structure, on which syntactic rules can be applied. A verb in an idomatic expression, for instance, can often be conjugated. Solutions for many specific problems have been provided (in e.g., Janssen, 1996), but it remains hard to fully understand the effect of including more rules in the grammar, and to quantify how much of natural language such a grammar can successfully model.

A second major issue concerning the monolingual part of compositional translation, is ambiguity. In programming languages and logical languages, utterances typically have only one analysis, and their meaning is thus unambigu-

ous.[1] Under all current existing grammar formalisms that assign structures to natural language, all sentences (of some length) have many different structural analyses, of which usually only one or two are perceived by humans (Scha, 1990). Even sentences that are considered unambiguous by humans thus need to be disambiguated, for which an appropriate model is needed to make the grammar applicable.[2] In practice, the rules of compositional grammars are often given weights, such that probabilities can be assigned to different analyses of the sentence. Humans are assumed to do a similar thing when interpreting a sentence, which provides an explanation for why they do not perceive more analyses for sentences: they are disregarded due to their relative implausibility with respect to another analysis.

3.1.2 Bilingual Compositionality, issues

Besides the issues going on on a monolingual level, there are several issues that come to play when trying to apply the principle of compositionality of translation to translation between two natural languages. We will now discuss the issues that complicate the design of a compositional grammar on a bilingual level.

First of all, an assumption prevalent in the principle, is that in translation not only meaning, but also form should be preserved (as much as possible). In other words, it is assumed that translation is literal. For artificial languages this property is straight-forward and useful, mostly because there are no a priori reasons to prefer a non-literal translation over a literal translation. In natural language, however, this assumption is rather questionable. Although there are many occasions in natural language in which the assumption seems fairly applicable - it captures, for instance, the fact that 'all ravens are black' is an adequate translation of 'alle raven zijn zwart', while the logically equivalent 'if something is not black, it is not a raven' is not (Landsbergen et al., 1989) - but in practice a translator can have many reasons to prefer a free translation, even if a more literal alternative is also available.

As MT models do not (yet) focus on literary translations, but merely aim for (preferably grammatical) translations with the correct meaning, it seems reasonable to ignore the fact that the most literal translation is not always the translation that is stylistically preferred by a human translator. However, even with this additional assumption, there are many occasions in which a literal translation is simply not available. These situations fall into the category of syntactic and lexical translation divergences. Languages do not always express

[1] Precedence relations or 'left-to-right'-rules usually determine the order in which the syntactic rules need to be applied in case of uncertainty. In programming languages, expressions that are are multi-interpretable are often considered ungrammatical.

[2] Note that this problem differs from one of the standard counter arguments of compositionality, that concerns sentences like 'two men carry two chairs', that *are* considered ambiguous by humans, but cannot be assigned two distinct syntactic analyses capturing this difference (Pelletier, 1994). Not particularly relevant, but certainly nice to notice, is that this type of ambiguity is not necessarily problematic for translation, as it might be preserved. For instance, the Dutch translation 'twee mannen dragen twee stoelen' of aforementioned sentence has the same two meanings as the English one.

the same set of meanings, or express meanings in the same way. Even in languages of cultures that are quite similar, one can find a number of words that do not have an adequate translation in the other language (e.g., in translation between English and Dutch the words 'gezellig' and 'evidence' do not seem to have a clear equivalent in the other language), and even if the same meaning is expressed there are many syntactic phenomena in natural language that seem to be problematic for a compositional translation. For example: different ways of role expression (e.g., 'I like obj' and 'мне нравится or syntactic mismatches (e.g., 'woonachtig zijn' and its translation 'reside', Landsbergen et al., 1989). The grammar rules and basic units can thus not simply be taken from a monolingual grammar, as there is no guarantee that the rules and basic units will have a translation equivalent rule or basic unit in the other grammar. The grammars must be constructed for translation, such that they are 'attuned' (Rosetta, 1994). Rosetta (1994) showed that previously mentioned examples do not necessarily stand in the way of compositional translation. They manually constructed a grammar for translation from English to Dutch, that covered many non-trivial translation phenomena.[3] Once again, it is unclear if this can be done to cover translation for larger parts of language.

3.2 Investigating Compositionality of Translation

To determine the overall level of compositionality of language or translation, it does not suffice to present solutions for specific phenomena known to be problematic for compositionality, nor does developing another (unsuccessfully yet slightly better than another) model carrying out compositional translation. In this thesis, we will do neither of these things, but resort to a third option: empirical analysis. We will look at real evidence in the form of translation data, and try to establish its level of compositionality through an empirical analysis. Now that huge parallel corpora are available, it seems that empirical analysis is a very powerful tool, and it might prove more useful to assess the suitability of compositional translation in practice by using empirical analysis than by developing more models and evaluate based on their performance.

To conduct an empirical analysis of the compositionality of a huge translation corpus, means of identifying compositionality are necessary. That is, we need to be able to identify possible translation parts, as well as devise descriptions of the compositional translation of a sentence, efficiently and on a large scale. In this section, we will give an intuitive description of such trees and what information they contain, by providing some examples. In the remainder of this chapter, we will present tools to replace intuition efficiently and on a large scale, and discuss current empirical research.

[3] Although it must be said that their grammar is more complicated than the case we are considering now, as it consists of separate semantic and syntactic modules.

3.2.1 Skeletons of Compositional Translation trees

The structure of a compositional translation can be described by means of a tree, which we will elucidate in this subsection through tree examples.

A simple Compositional Translation Tree

Consider the simple example sentence 'I gave my little brother a ball' and its translation 'Ik gaf mijn kleine broertje een bal'. A possible compositional translation of this sentence is:

1. Combine the translation of 'a' and 'ball' to get the translation of 'a ball': 'een bal'.
2. Combine the translations of 'little' and 'brother' to get the translation of 'little brother': kleine broertje.
3. Combine the translations of 'my' and 'little brother' to get the translation of 'my little brother': 'mijn kleine broertje'.
4. Combine the translations of 'I', 'gave', 'my little brother' and 'a ball', to obtain the translation of the entire sentence: 'Ik gaf mijn kleine broertje een bal'.

This compositional translation can be described in a tree structure, as depicted in Figure 3.2, where translation equivalence is made explicit through linking the 'parts' that were used in translation.

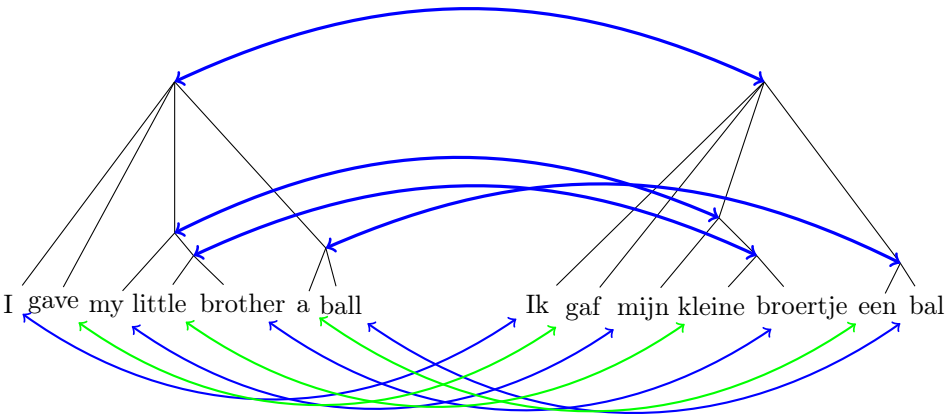


Figure 3.2: A tree description of the compositional translation of 'I give my little brother a ball' into 'Ik geef mijn kleine broertje een bal'

Lexical Divergence

The complexity of the previous example was far below average. The target sentence is a word-for-word translation of the source sentence, and the source sentence could thus have been translated according to every possible tree structure (as every contiguous subsequence has a translation equivalent contiguous subsequence in the other sentence). In the case of translational divergence, more complex structures are required. For instance, if I had given my brother a toy car, instead of a ball, this could not have been captured in the same tree, as 'toy car' is phrasally translated into 'speelgoedautootje'. Figure 3.3 shows how trees can account for phrasal translations.

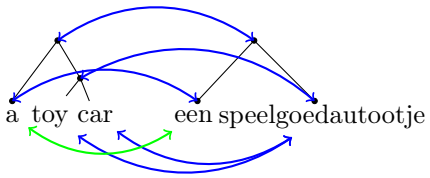


Figure 3.3: A compositional translation tree for a phrasally translated subsequence

Syntactic Divergence

Also syntactic divergence can be captured in translation trees, as is shown in Figure 3.4. The tree describes the translation of 'the girl has a car' into 'у девушки есть автомобиль'.

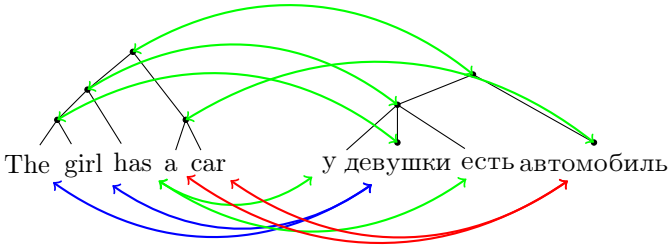


Figure 3.4: Translation of possession, Russian-English

3.2.2 Labelled Trees

All the shown examples of compositional translation trees were skeletons of descriptions, rather than full descriptions, as it was not specified *which* rules were used to compose the sentences and the translation. The skeleton of a

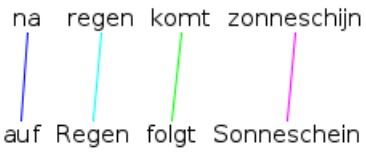


Figure 3.5: A visualisation of a word alignment for the sentence pair ('Auf regen folgt sonneschein', 'Na regen komt zonneschijn'). An arrow from source word w_s to target word w_t implies that w_t was involved in the translation of w_s. This picture was created using the Alignment Visualiser developed by Maillette de Buy Wenniger et al. (2010)

compositional translation tree merely specifies which parts were combined at which stage of translation, but does not give a complete description of the translation. The description can be completed by naming the rules that were used, for instance by labelling the nodes of the tree with their name, and specifying whether the parts stay in the same order or are permuted. Throughout this thesis we will often only consider skeletons of translation trees if knowledge of the rules is unknown or unimportant for what we are investigating.

3.3 Establishing Translational Equivalence

To construct compositional translation trees for a parallel corpus, knowledge of translation equivalence is required. The first step in determining the level of compositionality of a parallel corpus, is thus to establish for every sentence which subsequences have a translation equivalent in the other sentence, and thus could have been a part in a compositional translation. In this thesis, as is common practice in MT, the notion of translation equivalence is based on word-alignments: mappings from source to target words that describe which target words were involved in the translation of which source words. We will discuss word-alignments, and how they can be derived, in this section. Let us start by giving a definition:

Definition 3 (Word-alignment)**.**
Given a source sentence $s = s_0 \cdots s_n$ and its translation $t = t_0 \cdots t_m$, an alignment is a set $a \subseteq \{0, 1, \ldots, n\} \times \{0, 1, \ldots, m\}$ such that $(x, y) \in a$ iff s_x is translated into t_y.[4]

[4]In some definitions unaligned words are explicitly included in the alignment by adding an extra $NULL$ token to both source and target sets and including $(x, NULL)$ (or $((NULL, y))$ in a whenever word x (or y) is unaligned. In our definition, unaligned words are not explicitly included: when a word x is unaligned, this will be indicated by the absence of a word y such that, respectively, (x, y) or (y, x), depending on whether x was a source or target word.

one-to-one	$\forall x \forall y \big((x,y) \in y \rightarrow \forall z \big((z,y) \in a \rightarrow z = x \wedge (x,z) \in a \rightarrow z = y \big) \big)$
one-to many	$\forall x \forall y \big((x,y) \in y \rightarrow \forall z \big((z,y) \in a \rightarrow z = x \big) \big)$
many-to-one	$\forall x \forall y \big((x,y) \in y \rightarrow \forall z \big((x,z) \in a \rightarrow z = y \big) \big)$
many-to-many	-
monotone	$\forall w \forall x \forall y \forall z \big(((x,y) \in a \wedge (w,z) \in a \wedge x < w) \rightarrow y < z \big)$

Table 3.1: Different types of alignments and their logical characterization.

Word alignments give rise to a straight-forward notion of translation equivalence: two sequences are translation equivalent if the words in the one sequence are translated only into words in the other sequence, and vice versa (both sequences are then called 'translation admissible').

3.3.1 Types of Word-Alignments

There are several types of word-alignments, that describe different types of phenomena in translation. A summary of the restrictions corresponding to different kinds of alignments can be found in table 3.1. The example alignment depicted in Figure 3.5 is a very simple one. The alignment is one-to-one, as every word in both sentences is aligned to exactly one other word, and it is monotone, which means the order of source and target words is identical. A monotone one-to-one alignment indicates that no lexical or syntactical divergence complicated the translation.

In case of idiomatic translations, a one-to-one alignment does not suffice, as multiple source words are translated into multiple target words all at once. When aligning the English version of the saying in Figure 3.5 - 'Every cloud has a silver lining' - to the Dutch sentence, it is not clear what should be aligned to what, if every word can have only one counterpart. Arguably, 'has' should be aligned with 'komt', as they are the only verbs in the sentence pair. However, when asking a bilingual Dutch and English speaker if 'has' is a proper translation of 'komt', the odds of obtaining an affirmative answer would be very slim. In this case, a more plausible alignment would align every Dutch word to every English word, indicating that the expression is translated as a whole. Such an alignment is called a phrasal alignment.

A similar adaptation is necessary when syntactic divergence occurs, although in such cases it is often unclear what the best alignment is. Consider, for instance, the word 'does' in the sentence pair ('John does not live here', 'John wohnt hier nicht'). As 'does' has no clear translation in German, one might argue that it should be unaligned. However, the word seems to be connected with 'live', so it could also be aligned with 'wohnt'. A third option is to align 'does' to 'nicht', as it appeared with 'not' when the sentence was negated (example from Koehn, 2008, p.114).

3.3.2 Obtaining Word Alignments

There are a few manually aligned corpora, but given the labour intensiveness of manually aligning corpora, they are mainly used to evaluate automatic aligners. We will briefly discuss manual alignments in 3.3.2 after we have described how to generate alignments automatically.

Automatic Word-alignments

The reader may recall that word-alignments are established as a by-product of the word-based IBM models. Despite multiple efforts to improve on these techniques (see Koehn, 2008, p.119-122 for some examples), it is still common practice to use the alignments produced by the IBM tool GIZA++ (Koehn et al., 2007). In the following paragraphs we will briefly explain how the IBM word-alignments are derived, and how many-to-many alignments can be constructed from them.

Expectation Maximization One step in the IBM models, is to learn a lexical translation model from a parallel corpus. This would be an easy task if word-alignments were directly visible from the data, as one could just count for each word how often it occurred in the text and how it was translated, and estimate a probability from these counts using relative frequency estimation. Conversely, the most probable word-alignments could be estimated if the lexical probability model was known. Learning word-alignments and lexical probabilities from a parallel corpus can thus be seen as a problem of incomplete data, that can be addressed with the expectation maximization (EM) algorithm (Dempster et al., 1977), which works as follows:

1. Initialize the lexical probabilities (often with uniform distributions).
2. Compute the most probable alignment from the lexical probabilities (expectation).
3. Recompute the lexical probabilities from the alignment found in the previous step (maximization)
4. Iterate steps 2 and 3 until convergence.

Note that the use of such an algorithm means that the larger the corpus is, the better the resulting alignments become. It is thus not possible to align just one sentence.

In IBM model 1 and 2, the models that describe how the alignments depend on the lexical probabilities are sufficiently simple to exhaustively run the EM algorithm, and a (global) optimum is thus guaranteed to be found. To find the most probable alignments in the higher IBM models, stochastic hill climbing is used. Excellent examples of complete executions of the algorithm for the different IBM models on very small toy corpora can be found in (Koehn, 2008, p88-113).

Obtaining many-to-many alignments The IBM models consider the sequence of target words as being generated by the source words one by one. Therefore, although source words can be aligned to more target words, every target word is aligned to at most one source word and the resulting alignments are thus many-to-one. However, many-to-many alignments are often desired, as they match phenomena very common in practice. Alignments used to train SMT models on are often created by running the IBM models in both directions, and combining the resulting alignments. The three most common methods of merging two alignments A_1 and A_2 are:

1. Union: $A_1 \cup A_2$, containing all links from both alignments. The recall of the union will be high (as it contains many links), but as it contains all faulty alignment links from both alignments too, the precision is often quite low.[5]

2. Intersection: $A_1 \cap A_2$, containing only the links that occur in both alignments. The resulting alignment is thus a one-to-one alignment. Contrary to the union, the intersection of A_1 and A_2 generally has a high precision, but a lower recall.

3. A more sophisticated method for combining alignments A_1 and A_2, is to first take the intersection, ending up with a selection of reliable alignment points, and then extend the alignment by adding neighbouring links and links (i, j) for which holds that neither e_i nor e_j was aligned in the intersection (Och and Ney, 2000), a heuristic. Pseudocode of this heuristic, called 'grow-diag-final', can be found in Koehn (2008).

Most current MT models that make use of alignments use the grow-diag-final method to obtain their alignments. The resulting alignments have a relatively high precision and recall (Och and Ney, 2000), although they still contain several faulty alignment links. As mentioned before, several attempts to develop improved alignment models have been reported, where very different approaches have been explored. None of these methods has really found its way in the MT community. Besides the fact that it is hard to compare alignment methods across domains, it has been shown that improving alignment models does not necessarily result in better MT models (Indurkhya and Damerau, 2010)).

Manual Word-alignments

As mentioned before, there are also a few manually aligned corpora. To address the issues raised in 3.3.1, manual alignments often distinguish between the

[5] Given a set of desired alignment points A_{gold}, recall and precision of an alignment A are defined as follows:

$$\text{Recall} = \frac{|A \cap A_{gold}|}{|A_{gold}|} \qquad \text{Precision} = \frac{|A \cap A_{gold}|}{|A|}$$

Given the nature of A_{gold}, precision and recall are not the common metrics used to evaluate word alignments.

alignment links that are sure, and those that are possible (Lambert et al., 2005). The sure alignment links (indicated by the letter S) then represent unambiguous alignment links, while the possible alignment links (P) are less certain. Possible alignment links appear in case of phrasal translations, ambiguous translations, or in case two annotators disagree.

All existing manually aligned corpora - the only ones known to the author of this thesis are presented in Och and Ney (2000), Graca et al. (2008), Mihalcea and Pedersen (2003), Padó and Lapata (2006), and Ahrenberg et al. (2000) - are too small to train serious MT models on, and are mostly used to evaluate new alignment techniques (or sometimes for empirical analysis). A common metric used for this task is the alignment error rate (AER), which is defined as follows:

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

A perfect score can thus be achieved by an alignment that has all the sure alignment points and some of the possible alignment points.

3.3.3 Translation Equivalence through Word-alignments

Translation equivalence can be defined in terms of word-alignments, as is described in Definition 4.

Definition 4 (Translation Equivalence)**.**
If (s, t) is a pair of source and target sentences and A an alignment between s and t, two sets of source and target words w_s and w_t are translation equivalent if and only if

$$\forall x, y(x \in w_s \ (x, y) \in A \rightarrow y \in w_t \wedge x, y \in w_t \ (y, x) \in A \rightarrow x \in A))$$

The definition expresses the intuition that two sequences of words are translation equivalent if the words in the first sequence are translated only into words in the second sequence, and vice versa. This definition does not include a clause that states that such sequences need to be contiguous, a requirement that is often imposed when the translation parts are represented by nonterminal nodes in an SCFG.

3.4 Compositional Translation Structures

Knowledge about translation equivalence of a translation pair restricts the set of structures according to which the one could have been translated compositionally into the other, as sequences that do not have a translation equivalent could not have been a part during the translation. In this section, we will define the set of structures according to which a sentence could have been compositionally translated, given its translation equivalent parts. As such trees are thus solely

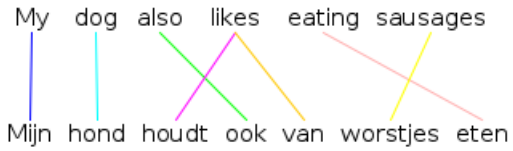


Figure 3.6: A one-to-many alignment of the English sentence 'My dog also likes eating sausages.' and its translation 'Mijn hond houdt ook van worstjes eten'. (tool used to create picture: Maillette de Buy Wenniger et al., 2010)

based on the alignment of the sentence, we will refer to such structures with the term 'alignment trees'.[6]

3.4.1 Alignment Trees

For the purpose of devising alignment trees, the current definition of alignment is not particularly transparent. We will introduce, as far as this thesis goes, a change in notation for word-alignments, that is more suitable and clear for this purpose. The following two paragraphs will thus be a short recap of alignments and translation equivalence, but with a new notation. Figure 3.6 is referred to as example dependency parse to clarify our new definitions.

Set-permutations

Following Sima'an and Maillette de Buy Wenniger (2013), we will represent an alignment by an ordered sequence of sets, of which the nth element specifies to which target word the nth source word maps. In case of a one-to-one alignment, this sequence is thus a permutation of the target word-positions, while in more complex alignments some of the target positions may appear multiple times in the sequence, or not at all. We will refer to such a sequence describing an alignment with the term 'set-permutation', which is defined as follows:

Definition 5 (Set-permutation)**.**
Given a source sentence $s = s_0 \cdots s_n$, its translation $t = t_0 \cdots t_m$, and an alignment a, let $a(i) = \{j \mid (i,j) \in a\}$ be the set with target positions that is linked to source position i. The set-permutation π uniquely describing a is defined as the ordered sequence of sets $\langle a(0), \ldots, a(n) \rangle$.

The set-permutation $\pi = \langle \pi_0, ..., \pi_n \rangle$ describing the alignment depicted in Figure 3.6 would thus be $\langle \{0\}, \{1\}, \{3\}, \{2,4\}, \{6\}, \{5\} \rangle$.

[6] This term is not new in this thesis, but is used by many authors in MT when referring to the hierarchical structures alignments give rise to.

Translation Equivalence for Set-permutations

With the altered definition for alignments, a new definition for translation equivalence is required. In the following definitions, we will assume that $\bigcup_{i=0}^{n} \pi_i$ constitutes a contiguous sequence of numbers (thus there are no unaligned words). Although this is not always the case in practice, such a situation can always be achieved by only numbering the aligned target positions that are aligned, effectively switching the position numbers to the left whenever an unaligned word is found, and can thus be assumed without loss of generality.

Definition 6 (Translation Admissibility)**.**
Let $\pi = \pi_1 \dots \pi_n$ be a set-permutation describing a sentence pair (s, t), a subset $\{\pi_i, \dots, \pi_j\}$ is translation admissible if and only if for every integer $x \in (\pi_i \cup \dots \cup \pi_j)$ holds that $x \notin (\pi_0 \cup \dots \cup \pi_{j-1} \cup \pi_{i+1} \cup \dots \cup \pi_n)$.

The notion of translation admissibility does not require contiguousness on either source or target side, which is desirable, as words are not necessarily translated into adjacent words in another language (for instance, in the running example 'likes' translates into 'houdt ... van'). A projective tree representation of a compositional translation, however, demands that the non-terminal nodes cover contiguous subsequences in both source and target language, which leads to the following definition of non terminal translation part:

Definition 7 (Non Terminal Translation Part)**.**
Let $\pi = \pi_1 \dots \pi_n$ be a set-permutation describing a sentence pair (s, t), a subset $\{\pi_i, \dots, \pi_j\}$ is translation admissible if and only if $(\pi_i \cup \dots \cup \pi_j)$ constitutes a contiguous range of integers (contiguousness on the target side), is a contiguous sequence in π (contiguousness on the source side) and is translation admissible according to Definition 6.

The alignment depicted in Figure 3.6 thus has 5 non-terminal translation parts of length 1, 3 of length 2, 1 of length 3, 2 of length 4, 1 of length 5, and 1 of length 6 (see Figure 3.7).

The number of translation parts in an alignment depends on the type of the alignment and is largest in case of a monotone alignment, that does not restrict the set of possible translation units at all. A completely monotone alignment of a sentence of n words has $\frac{n \times n+1}{2}$ translation units. Note that unaligned words can cause exponential growth in the number of translation units.

Set-permutation Trees

The set of alignment trees follows relatively straight-forwardly from the notion of parts: an alignment tree is a tree whose root dominates the entire sentence, and all of whose nodes correspond with translation equivalent units. To account for non-contiguous parts, nodes are allowed to expand in a combination of terminal and non-terminal nodes as well, provided that the terminals together constitute a translation admissible subset of the total sentence. An alignment tree will be

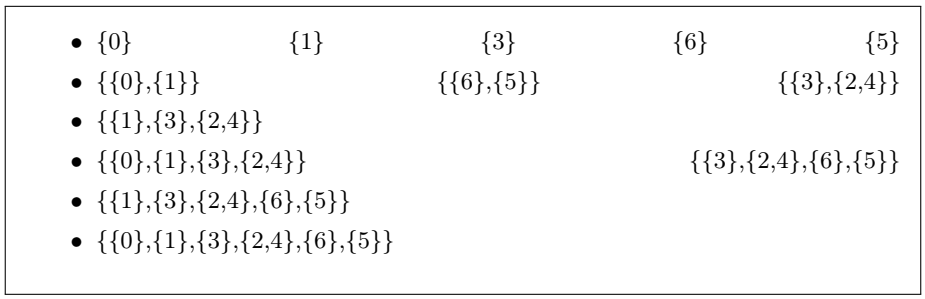


Figure 3.7: The translation units of the alignment for the sentence pair (My dog also likes eating sausages, Mijn hond houdt ook van worstjes eten), depicted in Figure 3.6.

defined in terms of allowed node expansions. If the top node of a tree covers the entire sentence, and all nodes are expanded as described in Definition 8, it recursively follows that all non-terminal nodes are translation parts and all leaf nodes together constitute a translation admissible subset, and the tree is thus a compositional translation tree for the corresponding sentence.

Definition 8 (Expansion of a translation part)**.**
Let $\pi = \langle \pi_0, \ldots, \pi_n \rangle$ be a set-permutation that constitutes a part of a translation. An allowed expansion of π into parts is described by a segmentation of π as an ordered set of indices $B = \{j_0 = 0, j_1, \ldots, j_{m-1}, j_m = n+1\}$ that segments π into m adjacent, non-overlapping and contiguous segments such that for all $0 \leq i < m$ holds that the subsequence $\pi_{j_i} \ldots \pi_{j_{i+1}-1}$ is a new non-terminal node that is a translation part, or $\pi_{j_i} \ldots \pi_{j_{i+1}-1}$ consists of a single word constituting a translation admissible unit with one or more other segments of π.

Using Definition 8, a translation tree can be constructed top down, by recursively segmenting the entire sequence until only sequences of length 1 are left. Figure 3.8 shows a possible alignment tree for the alignment from the running example: the sentence is firstly split into the two translation parts 'my dog' and 'also likes eating sausage', the former is then further split into two 'my' and 'dog', that also both constitute allowed translation parts. Also the right tree is further split up into translation parts, until the segmentation of 'also likes' is reached, and no further division into allowed translation parts is possible. The phrase is therefore segmented into the non-terminal node covering part 'also', and the leafnode covering the translation admissible word 'likes'. Depending on its type, an alignment may have many different possible alignment trees. The current alignment has more than 40 different trees. The number of alignment trees can be exponential in the length of the sentence, if no restriction is placed on the branching factor of the nodes. Every alignment can be assigned at least one structure (the completely flat one).

Note that the trees we have described are describing a compositional trans-

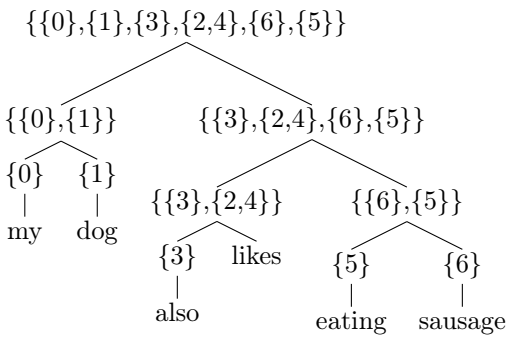


Figure 3.8: A possible alignment tree for the alignment depicted in Figure 3.6.

lation of the source side. The target side tree, however, can be constructed from this tree if it is known how the children of each node were reordered (this information is encoded in the set-permutation).

3.5 Empirically Studying Compositionality

We have showed that, given a word-alignment, it is always possible to construct a set of structures according to which a sentence could have been compositionally translated. Constructing such a set of structures, however, is not yet evidence for compositionality of translation. We do not know if the sentence was in fact translated as described by one of these structures, and whether this could have been predicted by a grammar. Investigating the set of alignment structures of sentences can be used to answer questions about the hypothetical underlying grammar that generated the data. Previously addressed questions can be roughly divided into two categories (we will later propose another perspective to these two categories):

1. **Formal questions**. Several studies focus on the formal properties of the underlying grammar, asking questions as: 'what is the minimal rank of the underlying grammar', or 'how much of the corpus can be covered by a pure permutation'.

2. **Linguistic questions**. Other studies follow a different direction, starting out from monolingual syntax. In these studies, the focus lies on the explanatory power of monolingual syntax, and questions addressed include: 'how many of the nodes of monolingual parse trees are translation equivalent according to an alignment', 'if we stay true to monolingual parses, how far from bijective is the mapping between the non-terminal nodes'.

In this section, we will discuss some previously conducted studies, and the answers they have found to these questions.

3.5.1 Linguistic Questions

Several studies focus on the explanatory power of transforming linguistic parse trees, hereby addressing the coherence between monolingual syntax and alignment trees. Even though they are run on different datasets, with different language pairs and use different criteria, they all find that linguistic parse trees do not coincide very well with translation corpora.

Constituency grammars

There are different ways of quantifying the suitability of monolingual linguistic parse trees. An often cited study is the one carried out in Fox (2002). Fox investigated how well linguistic phrases (i.e., constituents in a parse tree) stay preserved during translation from English to French. For her investigation, she used a manually aligned corpus created by Och and Ney (2000), which contains 500 randomly selected sentences from the Canadian Hansard corpus. The manual alignments in this corpus are of type 'sure' (S) and 'possible' (P). Fox counted the number of times the translation of distinct syntactic constituents (on the English side) overlapped or 'crossed'. We will not give a formal definition of a 'crossing', but provide an example in Figure 3.9. Fox concluded that crossings - even after filtering out phrasal translations that necessarily result in crossings - are too prevalent to ignore (on average 2.854 per sentence if all alignment links are considered).[7]

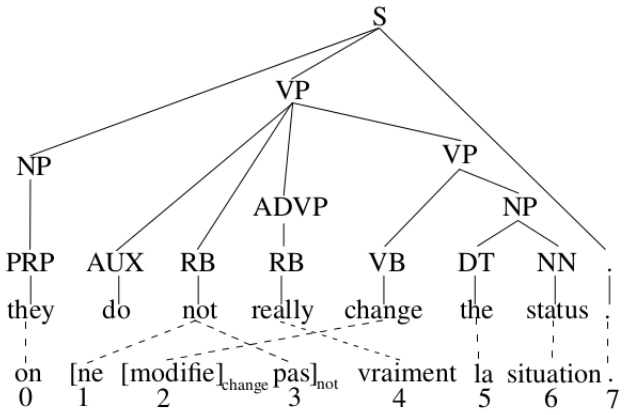


Figure 3.9: An example of a crossing according to Fox (2002).

Her results are supported by others. Galley et al. (2004) generated con-

[7] With a manual analysis of the crossings in the constituency parses she showed that many of them are not due to the lack of phrasal cohesion, but are caused by errors in the syntactic analysis or rewording and reordering in the translation. Her analysis, however, included only the crossings of the S alignment links - the ones on which all annotators agreed and that were not ambiguous - that constitute just a small part of the total set of crossings.

stituency trees on the English side of the aforementioned Hansard corpus and tested how powerful a synchronous grammar should be to be consistent with the translation corpus. The power of the grammar was expressed in terms of the depth of the subtrees it generated, a standard CFG rule, that generates only single non-terminals, thus has depth 1. If rules of larger depth than 1 are needed, the non-terminal nodes of the syntactic source tree cannot be mapped bijectively to any target tree in a way consistent with the word-alignments. Galley et al. found that only 19.4% of the trees in the corpus could be covered entirely by one-depth-rules, and 85% of the nodes (for the S alignments). Furthermore, he found that to cover the entire corpus with a grammar consistent with the allowed number of rule expansions should be no less than 17 for the S-alignments, and 23 for automatic alignments. For the English-Chinese corpus he analysed, the coverage of low-expansion rules was even lower: 16.5% (of the trees) for rules with a single expansion, and 100% only with a maximum of 43% expansions per rule.

Khalilov and Sima'an (2012) confirmed the inadequacy of child-reordering in work that focusses on source reordering preliminary to translation. Using LRscore (Birch and Osborne, 2010) as a measure of success, they concluded that permuting the children of nodes in a constituency tree is insufficient to reach a perfect permutation of source-words in English-Dutch and English-Spanish translation data, even when deleting up to 5 layers of nodes in the parse tree is allowed.[8]

Dependency Grammars

Not very much literature focusses on the consistency of dependency grammars with translation data, but some articles can be found on the matter. In her study about crossings, Fox also devoted a section on dependency grammars. She observed that dependency parses are more cohesive than constituency grammars, with 2.714 crossings per sentence, compared to 2.854 for constituency grammars. A study that focusses exclusively on dependency parses was presented in Hwa et al. (2002). She investigated how well predicate argument structures agree between English and Chinese, addressing the validity of the Direct Correspondence Assumption.[9] Hwa et al. evaluated the quality of Chinese dependency parses that were projected directly from English to Chinese through a manual word-alignment. The resulting parses have a very low F-score (38.1), which is not surprising, as phrasal translations (multiple aligned words on source or target side) and unaligned target words always result in errors. Hwa et al. also observed this fact. They developed a small set of linguistically motivated rules, which boosted the F-score to 68.3, which is significantly higher, but still rather low. Also, it makes their work very specific, and hard to extend to other language pairs or contexts.

[8] Their score for English-Spanish, however, is surprisingly high: around 94.

[9] Which expresses the intuition that there exists a mapping between the syntactic relationships in two sentences that are each others translation, and is thus directly allied to compositional translation

Another work along the same lines was presented by Fung et al. (2006). Fung et al. did not directly use dependency grammars, but learned cross-linguistic (English-Chinese) semantic verb frames. The learned argument mappings had an accuracy of 89.3%. It is unclear how their results compare to Hwa et al.'s (2002) results and dependency grammars in general, a fortiori because the exact nature of the learned semantic frames stays unclear.

3.5.2 Formal Questions

A second line of empirical research does not restrict source (or target) side trees to linguistic trees, but investigates the coverage of formal SCFGs. The majority of the empirical results on SCFGs focus on the coverage of binary trees (Zhang et al., 2006; Huang et al., 2009, e.g.,), or SCFGs in normal form (e.g., Søgaard and Kuhn, 2009; Søgaard and Wu, 2009; Søgaard, 2010). All concluded that the range of reordering phenomena occurring in real translation data are by far not as complicated as the worst case scenario sketched in Satta and Peserico (2005).

Wellington et al. (2006) seem to be the only onse who compared their results with linguistically restricted parse trees. On several dataset (covering translation from Chinese, Romanian, Hindi, Spanish and French to English), they found that maximally 5% of the alignments could not be explained by a completely binary tree, while the failure rate for binary trees that were constrained by monolingual parse trees on the English side climbed to 15% for French/English to 61% for Chinese/English. The failure rate they found for non constrained binary trees is much lower than the one found by Sima'an and Maillette de Buy Wenniger (2013), who reported a coverage of 71.46% for the manual alignments of the Hansard corpus for trees with a maximal branching factor of 2. The coverage of binary trees for automatic alignments was even lower: 52.84%. This difference between the results of Wellington et al. (2006) and Sima'an and Maillette de Buy Wenniger (2013) is most likely due to a different treatment of alignment links: the latter authors used all alignment links in the dataset, while the former treated many-to-one alignment links disjunctively, focussing on lower bounds. Sima'an and Maillette de Buy Wenniger (2013) also reported the coverage of non binarisable (permutation) trees, which is surprisingly enough not much higher: 72.14% and 56.56% for manual and automatic alignments, respectively.

3.6 Dependency Parses

The studies described in the previous section are conducted from two different perspectives, that we earlier described as linguistically oriented and formally oriented. The formally oriented studies take a bilingual perspective, starting from the data and trying to gain information about the system that generated it. The linguistically oriented studies start out from monolingual information about language, and try to assess its suitability as underlying system. Surprisingly enough, a study that combines the two perspectives, by trying to construct a bilingual grammar initially motivated by the data, but using monolingual

information, does not really exist. In this thesis, we will try to narrow the gap between the two perspectives, by performing a study in which we combine bilingual information from a corpus and monolingual information from dependency parses. In this section, we will provide information about the background of dependency parses, and describe them formally. This section stands apart from the rest of this chapter, as it is not directly related to translation, but presents the last piece of background of the investigation conducted in this thesis.

3.6.1 Background

The dependency grammar is a cognitively motivated grammar formalism, that describes the perception of a sentence by the brain. Given its cognitive aim, dependency grammars are largely semantically motivated. Contrary to phrase structure grammars, that establish relations between constituents of a sentence, a dependency grammar does not divide a sentence up in phrases. Rather, it is based on the idea that in a sentence all words but one depend on another word in the sentence, via a(n asymmetric) binary relationship, that describes how the former word modifies or complements the latter. For instance, in the sentence 'I really like writing my thesis', 'my' depends on 'thesis', as it complements it, and 'really' depends on 'like', which it modifies. Words can be said to have a valency, depending on how many dependents they need to be saturated (e.g., 'like' would have a valency of two 2, as it needs both a subject and an object). The semantic background, combined with the fact that dependency grammars earlier showed to behave more coherent during translation than constituency grammars (Fox, 2002), motivate a further investigation of this formalism.

Although traditional dependency grammar (DG) has been used by linguists since the Middle Ages (Covington, 1990), modern DG is often seen as being created by Tesnière and Fourquet (1959), whose cognitive motivation for it is worth citing:

> The sentence is an organised whole; its constituent parts are the words. Every word that functions as part of a sentence is no longer isolated as in the dictionary: the mind perceives connections between the word and its neighbours; the totality of these connections forms the scaffolding of the sentence. The structural connections establish relations of dependency among the words. Each such connection in principle links a superior term and an inferior term. The superior term receives the name governor; the inferior term receives the name dependent. (Translation: Nefdt, 2013)

The criteria for being a head-dependent pair are a mix of syntactic and semantic criteria (Nivre, 2005), and generally depend on the grammatical function the sentence or with respect to the word it depends on. Not all dependency grammars are identical in the relations they are considering, and their treatment of certain intuitively problematic constructions as coordination and conjunction (Nivre, 2005). In this thesis, we will follow the convention used in De Marneffe et al. (2006).

3.6.2 Formally

A dependency grammar is formally defined as follows (Hays, 1964; Gaifman, 1965):

Definition 9 (Dependency Grammar)**.**
A dependency grammar is a quadruple $\langle R, L, C, F\rangle$, consisting of a set L of terminal symbols (lexemes), a set C of auxiliary symbols (lexical categories), a set R of dependency rules over the auxiliary symbols C, and an assignment function $F : L \to C$.

An example of a very simple (and unplausible) dependency grammar (according to which the dependency parse in Figure 3.11 is grammatical) is shown in Figure 3.10.

$\mathcal{D} = \langle R, L, C, F\rangle$, where:

$L = \{$My, dog, also, likes, eating, sausage$\}$

$C = \{$poss, nsubj, xvmod, xcomp, dobj, root$\}$

$R = \{$(root, nsubj), (nsubj, poss), (root, xcomp), (xcomp, dobj), (root, xvmod)$\}$

$$F = \begin{cases} F(\text{My})=\text{poss} & F(\text{dog})=\text{nsubj} & F(\text{also})=\text{xvmod} \\ F(\text{likes})=\text{root} & F(\text{eating})=\text{xcomp} & F(\text{sausage})=\text{dobj} \end{cases}$$

Figure 3.10: A Toy Dependency Grammar

A dependency grammar prescribes dependency structures of sentences, that can be interpreted as graphs that satisfies the criteria that it is rooted and has only a single head (in other words, a dependency structure is a tree):

Definition 10 (Dependency Graph)**.**
A dependency graph of a sentence $s = w_1 \ \ldots \ w_n$ is a directed acyclic graph $\mathcal{G} = \langle V, E\rangle$, in which $V = \{w_1, \ldots, w_n\}$ and E is a set of edges such that $(w_i, w_j) \in E$ if and only if there is a dependency relation between w_i and w_j. Furthermore, the graph satisfies the following two criteria:

1. $\exists w \in V$ *s.t.* $\forall w' \in V \ (w, w') \notin E$ *(rootedness)*
2. $\forall w_1 w_2 w_3 \in V \Big((w_1, w_3) \in E \wedge (w_2, w_3) \in E \Big) \rightarrow w_1 = w_2$ *(single-headedness)*

The edges in $\mathcal{G}$ can be labelled with the function of the dependent.

An example of a dependency graph is depicted in Figure 3.11. As the general definition of a graph is used, it is not immediately clear how Definition 10 and

Definition 9 relate. To clarify: the set of vertices V in $\mathcal{G}$ correspond to the lexical items, and thus the set L in a dependency grammar. The edges V correspond to the dependency relations R, while their labels must be in C. The function F describes which functions a word is allowed to have in a sentence.

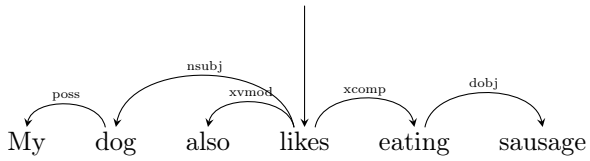


Figure 3.11: A dependency graph of the previously considered sentence 'My dog also likes eating sausage'.

A condition often put on dependency graphs, is projectivity, that prescribes a linear ordering of the nodes in the tree. Projectivity simplifies parsing, as it reduces the search space, but it is often argued that it deprives dependency grammar from its most important asset (Covington, 1990; Debusmann, 2000): the elegant method for handling discontinuous constituents (see Figure 3.12). For fixed word order languages like English, in which phrases belonging together tend to stay together, projectivity is thus a reasonable criterion, but to account for languages in which there are less restrictions on the word-order (e.g., Russian, Latin) non-projectivity is often required to provide an intuitive analysis. An example of a non-projective in an otherwise fixed word-order language (Dutch) is depicted in Figure 3.12.

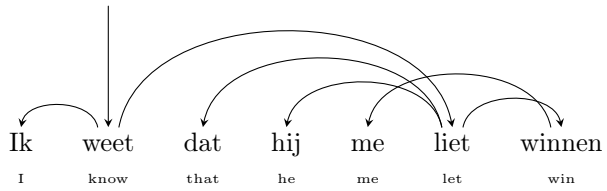


Figure 3.12: A non projective dependency graph of the Dutch sentence 'Ik weet dat hij me liet winnen'.

3.7 Summary

In this chapter, we have discussed the difficulties of designing a system to compositionally translate natural languages. We have showed that there are several issues, and that it is hard to investigate their influence on the suitability of compositional translation as a strategy for translating natural language. We have argued that finding compositional solutions for specific parts of natural

language for which a compositional treatment seems problematic will not provide us with a better understanding of the overall compositionality of translation, and neither will developing more translation models. We proposed to investigate the compositionality of translation through empirical analysis of real translation data, and laid the groundwork for such an analysis, by discussing how the compositionality of a corpus can be identified and investigated. We have discussed empirical studies related to the one conducted in this thesis, and we have pointed out that there seems to be a gap between the two perspectives they are taking. In the last section of the chapter we discussed dependency parses, that will be important for our own empirical analysis.

Chapter 4

An Empirical Account of Compositionality

In the previous chapters, we sketched the background for this thesis, introduced the translation strategy we are investigating, and argued that further research to this strategy is necessary. In the rest of this thesis, we will discuss the research that we have conducted that contributes to this topic. The remainder of this thesis is divided in three parts. In the current chapter, we will discuss the foundations of our study. We will formulate our research questions, discuss the assumptions that we make in investigating them, and discuss the design choices that are made regarding the basic ingredients of our experiments. In the subsequent chapter (Chapter 5) the empirical answers that we have found regarding the correspondence between dependency parses and alignment trees will be presented, as well as details about the experiments designed to obtain these results. Finally, in Chapter 6, we will provide a more general discussion of our work, propose an approach to overcome the difficulties that we have found in the experiments, and make suggestions for future work.

We will start this chapter by clearly stating what the main goals of this thesis are. This will include a recap of the motivation for this research. We will formulate the research questions that will be addressed, and of what nature the answers will be (Section 4.1). This section will end with a short remark on compositionality, and its empirical interpretation. In the subsequent section (Section 4.2), we will present the general set-up, and inform the reader about the proceedings of the rest of the chapter, in which the different ingredients of our research are discussed in more detail (Section 4.3 - 4.4). Once again, the Chapter finishes with a brief summary.

4.1 Summary and Objectives

This thesis addresses the appropriateness of compositional translation as a strategy for translation between two natural languages. Compositional translation is

a well examined topic in MT, where it is explored in the form of SCFGs, and its suitability is not likely to be confirmed or refuted in one thesis. Therefore, this thesis focusses on a subquestion, that relates to the role that monolingual syntax can play in creating a compositional grammar. It seems reasonable to assume that incorporating monolingual syntactic information about the source and/or target language could improve SCFG models, as the two sides of an SCFG describe the structures of the source and target side languages. However, in practice it proved to be very hard to actually exploit this information. In this thesis, we will examine this problem on an empirical level.

We are not the first to empirically assess the usefulness of monolingual information for MT models. Previous empirical studies have mainly focussed on the consistency between constituency grammars and translation data, and it was generally concluded that the bilingual coherence of phrases prescribed by such grammars was too low to be directly exploitable. Although unfortunate, this is not extraordinarily surprising, as constituency grammars are a purely syntactical system of language, while in translation it is but the semantics that should be preserved. We therefore propose that dependency grammars are a more suitable formalism to use for MT, as they are merely semantically motivated.

There are studies that have explored the usefulness of dependency grammars for MT, but there are very few. Hwa et al. (2002) investigated whether dependency grammars can be projected from English to Chinese through word alignments, but the quality of directly projected parses was very poor. The study did not account for phrasal translations or unaligned words, which made it relatively limited. A second study considering dependency parses was presented by Fox (2002). In a study of which constituency grammars were the main focus, she also investigated how well the phrases suggested by dependency grammars stay preserved during translation. She concluded that dependency parses have better cohesive properties than constituency grammars, but did not investigate the structure given rise to by dependency parses, nor the causes for deviation from them. She used a heuristic to detect contiguous phrasal translations, but did not account for phrasal translations on a more general level.

In this thesis, we will present a more thorough investigation of dependency parses from a bilingual perspective, that does not only assess their cohesive properties, but also focusses on the applicability of dependency parses in MT. We will focus on the following three questions:

1. Are dependency structures universal for languages?
Contrary to earlier studies, we will not just focus on the coherence of phrases prescribed by dependency parses during translation. Rather, we will consider whether the compositional structures dependency parses give rise to are preserved over language, by checking for consistency of individual dependency relations with compositional translation structures of the sentence. Whether dependency relations are preserved over language is not only interesting for MT models, but also from a linguistic point of view, as studying dependency grammars through translation data offers perspective to its universality as a grammar formalism.

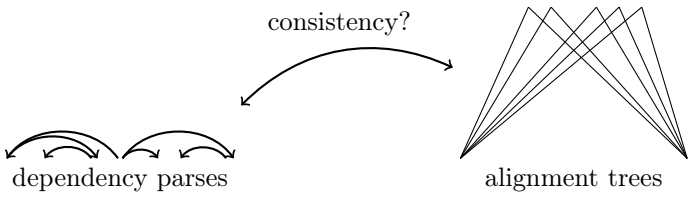


Figure 4.1: Is ther consistency between dependency parses and alignment trees?

2. What are the reasons dependency parses are not entirely preserved during translation?
Given previous studies, it is to be expected that dependency parses will not be entirely preserved during translation, resulting in the follow up question: what are the main bottlenecks? We will investigate what the phenomena are that cause dependency structures to change during translation.

3. Can dependency grammars be used to construct a bilingual compositional grammar?
When the direct preservation of dependency relations is assessed and the main bottlenecks are identified, a third question can be addressed, that touches on the bigger question regarding compositionality: can we use this information to construct a bilingual grammar matching translation data? As we do not develop an MT model, the answer to this last question will be rather speculative because even if a compositional grammar could be constructed, we will not test it in practice. However, we can investigate whether the answers to the previous questions can be used to create a bilingual grammar that can generalise over different sentences and alignments, has good coverage over a corpus of sentences, and is able to assign structures to sentences that are in line with their alignment.

In the next section, it will be discussed how this thesis intends to answer these questions. Before we, get there, we want to make a last remark about compositionality of translation, how the issue should be approached in an empirical study and how answers should be interpreted.

A remark about compositionality
It is often argued that translation cannot be treated compositionally because there are certain phenomena for which it is hard to find a compositional treatment.[1] In this paragraph, we want to anticipate arguments of this kind, by arguing that these phenomena are not necessarily problematic for compositional translation.

[1] A famous example that is often given is the translation of 'he swam across the river' into Spanish, where this is translated as 'he crossed the river swimming', which is hard to give a compositional treatment (Landsbergen et al., 1989)

As pointed out earlier, compositionality of translation is highly underspecified as a principle. It requires that translation of phrases should be able to be derived from the meaning of smaller parts by means of translation equivalent rules, but it is not defined how complex these rules or parts are allowed to be. When constructing a compositional grammar, phrases that cannot be translated compositionally can be included in the grammar, without that grammar formally losing the property of being compositional. We therefore want to argue that the non-compositional phenomena sitting in the corners of natural language are by themselves not arguments against the compositionality of language, and are thus not that interesting for the matter in practice. Rather, it is interesting *how many* phenomena are located in this corner, and whether we can practically account for them without specifying directly for (almost) every sentence in the language what its translation is. The question 'can every part of every sentence of natural language be translated compositionally' is thus not a sensible question, as the answer is: of course not. It is always possible to find odd constructions or idiomatic expressions that are hard to systematically map to another language. The important questions are: can we identify these non compositional parts in a corpus and include them in a grammar without losing the overall feeling that the grammar is compositional, and can such a grammar cover a reasonable part of natural language. Exactly these questions will be addressed in this thesis.

4.2 Set-up

The questions posed in the previous section will be addressed by studying alignment trees, recursive structures of translations based on word-alignments that can be interpreted as compositional translation structures. In this thesis, we will consider the subset of these structures that is maximally compositional, as defined in Sima'an and Maillette de Buy Wenniger (2013), that are called *hierarchical* alignment trees (HATs). We will investigate the consistency of these structures with dependency parses, investigate the main causes for deviation, and propose a method for using both dependency parses and HATs to learn a compositional grammar.

In this chapter, we will discuss the main ingredients of our study. In Section 4.3, we will discuss the basis of empirical studies of translation data, and the assumptions that have to be made to conduct such research. In Section 4.4, we will revisit compositional translation structures. We will motivate the choice for HATs and discuss how they can be efficiently generated and represented. In Section 4.5, we will briefly come back to dependency parses and provide some details of how they will be considered.

4.3 Foundations of Empirical Studies

Empirical analyses of compositionality are based on real data, that are not always perfect. When training MT models, infrequent mistakes in the data are generally

not problematic, as they will receive a low probability. The same cannot be said for empirical analyses, where mistakes in the data will almost always harm the outcome. It therefore seems sensible to see empirical analysis as providing an upper- or lower bound (depending on the context) rather than an exact number.

To appreciate empirical research, it is important to be aware of the factors that influence the results, and the necessary simplifying assumptions. In this section, we will discus these factors and assumptions, some of which may be obvious, to provide a complete picture of the foundations of empirical studies.

4.3.1 Correctness of the Translation Data

Empirical analyses based on parallel corpora with text that are each others translation rely heavily on the correctness of the data in these these corpora. As these parallel texts were not designed as data for translation models, they might not be perfectly suitable for this purpose. There are three bottlenecks.

Sentence level alignment

Aligning corpora on a sentence level is not as simple as it might seem. Texts are not always translated sentence by sentence. Short sentences may be merged or long ones broken up, and in some languages sentence delimiters do not really exist (Koehn, 2008, p.55). However, the techniques for sentence alignment are very good, and as the languages we are considering *do* have clear sentence delimiters it seems very reasonable to assume that the sentences in the corpora are correctly aligned.

Correctness of translation

The translations of the sentences are produced by humans, who sometimes make mistakes. To use the corpora, we have to assume that the aligned sentences are good translations of each other.

Translation is literal

One English sentence often has many translations in another language, as similar meanings can be expressed in multiple ways.[2] For instance 'jeg giver dig blomster' is a good Danish translation of 'I give you flowers', but so is 'jeg giver blomster til dig' (and this is not even an example in which many things are rephrased). Especially when one text is not a direct translation of the other text, but the two are, for instance, just separate reports of the same event, it might happen that sentences do have the same meaning, but differ in form. In our analysis, we will assume that at least the vast majority of the translations in the corpora are rather literal.

[2] In fact, considering only one target translation can also be seen as a simplification made in empirical research, and in MT in general.

4.3.2 Correctness of Word Alignments

In empirical analyses as well as MT-models, word-alignments are of crucial importance, as they are used to establish translational correspondences. Unfortunately, automatic alignments are not always as good as we want them to be (see Och and Ney, 2000, for concrete numbers). MT-models generally do not suffer much from this fact, because the number of wrong alignment links is dwarfed by the number that is correct. For empirical research, false alignment links are quite problematic, as even one wrong link can have a huge effect on the space of possible translation trees. An option is to use one of the few manually aligned corpora, but given their small size they are not suitable to draw conclusions about larger parts of language.

4.3.3 Correctness of Dependency Parses

To determine the dependency structures of the sentences in the corpus, an efficient fully-automated dependency parser is needed. For English, high quality dependency parses are available (Cer et al., 2010), but the parses they produce are not perfect, which can be problematic for an empirical analysis.

4.4 Translation Structures

HATs constitute a very important part of our thesis, as they represent our interpretation of compositionality. In this section we will motivate the choice for HATs, by explicating their advantageous properties. Furthermore, we will suggest a method for representing and generating them, which is given their huge number not trivially time or space efficient.

4.4.1 HATs

Although we use the term HATs to refer to them, the translation structures that are considered in this paper differ in one aspect from how they were originally defined in (Sima'an and Maillette de Buy Wenniger, 2013). Sima'an and Maillette de Buy Wenniger augmented the nodes of the HATs with operators that describe how their children should be permuted to obtain the corresponding target side HAT. We will not consider these operators in this thesis. Ultimately, the operators are necessary to fully describe the translation, as without them the target side structure of a sentence cannot be retrieved from the source side structure. However, in this thesis we take a step back, by solely considering if the skeletons of the purely bilingually motivated HATs can describe the source language in a monolingual fashion. We will thus not pay any attention to the target side languages, but only consider the structural restrictions the translation into target language sentences places on the structure of the source sentences. If it is possible to find a monolingual system for the source language that corresponds with these bilingual constraints, a next step is of course to take into account the operators and reordering phenomena to consider the target side structures.

4.4.2 Motivation for using HATs

Recall that a HAT is a maximally compositional alignment tree, which means it is characterised by the following set of properties:

1. A HAT is a projective tree whose leaf nodes form a sentence.
2. A HAT describes a compositional translation of the sentence constituted by its leaf nodes.
3. All non-terminal nodes of a HAT dominate a sequence that is translation admissible according to the alignment of the sentence, while sibling terminal nodes together constitute a translation admissible sequence.
4. A node in a HAT can have both non-terminal and terminal child nodes at the same time.
5. All nodes in a HAT expand into a minimal number of children.

A HAT is thus an alignment tree that describes a maximally compositional translation of a sentence.

The property of being maximally compositional gives HATs several attractive properties, which we will discuss in this subsection.

Computational

There is a clear computational advantage to considering only minimally branching trees. Not only does it significantly reduces the space of trees to be considered - the example sentence previously used in Section 3.4 for explaining alignment trees[3] has 44 alignment trees, but only 5 of them are minimally branching - it also simplifies parsing, as the lower-rank rules that can be extracted from minimally branching trees can be more efficiently treated by parsing algorithms.

Theoretical

Of course, such computational considerations are less important for an empirical analysis (although even for empirical analysis computational requirements should match the reality). However, maximal compositionality also has some attractive properties besides the compositional ones, which we will explicate in the following paragraphs.

Compositionality Considering only minimally branching trees secures that the system we are studying is in fact compositional. The set of all alignment trees contains many flat trees, that can strictly speaking be seen as compositional (as compositionality is highly underspecified in this respect), but do not capture the recursive and systematic nature of language. A compostional system containing

[3]Sentence pair: ('My dog also likes eating sausages', 'Mijn hond houdt ook van worstjes eten')
Set-permutation: $\langle\{0\},\{1\},\{3\},\{2,4\},\{6\},\{5\}\rangle$.

a separate rule for almost every sentence that specifies how its meaning can be derived from *all of its words* does certainly not correspond with human intuitions about compositionality, not to mention the fact that such a system should have an infinite number of rules to cover the entire language. Considering only minimally branching trees solves this problem.

Generalisation Considering only expansions that are maximally compositional maximises the chance that generalisation to new data is possible: a rule that specifies how a type of argument can be combined with a type of predicate is more useful than a rule specifying how the argument 'I' can be combined with the predicate 'like'. Minimum depth expansions are more probable to be applicable in new situations.

Figure 4.2 shows how French negation, often mentioned as problematic for structure based systems, is accounted for with a HAT skeleton. The translation tree shows that 'I dont like' is the translation of 'Je n'aime pas', but also contains the information that 'don't' is phrasally translated as 'ne ... pas'. Removing the negation in the English sentence results in the grammatical English sentence 'I like cars', removing its translation equivalent in the French sentence in its (almost) grammatical 'Je aime les voitures'. If 'I don't like' was generated in one rule, this generalisation would not have been possible.

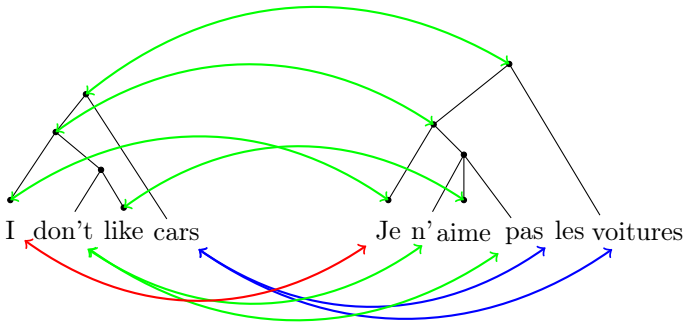


Figure 4.2: Translation of negation, French-English

Preservation of structure of phrasal translations An advantage that overlaps with the two previously mentioned advantages, but is worth noting nonetheless, is the fact that structure of phrasally translated sequences is preserved, if possible. As the rest of the tree, sequences of words will be translated as a phrase only if they do not have a deeper structure according to the translation data. In many phrase-based translation systems, including the successful hierarchical phrase based system proposed by Chiang (2007), the underlying structure of sequences that are translated phrasally gets lost in the process,[4]

[4]This simplification is mostly made for efficiency reasons.

whereby the system misses an opportunity to detect a pattern. HATs fully exploit recursiveness, also in idiomatic and phrasal translations. We will revisit an example containing syntactic divergence to illustrate how a structural treatment of phrasal translation is helpful.

In Russian, 'X has Y' is (somewhat communistically) translated as 'with X is Y', the object in English is thus the subject in Russian. Figure 4.3 shows how this is dealt with in a translation structure. Due to the structural treatment of the phrasal translation, this translation tree is easily extendible to longer sentences with the same construction. By expanding the non-terminal nodes it can also capture sentences like 'the girl with the long blond hair has a very old car with broken windows'.

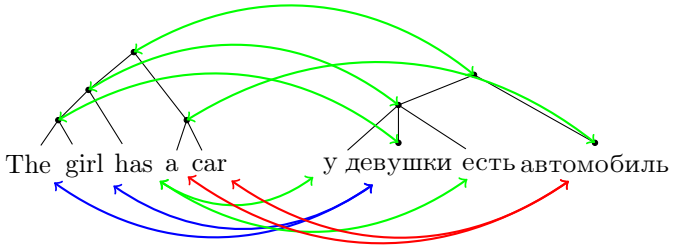


Figure 4.3: A graphical representation of the translation of possession from Russian to English.

Critical Note

Intuitively, maximal compositionality sometimes seems somewhat strict. When constructing a sentence that has an predicate with two arguments, it is linguistically not always desirable to assume that the predicate is combined with the arguments one by one. In translation, this issue is enhanced by the fact that arguments of a predicate may not be in the same order for different languages. A HAT describing a translation in which this happens is forced to combine the arguments with each other before combining them with the predicate, which may sometimes seem counterintuitive.

4.4.3 Representational Aims

To study HATs, we need to be able to generate HATs for every sentence, and store them in a suitable fashion. Generating and storing all trees separately would be both time and space consuming, and would impede a flexible search through them. Hence, a suitable representation of the set of HATs is required. This representation should be easy to search, preferably memory efficient, flexible to abstract information from and easy to combine and compare with other representations of sets of HATs.

(0-6] → (0-1] (1-6]	(4-6] → (4-5] (5-6]	(0-1] → My
(0-6] → (0-2] (2-6]	(0-4] → (0-1] (1-4]	(1-2] → dog
(0-6] → (0-4] (4-6]	(0-4] → (0-2] (2-4]	(2-3] → also
(1-6] → (1-4] (4-6]	(1-4] → (1-2] (2-4]	(4-5] → eating
(1-6] → (1-2] (2-6]	(0-2] → (0-1] (1-2]	(5-6] → sausages
(2-6] → (2-4] (4-6]	(2-4] → (2-3] likes	

Figure 4.4: The grammar generating the translation tree forest for the sentence 'My dog also likes eating sausages', with set-permutation $\langle {}_0\{0\}_1,\ {}_1\{1\}_2,\ {}_2\{3\}_3,\ {}_3\{2,4\}_4,\ {}_4\{6\}_5,\ {}_5\{5\}_6\rangle$ would be (the subscripts indicating the span annotation, which is left exclusive and right inclusive)

The original HAT paper also presented an implementation for generating and storing HATs. The HAT-forests of sentences were represented as a compact chart. Unfortunately, this implementation could not be used for the purpose of this paper. The chart representation was not easily accessible as an object, and modifying the program that generated it to obtain the desired properties did not seem within reach. Furthermore, unaligned words were completely ignored. In some cases, this might be a reasonable simplification, but for empirical analysis this is not desirable.

For this thesis, we aimed to find a new representation of a set of HATs that has both the efficient properties of the current implementation, but is more flexible and accessible, and easy to use for other purposes than precisely the one of this paper. In this subsection, we will explain how we will represent and generate HATs. Documentation of the complete implementation can be found in Appendix **??**. To make the representation easily accessable for people unfamiliar with the code, we have implemented several demo's to illustrate the use and possibilities of the HATs.

4.4.4 Representation

We have represented a set of HATs implicitly by a CFG that describes for every (translation admissible) span of the sentence how it is allowed to expand. By default, the nodes are labelled with the span they dominate, but labels for spans can be provided. Figure 4.4 provides an example.

HATS represented as context free grammars are easy to wield. As the labels are unique, the HAT CFG can be easily unpacked into all HATs for a sentence (which we will never do), but can also recursively be investigated. E.g., the number of HATs can be determined efficiently in a top-down fashion, and parsing with the HATgrammar can provide information about several of its properties.

4.4.5 Generation

To generate the HAT forest for an alignment, it suffices to find the minimal segmentations for all contiguous translation admissible subsequences of the alignment. We generated the set of contiguous translation admissible subsequences (i.e., the source sides of all phrase pairs) with the shift-reduce algorithm presented in Zhang et al. (2008). Some small adaptations were necessary to generate the full set of phrase pairs, rather than only the 'tight' subset, in which unaligned words at the boundaries were not included. To find the minimal segmentations of all phrases we used Dijkstra's (1959) shortest path algorithm (Algorithm 1), that we adapted to search for *all* shortest paths rather than just one, and to incorporate the information that, given the direction of the edges, only searching in nodes with a larger value could actually lead to a shortest path, which made the search more efficient. Details are provided in Algorithm 1. For memory efficiency, all paths were stored in a linked list with path points that were shared among different paths.

Algorithm 1 Shortest Paths

Input: A graph $G = (V, E)$ describing an alignment and two vertices i and j for which $(i, j) \in E$ is true.
Output: All non-trivial shortest paths from i to j
#Initialization
visited = $\emptyset$, depth = 0, paths = $\{j\}$
$\forall n \in \mathbb{N}$: reachable(n) = $\emptyset$; reachable(0) = $\{j\}$
depth_finished = False
 # Start backwards search through graph
while not depth_finished or $i \notin$ visited **do**
  while reachable(depth) $\neq \emptyset$ **do**
    depth_finished $\leftarrow$ False
    current_node $\leftarrow$ N an arbitrary element v from reachable(depth)
    reachable(depth) $\leftarrow$ reachable(depth) $-$ {current_node}
    for (l,current_node) $\in E$ **do**
      if $l \notin$visited $\cup$ reachable(depth) **and** depth $\neq 0$ **then**
        reachable(depth+1) $\leftarrow$ reachable(depth+1) $\cup$ $\{l\}$
        for path (current_node,. . . , j) $\in$ paths **do**
          path $\leftarrow$ (l,current_node,. . . , j)
        end for
      end if
      visited $\leftarrow$ visited $\cup$ $\{l\}$
    end for
    depth_finished $\leftarrow$ True
    depth $\leftarrow$ depth+1
  end while
end while
Return paths

4.5 Dependency parses

As we have elaborately discussed dependency parses earlier in Section 3.6, we will only briefly revisit them in this section, and discuss how we obtain them.

Several assumptions can be made about dependency parses. In this thesis it is assumed, following common practice, that a dependency structure satisfies the following two conditions:

1. When seen as a relation, D constitutes a single-headed acyclic connected graph in which the words in s are the nodes. (tree-constraint)
2. When the words are placed in the original order, the branches of the dependency tree do not cross. (projectivity)

4.5.1 Representation

We will represent a dependency parse of a sentence $s = w_0 \cdots w_n$ as a set of relations $D = \{(i, j)|$ there is a dependency arrow from word w_i to word $w_j\}$.

Formally, dependency grammars are not interpreted as compositional grammars, as they do not postulate the existence of non-terminal syntactic categories. That is, dependency parses describe predicate-argument relations between *words* and not between larger parts, and do thus not explicitly specify the recursive structure of a sentence. However, dependency graphs do give rise to a hierarchical structure, that specifies from which smaller parts the sentence was composed. For instance, the dependency graph depicted in Figure 4.5 tells us that 'likes' is the head word of the sentence, and that the sentence is composed of 4 parts: the head 'likes', its modifier 'also', its noun subject whose head is 'dog' and the open clausal complement whose head is 'eating'. The complement and subject are further divisible in 'My' and 'dog', and 'eating' and 'sausage', respectively. As the tree is projective, all parts are continuous. Also the graph in Figure 3.11 we saw earlier prescribes an hierarchical structure: it is composed of the subject 'I', the headword 'know', and the phrase headed by 'liet', that is in its turn built up from its head 'liet', 'dat', 'hij' and the discontinous phrase 'me winnen'. Such an hierarchical structure cannot be captured by a phrase structure grammar.

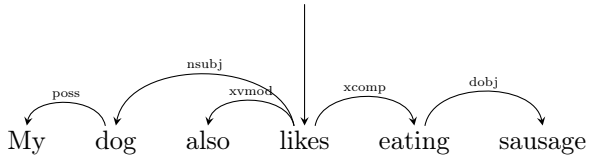


Figure 4.5: Stanford Dependency Tree

4.5.2 Generation

To assign dependency structures to sentences, we used the Stanford Dependency Parser, that can be downloaded from their website, as well as used online (De Marneffe et al., 2006). The parser provides 5 variants of a typed dependency representation, of which the most basic one corresponds to the earlier imposed conditions on dependency structures (De Marneffe and Manning, 2008). A picture of a dependency parse that was generated by the parser is depicted in Figure 4.5.

Unfortunately, the Stanford dependency parser used does not cover all tokens of the input sentences, as dependency relations between words and punctuation are not present in the Stanford Dependency Representation. The resulting dependency trees do thus not always cover the entire sentence.

4.6 Summary

In this chapter we have presented the basis for the original research conducted for this paper. At the start of this chapter we have stated the aims of this research, and provided a short summary of the background we presented in earlier chapters. We formulated three questions

1. Are dependency structures universal for languages?
2. What are the reasons that dependency parses are not entirely preserved during translation?
3. Can dependency grammars be used to construct a bilingual compositional grammar?

We have listed the main assumptions that we made in our empirical investigation of the three questions presented, and we have revisited the two main ingredients of this investigation: HATs and dependency parses.

Chapter 5

A Bilingual Perspective on Dependency Parses

After having discussed the main ingredients for our research, as well as its foundations, we will now present our findings on the consistency between dependency parses and HATs and the experiments that led to these findings. This chapter consists of 5 sections. Firstly, in Section 5.1, we will give a description of the data that were used for all experiments. In the subsequent sections 5.2 and 5.3 we will report and analyse the consistency of dependency parses and HATs according to two different consistency metrics. In Section 5.4, we will further analyse the consistency between HATs and dependency parses, by performing a manual analysis on a part of the data. As usual, we end the chapter with a short summary.

5.1 Data

We had 4 large automatically aligned parallel corpora available (see 5.1), for the language pairs English-Dutch, English-French, English-German and English-Chinese. The first thee corpora were data from the European Parliament taken from the Europarl corpus (Koehn, 2005), while the English-Chinese data came from the Hong Kong Parallel Corpus. The word-alignments were induced using GIZA++ (Och and Ney, 2003), using the 'grow-diag-and-final'-heuristic mentioned in Section 3.3, with 4 iterations on model 1, and 3 iterations with the hmm model, model 3 and model 4. The corpora were tokenised and lowercased before GIZA++ was run. In general, the guidelines for building a baseline for the WMT workshops were followed.[1]

In addition to the automatically aligned corpora, we had access to 5 manually aligned corpora (see 5.2). These corpora were much smaller than the automatically aligned corpora, and covered the language pairs English-French (Graca

[1] See http://www.statmt.org/wmt07/baseline.html

Language pair	Source	Size	Alignments
Eng - Du.	European Parliament	945167	GIZA++
Eng - Ge.	European Parliament	995909	GIZA++
Eng - Ch.	Hong Kong Parallel Corpus	1723487	GIZA++
Eng - Fr.	European Parliament	949408	GIZA++

Table 5.1: The automatically aligned datasets used for the experiments in this thesis.

Language pair	Source	Size	Alignments
	Europarl	987	Padó and Lapata (2006)
Eng - Ch.	Hong Kong Parallel Corpus	1723487	GIZA++
Eng - Fr.	Hansard	447	Och and Ney (2000)
	Europarl	100	Graca et al. (2008)
Eng. - Sp.	Europarl	100	Graca et al. (2008)
Eng. - Port.	Europarl	100	Graca et al. (2008)

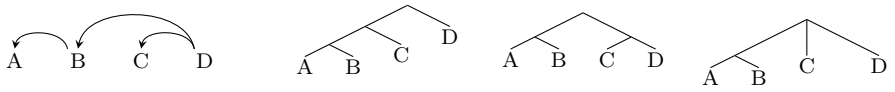
Table 5.2: The manually aligned the datasets used for the experiments in this thesis.

et al., 2008; Och and Ney, 2000), English-Spanish, English-Portuguese (Graca et al., 2008) and English-German (Padó and Lapata, 2006).

5.2 Direct Consistency

Quantifying the consistency between a dependency tree and an alignment is not a trivial task. When the parts described by a dependency tree are all parts according to the alignment, there is no dubiety: such an alignment should get an optimal score. If this is not the case it is less clear, as there are multiple things that should be taken into account. Not only should the measure of consistency be based on whether the head and its dependent are both phrases according to the alignment, they should also be combined in a reasonable fashion.

To give an abstract example, consider the following dependency parse and alignment trees:



All the parts that exist in the dependency tree also exist in all alignment trees. The third alignment tree is obviously most similar to the dependency parse, as it prescribes the same compositional structure and uses the same number of rules. However, the first and second alignment tree are indistinguishable if only the number of correct (and possibly incorrect) nodes is considered, as they both contain two correct and one incorrect node. However, the second alignment tree

seems more in line with the dependency parse than the first one, because it does not only prescribe that A should be combined with B into a new part, but also that C and D are combined with one rule.

5.2.1 Experiment 1

The analysis of the example provides an insight: as it seems, alignment trees and dependency parses are intuitively compatible if the relations in the dependency parse are respected by the alignment tree, which gives rise to a straightforward definition of consistency between dependency relations and HATs:

Definition 11 (Direct Consistency)**.**
Let $s = w_1\ w_2\ \cdots\ w_n$ be a sentence, and $D = \{(i,j)|$ there is a dependency arrow from word w_i to word $w_j\}$ a set of dependencies describing a dependency tree for s. Let $[k,l]$ denote the subsequence of s from word k to word l. Let span(j) $= [m,n]$, iff m and n are the minimum and maximum position that can be reached from w_j by following the directed dependency arrows, respectively. A dependency relation (i,j) is said to be respected by an alignment tree T over s if and only if there is a node in T of which both $[i,i]$ and span(j) are children.

In other words, a dependency relation (i,j) is consistent with a HAT if the head word i and the phrase headed by the j are siblings in the HAT. The consistency $C(D,H)$ of a set of dependency relations D forming a dependency parse with a HAT H can now be expressed as follows:

$$C(D,H) = \frac{|D' \cap S_H|}{|D'|}$$

Where $D' = \{(i, span(j)|\ (i,j) \in D\}$, the set of span relations prescribed by D, and $S_H = \{(i,j)|$ span i and j are siblings in $H\}$.

The score of a HAT can be computed recursively, as is expressed in the following definition:

Definition 12 (Score of an Alignment Tree)**.**
$s = w_1 w_2 \ldots w_n$ be a sentence, and $D = \{(i,j)|$ there is a dependency arrow from word w_i to word $w_j\}$ a set of dependencies describing a dependency tree for s. Let $D' = \{(i, span(j)|(i,j) \in D\}$, where span(j) is as defined before. Let H be an alignment tree of s. The (unnormalised) score of H with D is now defined as the score of its highest node N:

$$E(N_a, D) = \sum_{c \in C_{N_a}} E(c,D) + \sum_{c_1 \in C_{N_a}} \sum_{c_2 \in C_{N_a}} B(c_1, c_2)$$

With base case $E(N,D) = 0$, $B(c_1,c_2) = 1$ iff $(c_1,c_2) \in D'$, and C_N the set of child nodes of N.

The score can be normalised by dividing by $|D|$.

Under this definition of consistency, the first alignment tree in the previous example would receive a score of 1/3, the second alignment tree would receive a score of 2/3, and the third alignment tree a score of 3/3. Which seems to correspond with their adequacy of describing the dependency structure.

Note that the score we assign to alignment trees (the number of retrieved dependency relations divided by the number of dependency relations) corresponds to the *recall* of the dependency relations in the tree. Typically, recall is used in combination with precision, as both of them can often easily be fooled individually, but cheating the one will result in a low score for the other.

For HATs and dependency relations, it is not immediately clear how the precision measure should be constructed, as HATs are in some aspects underspecified with respect to dependency parses. It is not immediately clear which relations the HAT guesses should be counted as wrong, because a HAT guesses many relations than cannot coexist in one dependency parse (e.g., if it can make (x,y) true, it can also make (y,x) true), and if all these relations were to be considered, no HAT would ever receive the optimal score for the precision measure. It seems therefore more sensible to consider the maximal number of relations a HAT can make true all at once. However, any dependency parse of the sentence the leafnodes of the HAT dominate will have the same number of dependency relations, and the number of relations a HAT can maximally make true all at once will thus be equal to $|D|$, rendering the measure for precision and recall identical.[2]

5.2.2 Results

We have measured the consistency of the first 10.000 alignments of our automatically aligned datasets, and all the manually aligned sentence pairs. We have found the best scoring HATs by assigning weights to the grammar rules of the HATs, that were associated with the number of dependency relations a grammar rule made true, and parsing the HAT with this weighted grammar. We reported on the scores for sentences shorter than 10, 20 and 40 words.

	Consistency Score		
Language Pair	$\|s\| < 10$	$\|s\| < 20$	$\|s\| < 40$
English-Dutch	0.47	0.42	0.40
English-French	0.46	0.42	041
English-German	0.44	0.41	0.38
English-Chinese	0.59	0.48	0.42

Table 5.3: The empirically determined consistency scores of all available automatically aligned corpora according to consistency definition 11.

[2]Another way to positively bias the recall measure could be through manipulating the branching factors of the nodes in the HATs, as are of influence to the *number* of different groups of relations a HAT can make true all at once. Fortunately, this is not an issue when scoring HATs, as their maximum recursivity does not allow much flexibility in the branching factors of their wnodes.

Language Pair	Consistency Score $\lvert s\rvert < 10$	$\lvert s\rvert < 20$	$\lvert s\rvert < 40$
English-French (Hansard)	0.63	0.54	0.51
English-French (LREC)	0.49	0.47	0.47
English-German (Pado)	0.43	0.42	0.41
English-Portuguese (LREC)	0.47	0.45	0.45
English- Spanish (LREC)	0.51	0.48	0.48

Table 5.4: The empirically determined consistency scores of all available manually aligned corpora according to consistency definition 11

5.2.3 Analysis

The consistency scores, which are depicted in Table 5.3 and 5.4, are very low. On average not even half of the dependency relations of the English sentence are respected by any HAT. The dependency relations of shorter sentences are generally somewhat better respected than the dependency relations of longer sentences. The difference between the scores of the automatically aligned datasets and the manually aligned datasets is lower than we expected, which could indicate the influence of mistakes in the automatically aligned datasets is relatively small. However, the large difference between the two manually aligned French datasets indicates that is more likely due to the fact that the manually aligned datasets are too small to get a significant result.

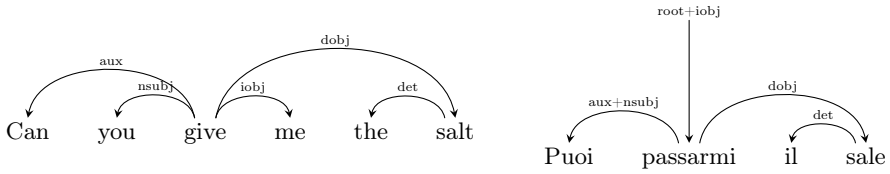
Without further elaboration, we have previously mentioned that the maximally recursive HATs and the linguistically motivated dependency structures possibly exploit compositionality in a different fashion. We suspect that this issue causes the scores to be lower under the current consistency definition, as we will illustrate with two examples.

Firstly, consider the sentence 'I give you flowers', and its (word-for-word) translation 'Ik geef jou bloemen', with dependency parse:

The tree depicted next to the dependency parse is the only tree that respects all dependency relations. The sentence is very short and the dependency parse completely flat, as the sentence consists of a predicate and three single word arguments. In a tree that respects all relations according to Definition 11, 'I', 'give', 'you', and 'flowers' are all siblings, which means the tree depicted next to the dependency parse is the only tree obtaining a maximal score. However, as the sentence is a word for word translation, all subsequences are translation admissible, and all HATs will be completely binary. Even though the translation

seems intuitively compositional, none of the HATs will receive a score higher than 1/4, because the dependency parse is not maximally branching.

A similar situation arises when two arguments are translated into one, which happens, e.g., when arguments are translated as pre- or suffixes, when verbs do not require a subject or when spaces are emitted. Consider for instance the sentence 'Can you give me the salt' and its Italian translation 'puoi passarmi il sale':



Once again, the predicate-argument structure of the sentence is well preserved. However, some of the arguments are merged into single words in the Italian sentence. Besides the issue raised in the previous paragraph (the dependency structure is not maximally compositional), an additional problem thus arose: except for 'me', none of the arguments can combine directly with 'give' in an alignment tree, because 'give' and 'me' are one word in Italian, and will thus form a unit on their own. 'Can' and 'you' cannot combine with 'give' at all, because they are first combined together. The maximum score a HAT of this translation could receive would thus be 2/5, in which only the relations (give, me) and (salt, the) are respected.

5.3 Deeper Consistency

Of course, we do not know if the low consistency scores can in fact be attributed to the fact that dependency parses are not maximally recursive, or have another cause. To investigate this issue, let us first consider the average branching factors of the nonterminal nodes in the HATs and the compositional structures given rise to by the dependency parses.

In Table 5.5 we see, that the average branching factor of a HATnode is around a point lower than the average branching factor of a similar tree that is constructed based on a dependency parse, which supports the hypothesis that the HATs are more recursive (deeper) than the dependency parses.

A closer look at the distribution of the branching factors 5.1 shows that the HATs have much more binary expansions, while the dependency structures have more nodes that have 3-7 children. In a second experiment, we test if higher consistency scores can be reached if there is a better match between the two types of compositionality.

Language Pair	Dependency	best HAT
English-Dutch	3.1	2.1
English-French	3.1	2.1
English-German	3.1	2.11
English-Chinese	3.0	2.2

Table 5.5: The average branching factors of the nodes in the found best scoring HATs and the compositional structures given rise to by the dependency parses accoring to which they were scored.

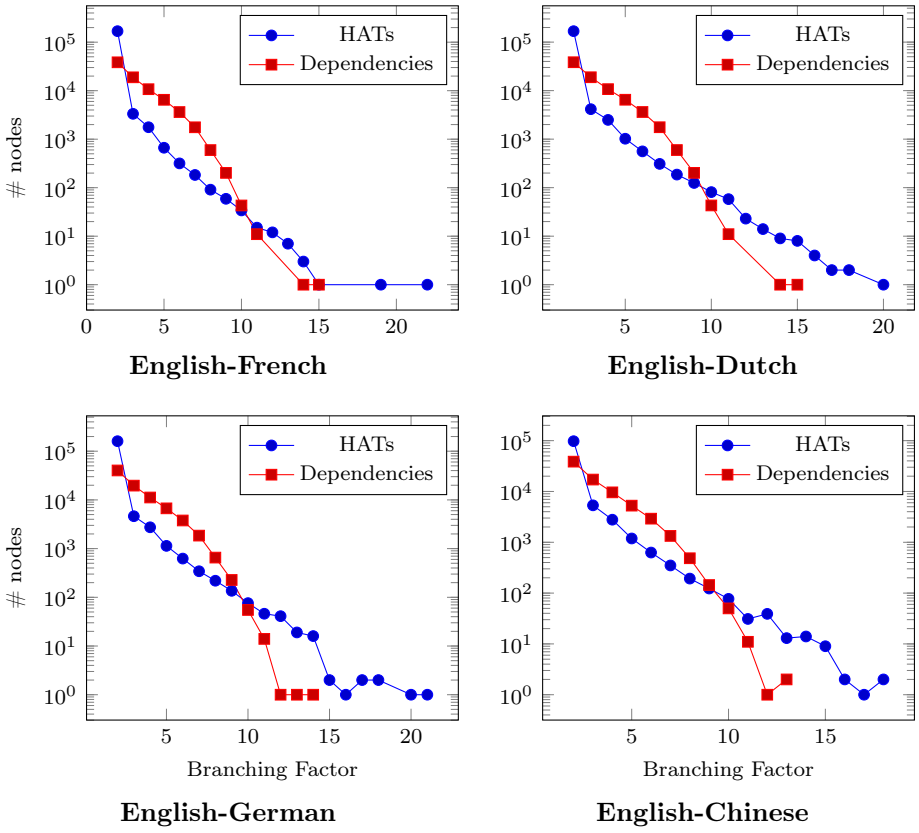


Figure 5.1: The branching factor of HATS and the structures given rise to by the dependency parses of the 4 automatically aligned datasets, plotted against the number of nodes with this branching factor. The branching factor is plotted on a logarithmic scale.

5.3.1 Experiment 2

To test the influence of discrepancy of compositionality in our data, we give a more flexible interpretation of consistency. Rather than defining consistency as direct similarity, as in Definition 13, we also assign maximal scores to HATs in which the arguments are combined with their predicates in stages. (e.g., combine 'give' with 'you'; combine 'give you' with 'flowers' and finally combine 'I' with 'give you flowers'). Under this perspective, a dependency relation is respected by a HAT if the phrase headed by the dependent is siblings with the head (as before), or the head plus arguments the head earlier combined with ('I' with 'give you flowers'). The set representation of this consistency measure is cumbersome, and we will therefore just give the recursive definition that was used to compute scores. As also this definition is quite hard to read, a somewhat formal description is given below.

Definition 13.
Let $s = w_1\ w_2\ \cdots\ w_n$ be a sentence, and $D = \{(i,j)|$ there is a dependency arrow from word w_i to word $w_j\}$ a set of dependencies describing a dependency tree for s. Let span(j) be the range $[m,n]$, where m and n are the minimum and maximum position that can be reached from w_j by following the directed dependency arrows, respectively, and span($i_1 \cdots i_n$) = $[m,n]$, where m and n are $\min_{j\in\{1,n\}} \min span(i_j)$ and $\max_{j\in\{1,n\}} \max span(i_j)$, respectively. Let (i,j) be a dependency relation in D, and let $l_1,\ldots,l_n$ and $r_1,\ldots r_k$ be the the left and right dependents of i, such that $r_k < j$ (if j is a right dependent) or $l_1 > j$ (if j is a left dependent). A dependency relation (i,j) is said to be respected by an alignment tree T over s if and only if one of the following conditions is true:

1. *There is a node in T f which both $[i,i]$ and span(j) are children.*
2. *$\exists x,y$ and a node in T of which span($l_x \cdots l_n\ i\ r_1 \cdots r_y$) and span(j) are both children.*

In other words, there are two conditions under which a dependency relation (i,j) that is part of a dependency tree D is considered to be made true by H:

1. i and $span(j)$ are siblings in H (as before)
2. $span(j)$ is siblings with a node all whose children are either i or $span(k)$, where k is a dependent of i.

The extension of the definition of consistency between HATs and dependency parses handles part of the discrepancy between the types of compositionality of HATs and dependency parses. The first example from the previous subsection receives an optimal score under consistency definition 13 (as will all word for word translations), as well as preserved structures in which the arguments are reordered a little. The second example, however, will still not receive an optimal score, as 'can' and 'you' can still not combine with give one by one.

The more flexible Definition 13 thus does not cover all discrepancy between dependency parses and HATs. It can nor account for two arguments that

are translated together, and neither for too severe reordering of arguments of dependency parses in translation. However, we have chosen to not make an even more flexible version of consistency, for which the most important reason is that we believe we detract too far from what dependency parses are encoding. Although allowing arguments to combine with each other first before combining with the predicate would in some situations lead to a more appropriate score, it will in many situations assign optimal scores to HATs that do not adequately represent the compositional structure prescribed by the dependency parse.

Remark As previously mentioned, the Stanford Dependency style does not include punctuation. Where in the first consistency definition this did not really get in the way, it becomes problematic for the second, as tokens that are not involved in any dependency relation can interfere with the definition of consistency. To account for this, we allowed punctuation (or other tokens not processed by the dependency parser) to combine freely with the closest units (of arbitrary size), without them affecting the score.

5.3.2 Results

In the second experiment we followed the same protocol as in the first: we scored the first 10.000 alignments of our automatically aligned datasets and all manual alignments, based on how many of their relations were true in their best HAT (according to Definition 13). We reported on the scores for sentences shorter than 10, 20 and 40 words.

	Consistency Score		
Language Pair	$\|s\| < 10$	$\|s\| < 20$	$\|s\| < 40$
English-Dutch	0.79	0.74	0.71
English-French	0.80	0.77	0.76
English-German	0.75	0.71	0.68
English-Chinese	0.76	0.67	0.62

Table 5.6: The empirically determined consistency scores of all available automatically aligned corpora according to consistency definition 13.

5.3.3 Analysis

When accounting for some of the discrepancy in compositionality, the consistency scores are much higher, indicating that a large part of the previous low scores were indeed due to the mismatch of compositionality of HATs and dependency parses. However, the scores are still far from maximal (at most 0.85, for the manually aligned Hansard set).

We have created a plot of the distribution of the scores of the automatically aligned datasets. Except for the Chinese datasets, which will not further be discussed as the author has no knowledge of this language whatsoever, the

Language Pair	Consistency Score $\|s\| < 10$	$\|s\| < 20$	$\|s\| < 40$
English-French (Hansard)	0.85	0.80	0.78
English-French (LREC)	0.78	0.82	0.82
English-German (Pado)	0.82	0.80	0.77
English-Portuguese (LREC)	0.75	0.76	0.76
English- Spanish (LREC)	0.79	0.80	0.80

Table 5.7: The empirically determined consistency scores of all available manually aligned corpora according to consistency definition 13.

maximal score is by far the most common score. Generally, there are few alignments that have a score just below the maximum score, which suggests that constructions do not prevent just one dependency relation from being preserved, but multiple at a time. The low number of sentences that have a very low score confirms the intuition that even sentences of which the dependency parses are not preserved are still for the larger part translated compositionally. In the next section, we will further investigate whether the lower scores can be explained by general phenomena, or are all individual cases.

5.4 Manual Analysis

We have conducted a manual analysis to gain more insight in the causes of inconsistency between dependency parses and HATs. We have restricted ourselves to an analysis of the manually aligned corpora, as their alignments will be more accurate. The author of this paper masters neither Spanish nor Portuguese, the analysis is thus restricted to the language pairs English-German and English-French.

We randomly selected 100 sentences from the English-German dataset from Padó and Lapata (2006) and the English-French from Graca et al. (2008).[3] For every sentence, we determined whether the suboptimal score was due an error (in the dependency parse, the alignment or the translation), to severe rewording where a more literal translation was also available, or had another reason (the set of reasons considered slightly differ per language pair). In case there was an error in the data, we checked if the sentence received an optimal score after correcting the error. In case of uncertainty a native speaker of the language was consulted. The results can be found in the following two subsections. We have plotted the score distribution for the two datasets, to confirm they show a pattern similar to the automatically aligned datasets.

[3] Actually, the latter contained only 100 sentences, but lets say they were picked randomly.

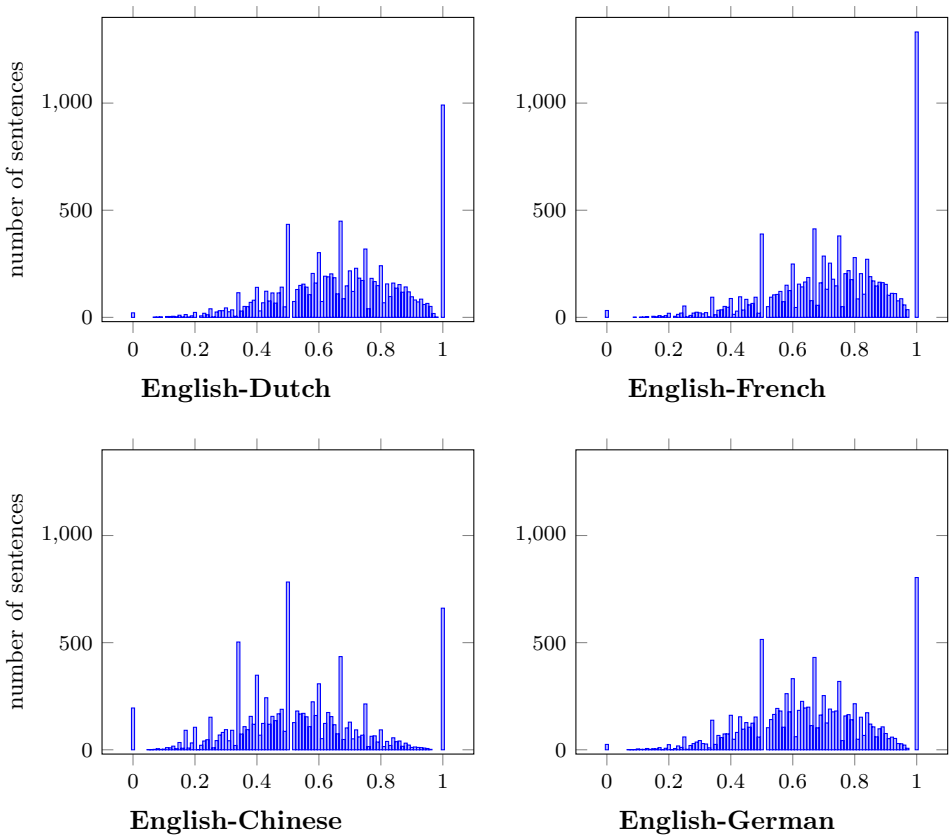


Figure 5.2: Distribution of the scores of the consistency scores of the first 10,000 alignments of the available automatically aligned datasets according to consistency definition 13.

5.4.1 English-German

The percentage of manually aligned sentences that obtained a maximal score is higher than the percentage of automatically aligned sentences that scored maximally. In the 100 sentences we selected from the manually aligned German-English corpus, 53 sentences have a score of 1.

In translation from German to English, we have found 3 main causes of dependency relations not being preserved during translation, besides errors in the data. Table **??** provides a summary of the occurrences of these cases in the data, and reports on the average scores of sentences classified as falling into these categories. We will discuss them in the following paragraphs.

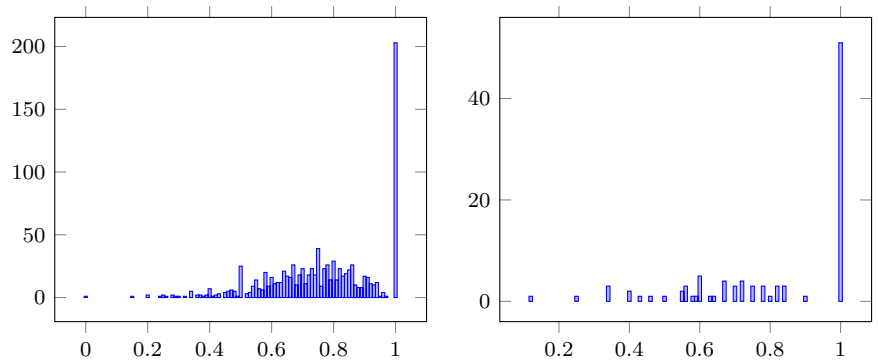


Figure 5.3: Distribution of the scores of the scores of the manually aligned datasets of Padó and Lapata (2006) (English-German) and Graca et al. (2008) (English-French) according to consistency definition 13.

Main Cause	**#sentences**	**Average**	**Average Corrected**
Dependency Error	8	0.60	0.84
Alignment Error	2	0.69	1
Translation Error	1	0.5	1
Literal	5	0.52	1
phrasal	9	0.57	
Max compositionality	15	0.61	
Different construction	3	0.72	
Other	4	0.71	

Table 5.8: The different causes we found for non-optimal scores, and how often they occurred in the manually studied sample. The fist column lists the category, the second the number of sentences that was classified into this category and the third column gives the average score of these sentences. In case of an error, also the average score of the sencences after correcting the error is reported. Specific examples of sentences classified in this category can be found in Appendix A.1

Maximal Compositionality

As we have previously discussed, dependency parses are generally not maximally compositional. Although our scoring metrics allow to combine predicates and arguments to combine one by one, they do not allow arguments to combine together before they combine with their predicate. In translation from English to German, a language in which the word-order is relatively free, reordering of the arguments was a major cause for low consistency scores.

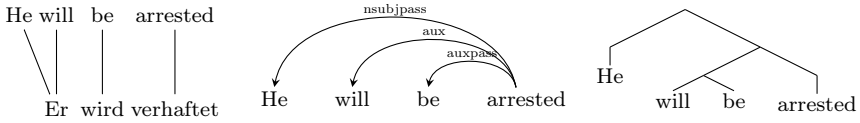
Such cases were mainly caused by modifiers of verbs that occurred in a different place (such as 'also'), verbs that split into two parts during translation, and hold all their arguments between their first and second part in German, or

switching of arguments in a fashion that cannot be resolved by a binary tree. Examples from the data are given in Appendix A.1.1.

Phrasal Translations

HATs account well for phrasal translations, and whenever a unit consisting of a predicate and one or more arguments consisting of just one word was translated phrasally, the dependency relations within were considered preserved. However, sometimes sequences of words that were phrasally translated did not constitute a unit according to the dependency parse, which caused at least two dependency relations not to be preserved.

In translation from German to English, there were two cases in which phrasal translation beyond the dependency level occurred often. Firstly, when English auxiliary or linking verbs are translated to English, they are often translated together with another word that is not its head. For instance: 'will be' is translated into 'wird', 'does not X', is translated into 'nicht X', or 'has been' is translated into 'wurde'. In dependency parses, auxiliary or linking verbs are rarely considered head of the sentence (e.g., in the sentence 'he is a doctor', 'doctor' will be marked head). If this linking verb is then translated phrasally together with another verb or word, this relation cannot be resolved (sometimes this can thus also be seen as a case of non maximal compositionality).



A second case for which phrasal translation beyond the dependency level occurs is in the translation of prepositions. Verbs that are accompanied by a preposition in English, are sometimes not accompanied by one in German. If the preposition is considered to be translated into the verb, this causes the dependency relation between the verb and the preposition to break down (and possibly all relations in the preposition). Examples are given in Appendix A.1.2.

Non literal translations

In the German-English corpus, we have found some examples of translations whose structure deviated much from the original sentence, where a more literal alternative was also available. For instance, the translation of 'Traffickers demand astronomical amounts to smuggle their customers to the West.' into 'Die Haendler fordern von den Kunden, die sie in den Westen schmuggeln, astronomische Summen.'. The rewording in this sentence is quite severe, whereby the sentence is assigned a score of 0.4. However, the also perfectly grammatical and acceptable translation 'Schmuggler verlangen astronomische Summen, um ihre Kunden in den Westen zu bringen', would receive an optimal score. In all cases where

we judged a more literal translation was available, this literal translation was checked and accepted by a native German speaker.

Other

Seven sentences, whose average score was 0.72, could not be grouped in one of the previously mentioned categories. In three of these cases, we judged the type of construction simply deviated so much during translation, that it was hard to account for compositionally. We will list these three examples:

1. The translation of 'I would like to see this question investigated' into 'diese Frage muss meines Erachtens geklaert werden'.
2. The translation of 'X would fail to find a buyer' into 'X wurde keine abnehmer mehr finden'.
3. The translation of 'The council received more than 100 questions' into 'Es sind mehr als 100 Anfragen an den Rat gerichted worden'.

Of the resulting four translations, we found it hard to pinpoint what the exact reason was for the lower score. In two of these cases, a word earlier in the sentence was repeated later in one sentence, but not in its translation ('**I** think **I** can say' - '**Ich** glaube sage zu koennen', and ('**to** withdraw and stop' - 'sich zuruck**zu**ziehen und schweigen **zu** lassen'). As the repeated word was aligned to both words in the other sentence, this caused the dependency structure to break. In the third sentence, the dependency parse did not seem to capture the structure of the sentence very well, but it was hard to fix without breaking the linear tree structure of the parse. In the fourth sentence, the English sentence seemed to be ambiguous, and the German sentence chose another meaning than the dependency parse.

In conclusion

In conclusion, we have seen that most of the lower scores were not due to non-compositionality. In Table 5.9 we see, that the majority of the sentences in the corpus obtained an optimal score, an additional 12 would have obtained an optimal score if the corpus were error free, or more literal (but acceptable) alternative translation were chosen, and 18 sentences did not receive a maximal score due to discrepancy of the type of compositionality of the HATs and the dependency parses. Furthermore, although not listed in Table 5.9, it seems that phrasal translations beyond the dependency level should be accountable for in a compositional grammar, if some small adaptations on a general level are made.

5.4.2 French

In the French-English corpus, reduction of the score due to a mismatch in recursion type was much less prevalent. Most of the low scores were due to severe rephrasing of the target sentence. In some cases, we judged that a more literal

Type	**#sentences**
Maximal score	53
Maximal score after correction error	7
Literal alternative with maximal score	5
Non compositionality due to maximality	18
Total	83

Table 5.9: Summary of the manual analysis of German sentences: how many of the sentences are intuitively considered to have a compositional translation.

translation would have been just as acceptable (9 times), but in cases where only a couple words were freely translated, we counted it as a phrasal translation beyond the dependency level. There were very many of such phrasal translations, some of which seemed to follow a pattern extendible to other sentences. We will discus the cases that could be described by a somewhat general rule.

Main Cause	**#sentences**	**Average**	**Average Corrected**
Dependency Error	12	0.57	0.96
Alignment Error	2	0.75	1
Translation Error	0	0	0
Literal	11	0.66	1
phrasal	20	0.65	
Max compositionality	1	0.75	
Other	3	0.42	

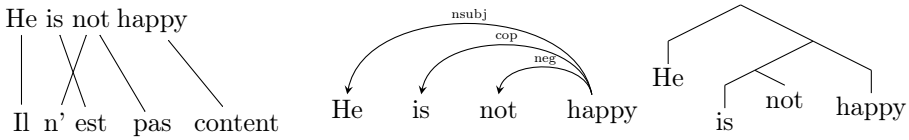
Table 5.10: The different causes we found for non-optimal scores, and how often they occurred in the manually studied sample. The fist column lists the category, the second the number of sentences that was classified into this category and the third column gives the average score of these sentences. In case of an error, also the average score of the sencences after correcting the error is reported. Specific examples of sentences classified in this category can be found in Appendix B.1

Phrasal translation of negation

In the previous chapter we have argued that HATs can account nicely for phrasal translation of negation. Dependency parses, however, do not. In dependency parses, a linking verb is never considered the head of the sentence, and negation of a sentence whose main verb is such a verb can thus not be resolved. We will illustrate this with a short example, examples from the data can be found in Appendix A.2.2.

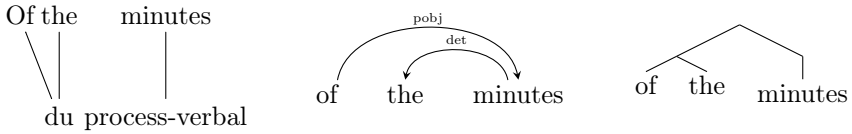
Consider the sentence ‘He is not happy’ and its French translation ‘Il n’est pas content’. According to the dependency parse, ‘happy’ is the head of the sentence, and ‘He’, ‘not’ and ‘happy’ are all arguments of this predicate. However, ‘is not’ is phrasally tranlsated into ’n’est pas’, and both (‘happy’, ‘is’) and (‘happy’, ‘not’)

will thus not be preserved during translation. However, this 'non-compositionality' seems to be due to an unfortunate choice in the dependency parse, rather than to the fact the translation could not be composed of the translation of its parts.



Phrasal translations of prepositions

In French, prepositions are often contracted with the verb. For instance, the translation of 'of the minutes', is 'du process-verbal'. A dependency parse will see 'of the minutes' as a prepositional phrase, of which 'of' is the head. 'the minutes' will be seen as the argument of 'of'. However, when 'of the' are phrasally translated into 'du', both the relations ('of','the minutes') and ('the','minutes') will be considered not preserved.



Other

The three unclassified sentences were either not understandable for the author in the context, or of a too specific particular political jargon to be assessable.

Conclusion

In conclusion, we have seen that many of the scores were due to non literal translation. In addition, it seems that some of the design choice made in dependency parses, in particular the choice to not always make a verb the head of a sentence, are not particularly suitable for translation from English to French. Table 5.11 shows how many sentences would have obtained a maximal score if the data were optimal for our purposes.

5.5 Summary

In this chapter we have described the experiments we have conducted to establish the level of consistency between dependency parses and HATs. We have showed that HATs and dependency parses are not very similar, if similarity is defined in a direct way (i.e., there is often not a HAT that is structurally identical

Type	**#sentences**
Maximal score	51
Maximal score after correction error	10
Literal alternative with maximal score	11
Non compositionality due to maximality	1
Total	73

Table 5.11: Summary of the manual analysis of 100 French sentences: how many of the sentences are intuitively considered to have a compositional translation.

to the dependency parse). In a second experiment, in which more recursive versions of dependency parses were considered, we found that the mismatch in compositionality type is an important reason for this non similarity. The low consistency of the corpus according to the dependency parses is thus not entirely due to non compositionality. However, even in the second experiment, it was still not possible to consistently select alignment trees using the dependency parses. On average, around only 80% of the compositional structure prescribed by the dependency parse was respected by any HAT. In a manual analysis, we showed that many of the inconsistencies should not be attributed to non-compositional translation, but are due to errors, non literal translation, design choices made in the dependency formalism, or further mismatches in recursivity that were not accounted for.

Chapter 6

Discussion and Future Work

In this thesis, we have tried to make a small step to a better understanding of compositional translation, and its feasibility for translation between two natural languages, by empirically analysing the patterns that can be found in translation data. After we motivated the need for such a better understanding and discussed related work (Chapter 2 and 3), we have laid out the framework that we used (Chapter 4), and presented and discussed the results (Chapter 5). In this chapter, we will discuss two things.

Firstly, we will look back at the work in this thesis, and discuss how well the results satisfy our expectations and how they can be interpreted from a wider perspective (Section 6.1). Secondly, we will propose a possible extension of this work, that can hopefully be the start of a new research project. As all previous chapters in this thesis, this chapter - and therefore this thesis - will end with a brief summary.

6.1 Discussion

In the introduction of this thesis, we formulated 3 research questions:

1. Are the compositional structures suggested by dependency parses universal for language?
2. What are the reasons dependency structures deviate during translation?
3. Can dependency parses be used to construct a bilingual compositional grammar?

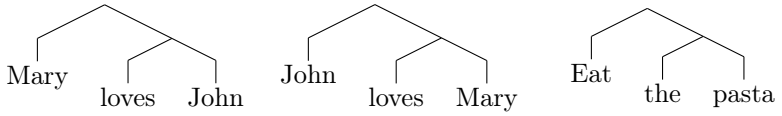
Before we discuss the answers that we have found to these questions, we would like to take a step back, and recall the reasons that we asked exactly these questions.

6.1.1 Compositional Translation from an Empirical Perspective

In the first chapters of this thesis, we have argued for an empirical approach to compositionality. Theoretically, it seems almost impossible to establish how compositional translation is, because compositionality - although often described by a seemingly simple definition - is a very flexible concept. Empirically, however, we can study the structures that exist in translation data, quantify how reasonable it is that they are generated by a compositional system, and gain information about what this system looks like. If we can find a method to consistently pick one of the compositional structures that is supported by the data for every sentence, this is, with the parsing techniques we dispose of today, likely to be extendible to unknown sentences, and therefore gives a solid grounding to compositional translation as a strategy.

6.1.2 Finding Consistency

Of course, finding consistency in a corpus of trees is far from trivial if no external information is used at all, which can be easily understood when looking at the following three trees:



Intuitively, the first two sentences are very similar - they express similar meanings with a similar form - while the third sentence is somewhat different. From only the trees, this observation cannot be easily captured, as the only thing they differ in are the leafnodes. Without knowledge of the world to tell us that 'John' and 'Mary' are in some sense comparable to each other, but not really to pasta, and that 'loves' takes two arguments, it is nigh impossible for a computer to detect the systematicity in these sentences. That is not to say, that it is entirely impossible (see e.g., Bod, 2006, for an almost completely unsupervised account of structure learning), but to make this computationally feasible many simplifying assumptions are necessary.

In this thesis, we therefore chose to look at a subproblem: the incorporation of monolingual information about the predicate argument structures of the source sentences. Given the elaborate monlingual knowledge about language already available in the scientific world, it seems sensible not to start from scratch, but to build on previous research. Although there are several studies that investigate the correspondence between translation structures and constituency structures, investigations about the usefulness of dependency grammars for MT seem to be underrepresented in literature. In this thesis, we provided such an investigation, focusing on the previously mentioned questions.

6.1.3 Summary of Results

As the results were elaborately discussed in the previous chapter, we will now suffice with a short summary, and focus on the conclusions that can be drawn from them.

Initially, we looked at the suitability of dependency parses as a bilingual system on themselves. We found that plain dependency relations as they are, are not sufficient to consistently select compositional translation trees for a corpus. Although dependency parses have a strong semantic motivation, the structures they prescribe are rarely entirely preserved during translation, and for none of the language pairs considered the overall consistency according to the dependency parses was much higher than 50%.

A large part of this inconsistency should be attributed to the fact that dependency parses are generally quite flat, which is undesirable for a bilingual grammar and conflicts with the precondition we imposed on the compositional translation trees we considered (maximal recursivity). When deeper versions of dependency parses were considered, the consistency of the selected trees according to the dependency parses was at least 50 percentage point higher for all datasets.

However, to learn a bilingual grammar that can successfully predict the compositional structures of new sentences, even an 80% consistency is still quite low. With a manual analysis, we investigated whether the lower scores were assigned to sentences that are intuitively not compositionally translated, or were caused by other reasons. The results of this analysis were hopeful. We found that errors in the data were a prevalent reason for low scores, as well as small phrasal translations that seemed to know a certain systematicity (e.g., the translation of prepositions was often problematic). Although the investigated corpus was small (2 times 100 sentences), these results make us optimistic for the future of compositional translation.

6.1.4 Conclusion

With the results summarised in the previous susbsection, we have satisfactory answered our first two questions, but we have not found a conclusive answer to the third one. Although our results indicate that there is some systematicity to the issues that are troublesome for finding consistency through dependency parses, we have not tested whether dealing with some of these issues could significantly improve the result. Although we believe that such adaptations can be effective, we have serious doubts about the scalability and robustness of such an approach. Rather, we think that a statistical learning approach that uses information from dependency parses (rather than strictly following it), would be more effective to learn such adaptations, and would be more robust to errors. In the remainder of this chapter, we propose an approach to efficiently *learn* a grammar using HATs and dependency grammars. Although an implemented version of the algorithms is available, we will leave the execution and evaluation of the resulting system for future work.

6.2 Learning a Compositional Grammar using Dependency Information

In this section, we will describe the algorithms and techniques that can be used to learn a bilingual grammar from dependency parses using the Expectation Maximization algorithm (Dempster et al., 1977). The focus lies on learning the source side of this bilingual grammar, the fact that alignment constraints are taken into account in designing this grammar ensures that the resulting source grammar indeed has a counterpart in the target language.

There are two main obstacles for learning a bilingual grammar from a translation corpus:

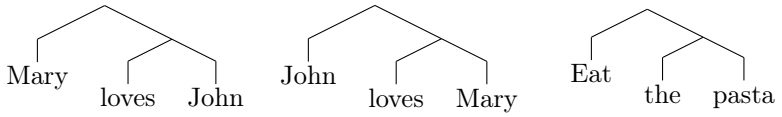
1. We cannot compare the structures for different sentences, and therefore we cannot detect similarities or differences.

2. For every sentence, we have many structures, and we do not know how to interpret these structures. That is, a compositional translation of a sentence is described by *one* structure, and we do not know which of these structures should be preferred over other structures.

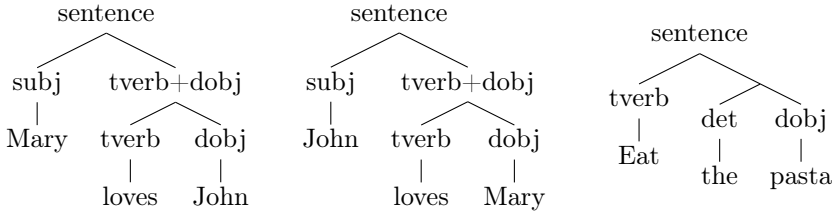
In this section, we will propose methods to encounter both these issues.

6.2.1 Comparing structures

Machines can only detect patterns in a set of structures if they have information that allows them to compare these. As is quite usual, we will provide this information by labelling the parts (nodes) of the structures. To some of the readers this may seem obvious, but we will provide a small example to illustrate it, revisiting the the following three trees:



Intuitively, the first two sentences are very similar - they expresses similar meanings with a similar form - while the third is somewhat different. This observation cannot be easily captured by a machine, as it lacks the world knowledge to make this inference. However, if it had seen the following structures:



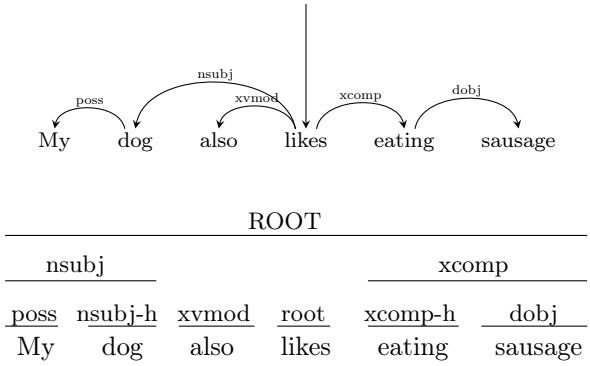


Figure 6.1: A dependency parse, and the basic labels that are inferred from it.

A machine could have used the node labels to figured out that the first two trees are structurally more similar to each other than to the third tree.

However, consistently providing labels for all nodes of all trees in a corpus is not a trivial task, as monolingual analyses of sentences typically only provide labels for a subset of all spans. We will propose a method to label all spans of a sentence, based on dependency parses, such that the labels are consistent over the corpus.

Basic Labels

As dependency parses assign labels to relations between words, they do not directly provide labels for spans. As earlier in this thesis, we will circumvent this issue by interpreting a relation X between A and B as a relation between A and the phrase headed by B. This last phrase will be assigned label X. If word B is not assigned a label by following this procedure, it will be labelled 'B-head'. Figure 6.1 provides an example.

Syntax Augmented Machine Translation Labels

Dependency parses provide labels for only a small set of spans. To compare all HATs, all translation admissible spans should be labelled. Zollmann and Venugopal (2006) proposed a method for extending a basic label set. Given a set of basic labels $\mathcal{L}$, every span (i, j) is assigned a label, according to the following protocol described in Algorithm 6.2.1.

In other words, every span is assigned either a basic label, a label that is a concatenation of two labels, or a 'minus-label' of the form $B\backslash A$ or A/B, indicating that the span is of form A, missing an span of form B at the left or right, respectively. Figure 6.2 provides an example.

In this sentence, only the span covering words 1 to 4 does not have a label. A test on the first 10,000 sentences of our datasets showed that on average less

Algorithm 2 SAMT labels

if $\exists L \in \mathcal{L}$ for (i, j) **then**
  return L
else if $\exists A, B \in \mathcal{L}$ s.t. $\mathcal{L}(i, k) = A$ and $\mathcal{L}(k, j) = B$ **then**
  return $A + B$
else if $\exists A, B \in \mathcal{L}$ s.t. $\mathcal{L}(i, k) = A$ and $\mathcal{L}(j, k) = B$ **then**
  return A/B
else if $\exists A, B \in \mathcal{L}$ s.t. $\mathcal{L}(k, j) = A$ and $\mathcal{L}(k, i) = B$ **then**
  return $A \backslash B$
else if $\exists A, B, C \in \mathcal{L}$ s.t.$\mathcal{L}(i, m) = A$, $\mathcal{L}(m, n) = B$ and $\mathcal{L}(n, j) = C$ **then**
  return $A + B + C$
else
  return FAIL
end if

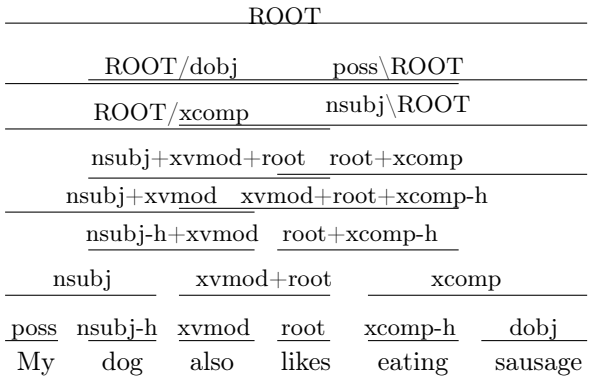


Figure 6.2: SAMT labels

than 60% of the translation admissible spans in the 4 automatically aligned datasets were assigned a label different from FAIL (see Table 6.1).

Language pair	**Spans total**	**Spans labelled SAMT**	**% Labelled**
English-Dutch	1078307	598583	0.56
English-French	1260720	644710	0.51
English-German	930273	540968	0.58
English-Chinese	609401	375765	0.62

Table 6.1: Success rate of SAMT labels

Extended Labelling

For the purpose of detecting similarities, the percentage of labelled spans should be as high as possible. We propose the following method for labelling *all* spans, based on dependency labels, in which we use the basic operations /, + and \ as before. Algorithm 3 describes how to label all spans.

Algorithm 3 Label all spans

Return a label for span (i, j), given a set of basic labels $\mathcal{L}$
if $\exists L \in \mathcal{L}$ for (i, j) **then**
  return L
else
  n = 2
  while 1 **do**
    if $\exists A_1, \ldots, A_n \in \mathcal{L}$ s.t. $A_1 + \cdots + A_n$ is a label for (i, j) **then**
      return $A_1 + \cdots + A_n$
    else if $\exists A_1, \ldots, A_n \in \mathcal{L}$ s.t.$A_1/A_2 + \cdots + A_n$ is a label for (i, j) **then**
      return $A_1/A_2 + \cdots + A_n$
    else if $\exists A_1, \ldots, A_n \in \mathcal{L}$ s.t. $A_1 + \cdots + A_{n-1} \backslash A_n$ is a label for (i, j) **then**
      return $A_1 + \cdots A_{n-1} \backslash A_n$
    else if $\exists A_1, \ldots, A_n \in \mathcal{L}$ s.t. $A_1 + \cdots + A_{k-1} \backslash A_k / A_{k+1} + \cdots + A_n$ is a label for (i, j) **then**
      return $A_1 + \cdots + A_{k-1} \backslash A_k / A_{k+1} + \cdots + A_n$
    else
      n ← n+1
    end if
  end while
end if

For the previously unlabelled span covering word 1 to 5, Algorithm 3 finds the label '$poss \backslash ROOT / dobj$'.

6.2.2 Learning the grammar

With all spans labelled, we can design a learning algorithm that learns a PCFG predicting the HATforest.[1] However, to learn a grammar from the data, a probability distribution over the trees in the corpus is required. If such a distribution was known, relative frequency estimation could be used to determine the PCFG that generated it. On the other hand, if the grammar was known, the probability distribution over the HATs would be easy to compute. This chicken and egg problem should seem familiar to statistical modellers. Fortunately, there is an algorithm that addresses the situation of incomplete data: the expectation

[1] For computational reasons, we have chosen for a PCFG, but similar algorithms can of course be devised to train, e.g., a grammar incorporating larger fragments.

maximization (EM) algorithm[2] (Dempster et al., 1977), that iteratively learns a model from the data, and estimates the data from the model. The EM algorithm is known to converge to a (local) optimum.

EM for HATs

In the case of learning a grammar from a set of HATs whose probability distribution is known, the EM algorithm has the following steps:

1. Initialize the model. Typically, this could be done by either distributing the probability mass uniformly over all rules (with the same left-hand side), or by distributing the probability mass uniformly over all HATs (for one sentence), and read of the model by relative frequency estimation

2. For every sentence, compute the probability distribution over its HATs, normalise such that the HAT probabilities sum up to 1 (expectation step).

3. Use relative frequency estimation to learn a new PCFG from the data (maximization step).

4. Iterate 2 and 3 until convergence.

Computation

Of course, naively following the protocol described in the previous subsection would not be computationally feasible. As the HATs are implicitly represented as CFGs, explicitly computing the probability of all HATs would require unpacking the entire HATforest, which grows exponentially with the length of the sentence. This would not only be time and space consuming, but would also require a lot of redundant computing, as the probability of subtrees that occur in multiple HATs need to be recomputed many times.

In this subsection, we propose an algorithm for computing the updates for one EM iteration without unpacking the HATforest, that focusses on reusing previously computed values and using only a limited amount of memory at a certain time. The algorithm describes how to compute the updates (i.e., the relative frequency counts) of the HAT grammar for one sentence, given the grammar that was computed in the previous EM iteration. It is assumed that the HAT grammars for all sentences are stored, and can be requested. The following abbreviations are used:

1. $P(A)$ is the sum of the probabilities of all subtrees of HATs headed by the node labelled A.

2. $P(A_{x_1 \cdots x_n})$ is the sum of the probabilities of all subtrees of HATs headed by $A(x_1 \cdots x_n)$

[2]This same algorithm was used in the IBM models to determine word-alignments of parallel corpora.

This probability function, that is used in computing the updates, can be pre-computed in a top-down fashion, by calling the function described in Algorithm 4 on the topnode of the HATforest.

Algorithm 4 $prob(\mathcal{P}, \mathcal{H}, \mathcal{G}, X, (y_1, \cdots, y_n) = None)$

Input: a dictionary with already computed probabilities $\mathcal{P}$, a HATgrammar $\mathcal{H}$, a global PCFG $\mathcal{G}$, a node X in $\mathcal{H}$, and optionally a set of children in $(y_1 \cdots y_n)$ such that $X \rightarrow y_1 \ \cdots y_n$ is a rule in $\mathcal{H}$

if $X \in \mathcal{P}$ **then**
  return $\mathcal{P}(X)$

else if $\neg\exists X \rightarrow C_1 \ \cdots \ C_N \in \mathcal{H}$ **then**
  return $\mathcal{P}(X) = 1$

else if $(y_1 \cdots y_n)!$ =None **then**
  return $\mathcal{P}(X_{y_1 \cdots y_n}) = \mathcal{G}(X \rightarrow y_1 \ \cdots y_n) \cdot \prod_n prob(\mathcal{H}, \mathcal{G}, y_n)$

else
  return $\sum_{X \rightarrow y_1 \ \cdots \ y_n \in \mathcal{H}} \mathcal{P}(X_{y_1 \ \cdots y_n})$

end if

To compute the updated count of a rule, one must know in which HATs the rule occurred, and what the probability of these HATs was. This information will be obtained by going through the HAT grammar in a top-down fashion, and computing the (normalised) probability mass of all HATs the rule occurred in, using the probability mass of all subtrees headed by its parent, and the relative probability mass of all subtrees headed by the expansion with respect to other expansions. The update process starts by calling the update function described in Algorithm 5 on the topnode of the HATforest. New function calls will be made while computing the probabilities of the productions with this topnode as left-hand side. Counts for productions can be updated multiple times for each rule. Note that it is assued that the algorithm does not return anything, but merely updates a global dictionary with counts for rules. Furthermore, it is assumed that the nodes of the HATs are uniquely defined by their label. The overall complexity of computing the update for a sentence is polynomial in its length (compute precise).

The above mentioned algorithms can be used to train a grammar that assigns probabilities to the maximally recursive structures in the corpus.

6.3 In conclusion

In this thesis, we have studied compositional translation on an empirical level. We have elaborately discussed several aspects of compositional translation, and

Algorithm 5 $update(N, \mathcal{H}, \mathcal{G}, \mathcal{P}, \mathcal{C}, p_{total})$

Input: a HAtforest $\mathcal{H}$, a global grammar $\mathcal{G}$, a function $\mathcal{P}$, the current counts $\mathcal{C}$, a node N and a number p_{total} describing the probability mass that is assigned to the parent N

if $\neg\exists N \rightarrow X_1 \ \cdots X_n \in \mathcal{H}$ **then**
  return
end if
for $R = N \rightarrow X_1 \ \cdots X_n \in \mathcal{H}$ **do**
  $C_{new} = p_{total} \cdot \frac{\mathcal{P}(N_{X_1 \cdots X_n})}{\mathcal{P}(N)}$
  $\mathcal{C}(N \rightarrow X_1 \cdots X_n) \leftarrow \mathcal{C}(N \rightarrow X_1 \cdots X_n) + C_{new}$
  for $X \in \{X_1, \cdots X_n\}$ **do**
    $update(X, \mathcal{H}, \mathcal{G}, \mathcal{P}, \mathcal{C}, C_{new}))$
  end for
end for

presented a careful analysis of the usefulness of dependency parses for constructing a compositional translation system. Although a conclusive answer to the most general question - can dependency parses be used to construct a bilingual compositional grammar - was not yet found, we believe a useful foundation was created to answer this question. This thesis provides insights to the empirical investigation of compositionality in general, a thorough investigation of dependency parses in the light of compositional translation, a worked out proposal to extend this research, and package of comprehensible and well documented classes that can be used to conduct this, and further research in the same category.

Although we believe that in the nearby future, developing systems that can automatically produce high quality translations from one natural language will remain a difficult challenge for both the academic and business world, we hope that this work can eventually contribute to being able to proceed to decode.

Appendix A

Examples from Manual Analysis

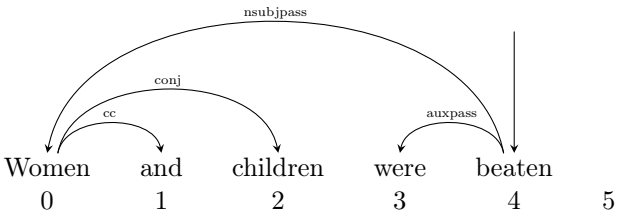
This appendix contains a subset of the manually analysed alignments.

A.1 English-German

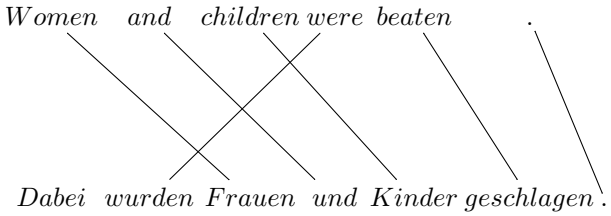
A.1.1 Maximal Compositionality

Part of English sentence: Women and children were beaten .
German Translation: Dabei wurden Frauen und Kinder geschlagen .

Dependency Parse:



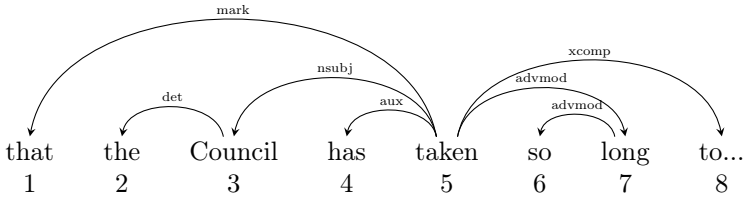
Alignment:



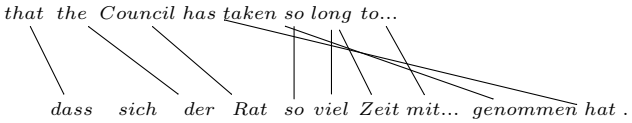
Reason non optimal score:
The score of this sentence is 0.5, which is caused by the fact that the dependency tree is much flatter than a maximally recursive tree. The order in which the arguments in the dependency tree are combined is not respected by a maximally recursive tree, whereby both the dependency relation between 'were' and 'beaten' and 'woman and children' and 'beaten'.

English phrase: that the Council has taken so long to...
German translation: dass sich der Rat so viel Zeit mit ... genommen hat

Dependency Parse:



Alignment:

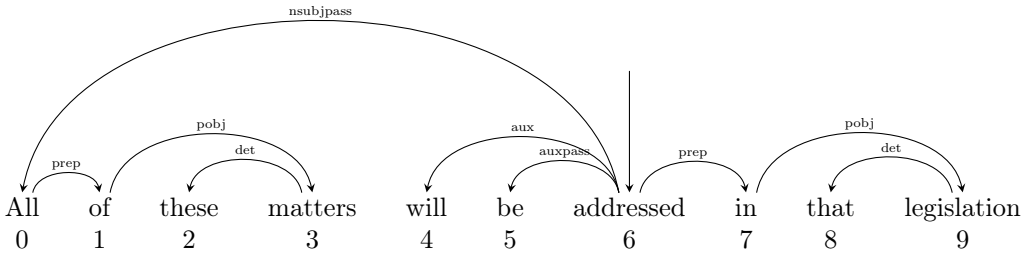


Reason non optimal score:
The arguments of 'taken' in the dependency parse are reordered in German in a fashion that can not be accounted for by a maximally compositional tree: none of the HATs will obtain the score for the dependency relations ('taken','so long'), or ('taken', 'to...').

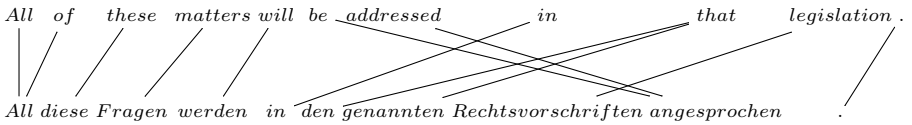
A.1.2 Phrasal Translations beyond the Dependency Level

Part of English sentence: All of these matters will be addressed in that legislation .
German Translation: All diese Fragen werden in den genannten Rechtsvorschriften angesprochen .

Dependency Parse:



Alignment:



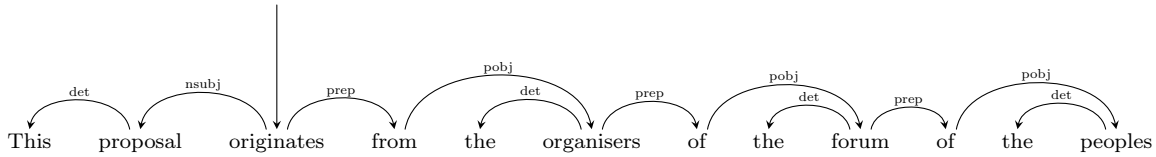
Reason non optimal score:
The score of this sentence is 0.77. The non optimal score is caused by the phrasal translation of 'All of' into 'all', such that the dependency relations between 'All' and 'of these matters', and between 'of' and 'these matters' are not respected.

A.2 English-French

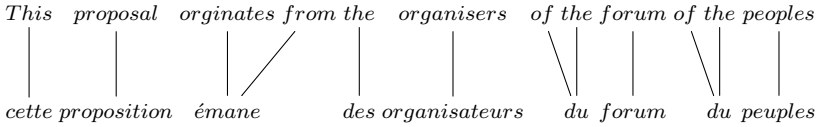
A.2.1 Phrasal Translations beyond the Dependency Level

Part of English sentence: This proposal orignates from the organisers of the forum of the people .
French Translation: cette proposition mane des organisateurs du forum des peuples

Dependency Parse:



Alignment:

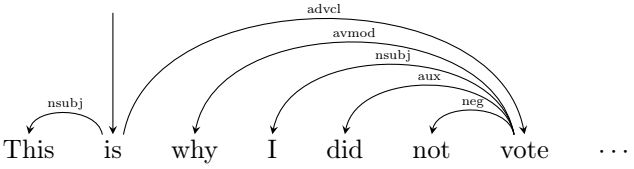


Reason non optimal score:
The score of this sentence is 0.45. All deviations from the dependency parse are caused by phasal translations that go beyond the dependency level.
Firstly, the translation of 'originates from' into 'mane', prevents the relations ('originates','from') and ('from', 'organisers') from being preserved. Then, additionally phrasal translations of prepositions and articles into one word cause an additional 4 relations to break during maximally compositionally translation.

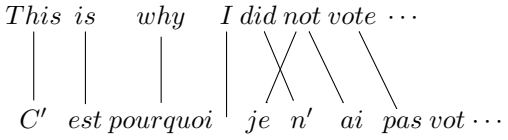
A.2.2 Negation

Part of English sentence: This is why I did not vote $\cdots$.
French Translation: C'est pourquoi je n'ai pas vot $\cdots$

Dependency Parse:



Alignment:



Reason non optimal score:
The score of this sentence is 0.83. The score is close to optimal, but the translation of the negating word 'not' into a non-contiguous French sequence hinders the preservation of the relations ('vote','not') and ('vote', did').

Appendix B

Implementation

The module that was developed to realise the experiments in this thesis has been made available online on `https://github.com/dieuwkehupkes/Thesis`. The module uses the external toolkit NLTK, that can be downloaded from `http://nltk.org/`. For users without root access, a folder with the source code of the relevant NLTK classes is included in the git-project. If NLTK is installed on your computer, this folder may be deleted.

All classes contain a demo that demonstrates the basic workings of the class. The demonstrations of a class can be called by running the file containing the class as main. Extensive documentation of all classes, that should suffice for both using and extending the current functionality, is provided in this appendix. In addition to the module, two programs are provided: AlignmentScorer.py - which can be used to score files with alignments according to the consistency of their HATs with provided dependency files - and HATgenerator.py - which can be used to compute and store the HATforest of a file with alignments and sentences. Both files contain a help function that specifies how they can be ran.

CONTENTS

1 alignments Module 2

2 dependencies Module 7

3 dependencies Module 8

4 file_processing Module 12

5 labelling Module 16

6 dependencies Module 18

7 scoring Module 20

8 tests Module 22

9 Indices and tables 27

CHAPTER

ONE

ALIGNMENTS MODULE

Module for processing alignments. This module contains three classes. Running alignments.py will give a demonstration of the functionality of the classes.

class alignments.**Alignments**(*links*, *source*, *target=''*)

A class that represents alignments. Important methods in this class compute the monolingual source phrases according to this alignment, generate all rules or all maximally recursive rules that are associated with the alignment and generate a dictionary representing all trees that are generated with this rules in a compact fashion.

Construct a new alignment object with the alignment links given by 'links', the source sentence given by 'source', and possibly a target sentence 'target'. Construct a set-representation of the alignment, and compute a lexical dictionary describing the spans of the words.

Parameters

- **links** – a string of the form '0-1 2-1 ...' representing the alignment links. Word numbering starts at 0.
- **source** – A string representing the source sentence.
- **target** – A string representing the target sentence.

Class is initialized with a string representation of the alignment that starts counting at 0 of the form '0-1 2-1 ...' and the sentence the alignment represents. During initialisation, a set-representation of the alignment is created.

HAT_dict(*labels={}*)

Transform all HATrules into a dictionary that memory efficiently represents the entire forest of HATs. As a HAT_dict uniquely represents a HATforest, the labels of all spans should be unique avoid ambiguity.

Parameters **labels** – A dictionary assigning labels to spans.

Returns A dictionary that represents the HATforest, by describing for every allowed span what is al-

lowed expansions are. Entries are of the form {lhs: [(rhs_11,...,rhs_1m),...,(rhs_n1,...,rhs_nk)]

_links_fromE ( )
Precompute values for the function E_c(j) = **|{(i',j')\in A | j' =< j}|**.

_links_fromF ( )
Precompute values for the function F_c(j) = **|{(i',j')\in A | i' =< i}|**.

_maxspan (*(x, y)*)
Returns the maximum position on the target side that are linked to positions [x,y].

_minspan (*(x, y)*)
Returns the minimum position on the target side that are linked to positions [x,y].

agreement (*tree*)
Output what percentage of the nodes of an inputted tree are consistent with the alignment. :param tree: An nltk tree object. :return: a float that describes the percentage of the nodes

of tree that were nodes according to the alignment.

compute_phrases ( )
Return a list with all source phrases of the alignment. Similar to Alignments.spans, but returns a list rather than a generator, and does not include all one-length units.

Returns A list with all valid source phrases

consistent_labels (*labels*, *label_dict*)
Measures the consistency of the alignment with a dictionary that assigns labels to spans. Outputs a dictionary with labels, how often they occurred in the input set and how often they were preserved. Ignore word-labels from dep-parse that end in -h.

hat_rules (*prob_function*, *args*=[ ], *labels={}*)
Return a generator with all rules of a PCFG uniquely generating all *hierarchical* alignment trees. The rules are assigned probabilities using the input probability function, args are the arguments of these function.

The rules are computed by transforming the alignment into a set of Node objects and links between them that together constitute a graph whose edges correspond to valid spans and partial sets, and using the shortest_path function of the Node class.

Parameters

- **prob_function** – A probability function from the Rule class, according to which probabilities should be assigned.
- **args** – The arguments the probability function needs.

- **labels** (*A dictionary assigning labels to spans.*) – The labels that should be assigned to the spans.

lex_dict()

Use self.sentence to create a lexical dictionary that assigns lexical items to spans. :return: A dictionary {(0,1) : word1,..,(n-1,n): wordn}

lexrules(*labels={}*)

Returns an generator with the terminal rules of the grammar. (i.e., the 'lexicon', that tells you the span corresponding to a word). If labels are provided for the spans, the rules will be labelled accordingly.

make_set(*alignment*)

Return a set with all alignment links, and keep track of the length of source and target sentence. Output a warning when alignment and sentence do not have the same number of words.

Parameters **alignment** – A string representing the alignment, as was passed during initialisation

percentage_labelled(*labels*)

Output which percentage of the spans in the alignment are labelled by the set of inputted labels. :return: total, labelled

prune_production(*rule*, *lex_dict*)

Replace all leafnodes that do not constitute a valid source phrase with the lexical item the leafnode dominates.

Parameters **lex_dict** – a dictionary with spans as keys, and the corresponding lexical items as values.

Returns a Rule object in which all nodes are either valid source spans or lexical items.

rules(*prob_function*, *args*, *labels={}*)

Returns a generator with all rules of a PCFG uniquely generating all alignment trees. Rule probabilities are assigned according to specified input probability function, args should contain a list of arguments for this function.

The rules are computed by transforming the alignment into a graph whose edges correspond to valid spans and partial sets and using the path function of the Node class. This function is not as extensively tested as the hat_rules function, as it is rarely used for computational issues.

spans()

Return all a generator with all valid source side spans that are part of a phrase pair, and all one-length units that are necessarily part of a tree describing the translation. Contrary to the convention, also unaligned sequences of words are allowed as spans.

Spans are computed using the first shift-reduce algorithm presented in Chang & Gildea (2006). This is not the most efficient algorithm to compute phrase pairs.

texstring()
Generate latexcode that generates a visual representation of the alignment.

alignments.**HAT_demo**()
Demonstration 3. Simple one to many alignment, HATfunctionality.

class alignments.**Node**(*value*)
Defines a node in a directed graph. You can add edges from this node to other nodes by using link_to. The paths_to method calculates all paths from this node to the given node. The Node class is used to represent alignments as graphs.

Initializes a new node; value can be used for representation

link_to(*node*)
Add a directed edge from this node to the given node. :type node: A Node object.

paths_to(*node*)
Returns a generator that calculates all paths to the given node. These paths are calculated recursively. :type node: a Node object

remove_link(*node*)
Remove the edge to this node, if present :type node: A Node object.

shortest_paths_to(*node*)
Finds all shortest paths from current node to node using an adapted Dijkstra algorithm starting from the end. The function also stores paths that can be used for later(i.e paths longer than 1 from self to intermediate nodes). :type node: A Node object.

class alignments.**Rule**(*root*, *path*, *labels={}*)
Defines a rule from one span to a set of consecutive spans which union forms it.

A string representation is provided for convenient displaying.

Initialize a new rule as its root span and the path in the graph (consisting of an array of nodes) that it produces. Labels can be provided to annotate the spans of a rule. :param root: The rootspan of the node. :type path: A Waypoint. :type labels: A dictionary assigning labels to spans.

_lhs(*labels*)
Create the left hand sides of the rule and set as an attribute.

_rhs(*labels*)
Create the right hand sight of the rule and set as attribute.

_str(*rhs*)

lhs()
Return the left hand side of the rule. :type return: nltk.Nonterminal object.

probability_labels(*labels*)
Compute the probability of a rule according to how many of the nodes it generates can be labelled according to a set of given labels. :param labels: A list containing a dictionary that

assigns labels to spans.

probability_spanrels(*span_relations*)
Compute the probability of a rule according to how many span_relations it makes true. :param span_relations: A list containing a dictionary

which describes which spanrelations are desired.

rank()
Return the rank of a rule.

rhs()
Return the right hand side of the rule. :type return: a tuple with nltk.Nonterminal objects

uniform_probability(*args*=[])
Set probability to 1.

class alignments.**Waypoint**(*node*, *link*=*None*)
Defines a waypoint in a one-directional path. The class Waypoint is used in the representation of paths. Multiple paths can be saved more memory efficient as path arrays can be shared between paths. As they contain link to other Waypoints, Waypoints can represent paths as linked lists.

Create a waypoint object.

Parameters

- **node** (*A Node Object.*) – The node it represents.
- **link** (*A Waypoint object.*) – A link to the next waypoint.

alignments.**demo_basic**()
Demonstration 1, basic monotone one-to-one alignment.

alignments.**demo_basic2**()
Demonstration 2. Simple one-to-many alignment.

alignments.**demos**()

CHAPTER

TWO

CONSTITUENCIES MODULE

class constituencies.**ConstituencyTree**(*tree*, *sentence=False*)

A class representing a constuency tree. The classes uses the nltk class nltk.Tree, but adds some functionality that is useful with respects to alignments.

Create a ConstituencyTree object.

Parameters

- **tree** (*str or nltk.Tree object.*) – A constituency tree.
- **sentence** – The sentence that the leafnodes of the tree constitute.

branching_factor(*branching_dict={}*)

Return a dictionary that summaries the different branching factors of the trees. If initialised with a dictionary, update this dictionary with the valeus of the current tree.

nr_of_nonterminals()

Return the number of nonterminals in self.tree.

phrases_consistent(*subtree*, *startpos*, *phrase_list*)

Return the number of non-terminal nodes in the tree that occur in the provided list of phrases. :param subtree: A subtree of self.tree. :param startpos: The left-most word position the subtree dominates. :param phrase_list: A list of allowed phrases. :return: the number of nodes in the tree that is in phrase_list.

reconstruct_sentence()

Reconstruct the sentence from the leaf nodes of the tree. If a sentence was passed in initialisation, return this sentence. :type return: str

root_span(*subtree*, *startpos*)

Recursively compute the span a node covers :param subtree: a subtree of self.tree :param startpos: the first position the subtree dominates

constituencies.**demo**()

CHAPTER

THREE

DEPENDENCIES MODULE

class dependencies.**Dependencies**(*dependency_list*, *sentence=False*)

A class representing the dependencies of a sentence in a dictionary. The dependencies are created from a list with dependencies formatted like the Stanford dependency parses.

Create a Dependencies object, based on the passed dependency list. If no sentence is passed, the sentence is reconstructed from the dependency list, leaving gaps for items that were not included.

Parameters **dependency_list** – A list with dependencies of the form reltype(head-pos_head, dependent-pos_dependent) min(pos-dependent) = 1

Returns A dictionaries with entries of the form pos_head: [pos_dependent, reltype]

POStag(*word*)

Find a postag for a word.

SAMT_labels()

Create SAMT-style labels based on the basic dependency labels created in dependency_labels. The order if precedence is as follows:

- Basic labels
- labels A + B, where A and B are basic labels
- labels A/B or AB where A and B are basic labels
- labels A + B + C where A,B and C are basic labels

annotate_span(*labels*)

Annotate labels with their span, to make the grammar unique.

argument_list(*head_span*)

return a list with spans of the head and its arguments

branching_factor(*b_dict={}*)

Compute the branching factor of all nodes in the dependency tree. If an

input dictionary is given, update the branching factors in the dictionary with the newly found branching factors.

checkroot()
Check if dependencies form a tree by checking coverage of the rootnote.

comp_score()
Returns the percentage of words that is head of another word, thereby giving a measure of the level of compositionality of the parse

dependency_labels()
Produces standard labels for spans according to the following scheme:

- label[(i,i+1)] = HEAD iff word i+1 is the head of the sentence
- label[(i,j+1)] = rel iff there is a dependency relation rel(x, y) and wordspan(y) = (i,j+1)
- label[(i,i+1)] = rel-head iff there is a dependency relation rel(x,i+1) and word i+1 was not labelled by one of the previous conditions

find_dependent(*relation*)
Find the depending word of a dependency relation using a regular expression.

find_dependent_pos(*relation*)
Find the position of the dependent of a dependency relation using a regular expression.

find_head(*relation*)
Find the head word of a dependency relation using a regular expression.

find_head_pos(*relation*)
Find the position of the head of a dependency relation using a regular expression.

find_relationtype(*relation*)
Find the type of a dependency relation using a regular expression.

get_comp_spanrels()
Create a dictionary of dependencies between word positions and word spans. Go through the dependency dictionary, but select only the relations that display compositionality (i.e. no relations between words)

get_span(*key*)
Recursively compute the span of a word. The span of a word is constituted by the minimum and maximum position that can be reached from the word by following the directed dependency arrows. The spans are left exclusive and right inclusive. I.e. if positions i and j are the minimum and maximum positions that can be reached from a word, its span will be [i-1,j]. Every word necessarily spans itself, a word at position i without dependents will

thus have span [i-1,i]. The dependency from root to head of the sentence is not considered.

label_all()
Label all spans of the sentence.

labels(*label_type='basic'*)
Return labels of given type.

Parameters **type** – describes which label_type should be used. Options: all, basic or SAMT.

The default labeltype is basic.

percentage_SAMT()
Compute how many spans were labelled by an SAMT label. :return: number of spans, number of labelled spans

print_deps()
Displaying function. Print all the dependency relations in the dependency parse.

print_labels(*labels*)
Print out the contents of a dictionary in a nice format.

print_spans()
Displaying function. Print all word_spans of of the dependency parse.

reconstruct_sentence(*sentence*)
Reconstruct the sentence corresponding to the dependency parse. :return: a list with the words of the sentence.

set_dependencies(*dependency_list*)
Read in a file and create a dictionary with its dependencies using regular expressions.

set_wordspans()
Compute the span of each word and store it in a dictionary with word positions and a tuple that represents their span as key and value, respectively.

spanrelations(*rightbranching=False*, *leftbranching=False*, *interpunction=True*)
Create a dictionary with spanrelations that are 'deeper' than the standard relations in the dependency parse, such that stepwise combining head and arguments is allowed. Parameters rightbranching, leftbranching and interpunction describe how exactly arguments and heads are allowed to combine.

Parameters

- **rightbranching** – allow an argument to combine with the arguments one by one, giving preference to arguments to the right.

- **leftbranching** – allow an arguments to combine with the head one by one, giving preference to arguments to the left.
- **interpunction** – Take gaps in the dependency parse into account, by adding extra relations in which the gap is already combined with one of its left or right adjacent units.

If both left- and rightbranching are true, all combination orders are allowed.

textree()
Print string that will generate a dependency tree in pdf with package tikz-dependency.

update_labels(*label_dict*)
Update an inputted dictionary with the labels from dependency object.

dependencies.**demo**()
A demonstration of how the Dependencies class can be used.

CHAPTER

FOUR

FILE_PROCESSING MODULE

File_processing is a module for processing sentences, alignments and treestructures. It brings together the functions from the other classes, enabling the user to apply the functions using information from three files containing alignments, sentences and parses. Explain the different possibilities of the class.

class file_processing.**ProcessConstituencies**(*alignmentfile*, *sentencefile*, *treefile*, *targetfile=False*)

Bases: file_processing.ProcessFiles

Subclass adapted for constituencies

During initialization the files are loaded for reading. Allows to leaf empty one of more files if they are not needed for functions that will be used

all_rules(*max_length=40*)

branching_factor(*max_length=40*)

Compute the average branching factor of all head nodes of the dependency parses or the corpus. Can be restricted to a sentence length.

consistent_labels(*label_type*, *max_length=40*)

Determines the consistency of a set of alignments with a type of labels over the entire corpus.

relation_count(*max_length*)

Counts occurences of all labels in the constituent parse.

score_all_sentences(*rule_function*, *probability_function*, *prob_function_args*, *label_args*, *max_length=40*, *scorefile=''*, *treefile=''*)

texstring(*new*)

Output a texstring with the alignment, the constituency tree and the alignment.

class file_processing.**ProcessDependencies**(*alignmentfile*, *sentencefile*, *treefile*, *targetfile=False*)

Bases: file_processing.ProcessFiles

Subclass of ProcessFiles that is focussed on the specific occasion in which trees are dependencies.

During initialization the files are loaded for reading. Allows to leaf empty one of more files if they are not needed for functions that will be used

all_HATs(*file_name*, *max_length=40*)
Compute all HATs grammars, represent them as a dictionary and and pickle [sentence_nr, rootlabel, HATdict] to a file with the provided name file_name.

branching_factor(*max_length*)
Compute the average branching factor of all head nodes of the dependency parses or the corpus. Can be restricted to a sentence length.

check_consistency(*sentence*, *dep_list*)
Check whether a list with dependencies is consistent with a sentence, by comparing the words. Some flexibility is allowed, to account for words that are spelled differently. Return True if the dependency parse contains no more than 3 words not present in the sentence and False otherwise.

close_all()
Close all input files.

consistent_labels(*label_type*, *max_length=40*)
Determines the consistency of a set of alignments with a type of labels over the entire corpus.

percentage_labelled(*max_length*, *label_type*)
Compute the percentage of the spans in the dictionary that is labelled by a labelling method

print_dict(*dictionary*, *filename*)
Print the contents of a dictionary to a file.

relation_count(*max_length*)
Counts occurrences of all relations in dependency parses of sentences shorter than max_length.

relation_percentage(*all_relations*, *relations_present*)

sample(*samplesize*, *maxlength=False*, *display=False*)
Create a sample of sentence from the inputted files. Create a file with the sentences, and files with the matching alignments, dependencies and target sentences. If display = True, create a texfile that can be ran to give a visual representation of the selected sentences. Return an array with the list of sentence numbers that were selected.

score_all_sentences(*rule_function*, *probability_function*, *prob_function_args*, *label_args*, *max_length=40*, *scorefile=False*, *treefile=False*)

tex_preamble()
Print the pre-amble of a tex document in which both dependency parses and alignments are printed.

texstring(*new*)
Output a texstring with the alignment, the dependency and the ew = alignment, sentence, dep

transform_contents(*value*)
Return a suitable string representation of input

class file_processing.**ProcessFiles**(*alignmentfile*, *sentencefile*, *treefile*, *targetfile=False*)
Brings together all functions by enabling the user to apply functions from the other classes to files containing alignments, sentences and dependency parses.

During initialization the files are loaded for reading. Allows to leaf empty one of more files if they are not needed for functions that will be used

close_all()
Close all input files.

consistent_labels(*label_type*, *max_length=40*)
Determines the consistency of a set of alignments with a type of labels over the entire corpus.

evaluate_grammar(*grammar*, *max_length*, *scoref*)
Parse the corpus with inputted grammar and evaluate how well the resulting parses cohere with the alignments.

next()
Return the next alignment, sentence and tree_list. If the end of one of the files is reached, return False.

next_sentence()
Return the next sentence. If the end of the file is reached, return None.

print_dict(*dictionary*, *filename*)
Print the contents of a dictionary to a file.

print_function(*to_print*, *filename*)

relation_count(*max_length*)
Counts occurrences of all relations in dependency parses of sentences shorter than max_length.

relation_percentage(*all_relations*, *relations_present*)

score_all_sentences(*rule_function*, *probability_function*, *prob_function_args*, *label_args*, *max_length=40*, *scorefile=''*, *treefile=''*)

Not implemented in general class, use from more specific subclasses. If not present, raise not implemented error.

transform_contents(*value*)

Return a suitable string representation of input

CHAPTER

FIVE

LABELLING MODULE

class labelling.**Labels**(*labels*)

Class to create different kinds of labels from a set of basic labels. Currently, some set of labels are computed multiple times to be used in the computation of other labels. Class can be made more efficient by storing such sets of labels as attributes of the Labels instance.

Create a labelling object, with basic labels. :param labels: A dictionary assigning labels to spans

SAMT_labels()

Return all SAMT labels based on the basic labels of the object. The order if precedence is as follows:

- Basic labels
- labels A + B, where A and B are basic labels;
- labels A/B or A
  B where A and B are basic labels;
- labels A + B + C where A,B and C are basic labels;

annotate_span(*labels*)

Annotate labels with their span, to make the grammar unique.

concat(*depth*, *o={}*, *i=None*)

Compute all concatenated labels up to inputted depth, with basic labels i. If an output dictionary o is passed, extend this dictionary with the found spans that do not yet have a label in o.

Parameters

- **depth** – The maximum number of variables in the labels.
- **o** – An output dictionary with already existing labels.
- **i** – A dictionary with basic labels to be concatenated.

label_complexity(*label*)

Return the number of variables in a label.

label_most()

Label all spans within the range of labels, following the following rules:

- Labels with a lower depth are always preferred;
- Concatenated labels are preferred over minus and double/minus labels;
- Single minus labels (with concatenated spans) are preferred over double minus labels.

SAMT labels thus have precedence over other labels.

minus(*depth*, *o={}*, *i=None*)

Compute all labels of the form A
B and B/A where B is a basic label, and A a concatenated label that contains no more than depth-1 variables. If an output dictionary o is passed, extend this dictionary with the found spans that do not yet have a label in o.

Parameters

- **depth** – The maximum number of variables in the labels.
- **o** – An output dictionary that is to be updated with the new labels
- **i** – A dictionary with basic labels.

minus_double(*depth*, *o={}*, *i=None*)

Compute all labels of the form A
B/C, where A is a basic label, and B and C are concatenated labels. The outputted labels have a number of variables that is no higher than depth. If an output dictionary o is passed, extend this dictionary with the found spans that do not yet have a label in o.

Parameters

- **depth** – The maximum number of variables in the labels.
- **o** – An output dictionary that is to be updated with the new labels
- **i** – A dictionary with basic labels.

Return a dictionary with all labels of the form , where B is in self.labels or in i if i is provided, and A and C are in A1 + A2 + An where A1.. An are in self.labels or i and n <= depth.

labelling.**demo**()

CHAPTER

SIX

HAT PROCESSING MODULE

class process_hats.**HATGrammar**(*HATdict*, *root*)

Class that

Initialise with a dictionary uniquely representing a HAT

plain_label(*label*)

strip the label from the part determining its span, to make it uniform

probmass(*head_node*, *children=()*, *external_pcfg={}*, *probs={}*)

Compute the probability mass of all subtrees headed by head_node with direct children children (possibly empty), given the input pcfg.

to_WeightedGrammar(*rule_dict*, *remove_old=False*)

Transforms a set of rules represented in a nested dictionary into a WeightedGrammar object. It is assumed that the startsymbol of the grammar is TOP, if this is not the case, parsing with the grammar is not possible. If remove_old = True, remove the old grammar during the process to save memory.

update(*external_pcfg*, *probs*, *grammar*, *p_cur*, *lhs*)

Compute the updated counts for a node, given its parent and how often this parent occurred in the forest. Does not return a grammar, but modifies it globally.

update_weights(*grammar*, *external_pcfg={}*)

Implicitly assign all HATs in the HATforest a probability, normalise, and compute the counts of the rules in them through relative frequency estimation. Update the inputted grammar with these counts.

class process_hats.**ProcessHATs**(*HATfile*)

Class with functions that can be applied to a file containing pickled precomputed HATs. ProcessHATs has functional overlap with the class FileProcessing, but is more efficient as it avoids recomputing HATforests.

Pass the name of the file containing the pickled HATs.

em(*max_iter*)

When passing a grammar represented by a dictionary, iteratively assign probabilities to all HATs of the corpus and recompute the counts of the grammar with relative frequency estimation until convergence or until a maximum number iterations is reached. Return the new grammar :param start_grammar Grammar represented as a nested dictionary :param max_iter Maximum number of iterations :param max_length Maximum sentence length considered

em_iteration(*old_grammar*, *new_grammar*)

Assign probabilities to all HATs in the corpus with the current grammar, recompute probabilities and return the new grammar. It is assumed that the HATs are precomputed and pickled into a file in the correct order. Every sentence under max_length should be represented in the file as: [sentence_nr, HAT_dict, root].

initialise_grammar()

Initialise a grammar based on all HATs in the corpus

next()

Return the next item in the file. If no :return [sentence_nr, HATdict, root]

normalise(*rule_dict*)

Given a nested dictionary that represent rules as follows: {lhs : {rhs1 : count, rhs2: count ...}, ...}, return a similar nested dictionary with normalised counts

unique_rules(*stepsize*)

Go through HATcorpus and keep track of the percentage of the rules that is unique. Store the number of rules and the number of unique rules if the number of HATs processed % stepsize is 0

CHAPTER

SEVEN

SCORING MODULE

class scoring.**Scoring**(*alignment*, *sentence*, *labels={}*)

Class that provides methods for scoring an alignment according to a set of preferred relations. The corresponding tree is created, span labels can be entered to label the nodes in the tree.

During initialization an alignment, a corresponding sentence and a string with dependencies are passed. A weighted CFG generating all HATs is created, the rules are assigned 'probabilities' according to preferred_relations or labels. The adapted Viterbi parser from the NLTK toolkit is used to parse the sentence and obtain the score.

grammar(*rules*)

Return a weighted grammar (NLTK-style) and its rank given a generator object with all rules.

grammar_rank(*rules*)

Determine the maximum rank of a set of rules.

list_productions(*rules*)

make_lexdict()

Create a dictionary assigning words to spans.

parse(*grammar*, *trace=0*)

Parse the sentence with the given grammar using the Viterbi parser from the NLTK. Return the best parse and its score.

score(*rule_function*, *prob_function*, *args*, *trace=0*)

Score, args are arguments for prob_function. Thus: if probfunction = Rule.probability_labels, then args should be [labels], if it is Rule.probability_spanrels then args should be [spanrels, normalization_factor]

transform_to_Production(*rule*)

Transform rule to Production object (NLTK-style)

transform_to_WeightedProduction(*rule*)

Transform Rule object to WeightedProduction object (NLTK-style)

CHAPTER

EIGHT

TESTS MODULE

class tests.**Tests**

test_all()

class tests_alignments.**AlignmentsTests**
Tests Alignments Module

alignment_test_all()

consistency_test1()

consistency_test2()

consistency_test3()

dict_test()
Test the function Alignments.HAT_dict()

span_test1()
Test if correct spans are found for a monotone alignment with no unaligned words

span_test2()
Test if correct spans are found for a non monotone many-to-many alignment, with no unaligned words on source nor target side.

span_test3()
Test if correct spans are found for a one-to-one alignment with some unaligned words on source and target side

span_test4()
Test if correct spans are found for a non monotone many-to-many alignment with unaligned words on both and target side.

spans_test_all()
Return True if all span tests return True

class tests_dependencies.**DependencyTests**

allr_test1()
Test left branching relations for sentence 'I give the boy some flowers'

allr_test2()

dependencies_test_all()
Run all dependency tests.

labels_annotation_test()
Test annotated labels for a manually constructed sentence

labels_test1()
Test functioning plain labelling for sentence 'I give the boy some flowers'

lr_test()
Test left branching relations for sentence 'I give the boy some flowers'

rr_test()
Test right branching relations for sentence 'I give the boy some flowers'

spanrelations_test()
Test if spanrelations are extracted correctly from a list of dependencies

test_all_labels()
Test for label all function

test_samt_labels()
Test SAMT labels

class tests_node.**NodeTests**
Tests functionality of Node class.

path_test1()
Test if paths are computed as intended by manually constructing a graph with 5 nodes and a couple of edges. Output True if correct paths are found, False otherwise.

path_test2()
Test if shortest paths are computed as intended by manually constructing a graph with 5 nodes and a couple of edges. Output True if correct paths are found, False otherwise.

path_test3()
Test if shortest paths are computed as intended by manually constructing a fully connected graph with 5 nodes. Output True if correct paths are found, False otherwise.

path_test_all()

worst_case_test(*nr_of_nodes*)
Speed test for shortest_paths_to. Create a fully connected graph with

nr_of_nodes nodes and compute the shortest_paths between all nodes. Output running time.

class tests_rule.**RuleTests**

Testing class for the rule class.

rules_test_all()

Return True if all rule tests return True

test2()

test_hatrules()

Test if the correct HATgrammar is generated for the sentence 'My dog likes eating sausages', with alignment '0-0 1-1 2-2 2-3 3-5 4-4'.

test_hatrules2()

Test if the correct HATgrammar is generated for the sentence 'My dog likes eating sausages', with alignment '0-0 1-1 2-2 2-3 3-5 4-4'.

test_rules()

Test if the correct grammar is generated for the sentence 'My dog likes eating sausages', with alignment '0-0 1-1 2-2 2-3 3-5 4-4'.

class tests_scoring.**ScoreTests**

Test Scoring Class

score_test1()

Sentence: 'my dog likes eating sausage'
Alignment: '0-0 1-1 2-2 2-3 3-5 4-4'
Dependencies: 'nsubj(likes-3, dog-2)', 'root(ROOT-0, likes-3)', 'xcomp(likes-3, eating-4)' and 'dobj(eating-4, sausages-5)'.

Manual score normal rules spanrels: 1.0
Manual score hatrules spanrels: 0.75
Manual score hatrules spanrels deep: 1.0

score_test2()

Sentence: 'european growth is inconceivable without solidarity .'
Alignment: '0-0 1-1 2-2 3-3 4-4 5-5 6-6'
Dependencies: 'nn(growth-2, european-1)', 'nsubj(inconceivable-4, growth-2)', 'cop(inconceivable-4, is-3)', 'root(ROOT-0, inconceivable-4)', 'prep(inconceivable-4, without-5)' and 'pobj(without-5, solidarity-6)'

Manual score all rules spanrels: 1.0
Manual score hat rules spanrels: 0.6
Manual score hat rules spanrels deep: 1.0

score_test3()

Sentence: 'approval of the minutes of the previous sitting'

Bibliography

Alfred V. Aho and Jeffrey D. Ullman. Properties of syntax directed translations. *Journal of Computer and System Sciences*, 3(3):319–334, 1969.

Lars Ahrenberg, Magnus Merkel, Anna Sågvall Hein, and Jörg Tiedemann. Evaluation of word alignment systems. In *LREC*, 2000.

Hala Almaghout, Jie Jiang, and Andy Way. CCG augmented hierarchical phrase based machine-translation. 2010.

Hiyan Alshawi, Shona Douglas, and Srinivas Bangalore. Learning dependency translation models as collections of finite-state head transducers. *Computational Linguistics*, 26(1):45–60, 2000.

Alexandra Birch and Miles Osborne. LRscore for evaluating lexical and reordering quality in MT. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 327–332. Association for Computational Linguistics, 2010.

Phil Blunsom, Trevor Cohn, and Miles Osborne. Bayesian synchronous grammar induction. In *Advances in Neural Information Processing Systems*, pages 161–168, 2008.

Rens Bod. Unsupervised parsing with U-DOP. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 85–92. Association for Computational Linguistics, 2006.

Ch Boitet, Pierre Guillaume, and Maurice Quezel-Ambrunaz. Implementation and conversational environment of ARIANE 78.4, an integrated system for automated translation and human revision. In *Proceedings of the 9th conference on Computational linguistics-Volume 1*, pages 19–27. Academia Praha, 1982.

Peter Brown, John Cocke, S Della Pietra, V Della Pietra, Frederick Jelinek, R Mercer, and P Roossin. A statistical approach to french/english translation. In *Proceedings, RIA088 Conference*, 1988.

Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2): 79–85, 1990.

Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311, 1993.

Daniel M Cer, Marie-Catherine De Marneffe, Daniel Jurafsky, and Christopher D Manning. Parsing to stanford dependencies: Trade-offs between speed and accuracy. In *LREC*, 2010.

John Chandioux. METEO: an operational system, for the translation of public weather forecasts. In *FBIS Seminar on Machine Translation. American Journal of Computational Linguistics, microfiche*, volume 46, pages 27–36, 1976.

David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics, 2005.

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

Noam Chomsky. Three models for the description of language. *Information Theory, IRE Transactions on*, 2(3):113–124, 1956.

Michael A Covington. *A dependency parser for variable-word-order languages*. Citeseer, 1990.

Marie-Catherine De Marneffe and Christopher D Manning. Stanford typed dependencies manual. *URL http://nlp. stanford. edu/software/dependencies_ manual. pdf*, 2008.

Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454, 2006.

Ralph Debusmann. An introduction to dependency grammar. *Hausarbeit fur das Hauptseminar Dependenzgrammatik SoSe*, 99:1–16, 2000.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

Leon E Dostert. The Georgetown-IBM experiment. *1955). Machine translation of languages. John Wiley & Sons, New York*, pages 124–135, 1955.

Jason Eisner. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 2*, pages 205–208. Association for Computational Linguistics, 2003.

Heidi J Fox. Phrasal cohesion and statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 304–3111. Association for Computational Linguistics, 2002.

Stefan L Frank, Rens Bod, and Morten H Christiansen. How hierarchical is language use? *Proceedings of the Royal Society B: Biological Sciences*, 279 (1747):4522–4531, 2012.

Pascale Fung, Wu Zhaojun, Yang Yongsheng, and Dekai Wu. Automatic learning of chinese english semantic structure mapping. In *Spoken Language Technology Workshop, 2006. IEEE*, pages 230–233. IEEE, 2006.

Haim Gaifman. Dependency systems and phrase-structure systems. *Information and control*, 8(3):304–337, 1965.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What's in a translation rule? In *HLT-NAACL*, pages 273–280, 2004.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 961–968. Association for Computational Linguistics, 2006.

Daniel Gildea, Giorgio Satta, and Hao Zhang. Factoring synchronous grammars by sorting. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 279–286. Association for Computational Linguistics, 2006.

Joao Graca, Joana Paulo Pardal, Luísa Coheur, and Diamantino Caseiro. Building a golden collection of parallel multi-language word alignment. In *LREC*, 2008.

Hany Hassan, Khalil Sima'an, and Andy Way. Supertagged phrase-based statistical machine translation. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 288, 2007.

David G Hays. Dependency theory: A formalism and some observations. *Language*, 40(4):511–525, 1964.

Liang Huang, Hao Zhang, Daniel Gildea, and Kevin Knight. Binarization of synchronous context-free grammars. *Computational Linguistics*, 35(4):559–595, 2009.

William John Hutchins and Harold L Somers. An introduction to machine translation. 1992.

Rebecca Hwa, Philip Resnik, Amy Weinberg, and Okan Kolak. Evaluating translational correspondence using annotation projection. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 392–399. Association for Computational Linguistics, 2002.

Nitin Indurkhya and Fred J Damerau. *Handbook of natural language processing*, volume 2. CRC Press, 2010.

Theo Janssen. Algebraic translations, correctness and algebraic compiler construction. *Theoretical Computer Science*, 199(1):25–56, 1998.

Theo MV Janssen. Compositionality. 1996.

Maxim Khalilov and Khalil Sima'an. Statistical translation after source reordering: Oracles, context-aware models, and empirical analysis. *Natural Language Engineering*, 18(4):491, 2012.

Philip Koehn. *Statistical Machine Translation*. Cambridge University Press, 2008.

Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, 2005.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics, 2003.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics, 2007.

Patrik Lambert, Adrià De Gispert, Rafael Banchs, and José B Mariño. Guidelines for word alignment evaluation and manual alignment. *Language Resources and Evaluation*, 39(4):267–285, 2005.

Jan Landsbergen, Jan Odijk, and Andre Schenk. The power of compositional translation. *Literary and Linguistic Computing*, 4(3):191–199, 1989.

Dekang Lin. A path-based transfer model for machine translation. In *Proceedings of the 20th international conference on Computational Linguistics*, page 625. Association for Computational Linguistics, 2004.

Gideon Maillette de Buy Wenniger, Maxim Khalilov, and Khalil Sima'an. A toolkit for visualizing the coherence of tree-based reordering with word-alignments. *The Prague Bulletin*, pages 97–106, 2010.

Daniel Marcu and William Wong. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 133–139. Association for Computational Linguistics, 2002.

I Dan Melamed, Giorgio Satta, and Benjamin Wellington. Generalized multitext grammars. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 661. Association for Computational Linguistics, 2004.

Arul Menezes and Stephen D Richardson. A best-first alignment algorithm for automatic extraction of transfer mappings from bilingua corpora. In *Recent advances in example-based machine translation*, pages 421–442. Springer, 2003.

Rada Mihalcea and Ted Pedersen. An evaluation exercise for word alignment. In *Proceedings of the HLT-NAACL 2003 Workshop on Building and using parallel texts: data driven machine translation and beyond-Volume 3*, pages 1–10. Association for Computational Linguistics, 2003.

Markos Mylonakis and Khalil Sima'an. Phrase translation probabilities with ITG priors and smoothing as learning objective. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 630–639, Honolulu, USA, October 2008. Association for Computational Linguistics.

Markos Mylonakis and Khalil Sima'an. Learning probabilistic synchronous CFGs for phrase-based translation. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 117–125. Association for Computational Linguistics, 2010.

Markos Mylonakis and Khalil Sima'an. Learning hierarchical translation structure with linguistic annotations. In *ACL*, pages 642–652, 2011.

Ryan Nefdt. Constituentless compositionality: A compositional account of dependency grammar. Master's thesis, University of Amsterdam, 2013.

Joakim Nivre. Dependency grammar and dependency parsing. *MSI report*, 5133 (1959):1–32, 2005.

Franz Josef Och and Hermann Ney. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 440–447. Association for Computational Linguistics, 2000.

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Franz Josef Och and Hermann Ney. The alignment template approach to statistical machine translation. *Computational linguistics*, 30(4):417–449, 2004.

Franz Josef Och, Christoph Tillmann, Hermann Ney, et al. Improved alignment models for statistical machine translation. In *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, 1999.

Sebastian Padó and Mirella Lapata. Optimal constituent alignment with edge covers for semantic projection. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1161–1168. Association for Computational Linguistics, 2006.

Francis Jeffry Pelletier. The principle of semantic compositionality. *Topoi*, 13(1): 11–24, 1994.

Arjen Poutsma. Data-oriented translation. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*, pages 635–641. Association for Computational Linguistics, 2000.

Chris Quirk and Arul Menezes. Do we need phrases?: challenging the conventional wisdom in statistical machine translation. In *Proceedings of the main conference on human language technology conference of the north american chapter of the association of computational linguistics*, pages 9–16. Association for Computational Linguistics, 2006a.

Chris Quirk, Arul Menezes, and Colin Cherry. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 271–279. Association for Computational Linguistics, 2005.

Christopher Quirk and Arul Menezes. Dependency treelet translation: the convergence of statistical and example-based machine-translation? *Machine Translation*, 20(1):43–65, 2006b.

MT Rosetta. *Compositional translation*. Kluwer academic publishers Dordrecht, 1994.

Giorgio Satta and Enoch Peserico. Some computational complexity results for synchronous context-free grammars. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 803–810. Association for Computational Linguistics, 2005.

Remko Scha. Taaltheorie en taaltechnologie; competence en performance. *Computertoepassingen in de Neerlandistiek, Almere: Landelijke Vereniging van Neerlandici (LVVN-jaarboek)*, 11:7–22, 1990.

Khalil Sima'an and Gideon Maillette de Buy Wenniger. Hierarchical alignment trees: A recursive factorization of reordering in word alignments with empirical results. 2013.

Anders Søgaard. Can inversion transduction grammars generate hand alignments. In *Proccedings of the 14th Annual Conference of the European Association for Machine Translation (EAMT)*, 2010.

Anders Søgaard and Jonas Kuhn. Empirical lower bounds on alignment error rates in syntax-based machine translation. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation*, pages 19–27. Association for Computational Linguistics, 2009.

Anders Søgaard and Dekai Wu. Empirical lower bounds on translation unit error rate for the full class of inversion transduction grammars. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 33–36. Association for Computational Linguistics, 2009.

Harold Somers. Review article: Example-based machine translation. *Machine Translation*, 14(2):113–157, 1999.

Mark Steedman and Jason Baldridge. Combinatory categorial grammar. *Non-Transformational Syntax Oxford: Blackwell*, pages 181–224, 2011.

Lucien Tesnière and Jean Fourquet. *Eléments de syntaxe structurale*, volume 1965. Klincksieck Paris, 1959.

Dániel Varga, Péter Halácsy, András Kornai, Viktor Nagy, László Németh, and Viktor Trón. Parallel corpora for medium density languages. *AMSTERDAM STUDIES IN THE THEORY AND HISTORY OF LINGUISTIC SCIENCE SERIES 4*, 292:247, 2007.

Ye-Yi Wang. *Grammar inference and statistical machine translation.* PhD thesis, Citeseer, 1998.

Warren Weaver. Translation. *Machine translation of languages*, 14:15–23, 1955.

Benjamin Wellington, Sonjia Waxmonsky, and I Dan Melamed. Empirical lower bounds on the complexity of translational equivalence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 977–984. Association for Computational Linguistics, 2006.

Dekai Wu. An algorithm for simultaneously bracketing parallel texts by aligning words. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 244–251. Association for Computational Linguistics, 1995.

Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational linguistics*, 23(3):377–403, 1997.

Dekai Wu. MT model space: statistical versus compositional versus example-based machine translation. *Machine Translation*, 19(3-4):213–227, 2005.

Hao Zhang, Liang Huang, Daniel Gildea, and Kevin Knight. Synchronous binarization for machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 256–263. Association for Computational Linguistics, 2006.

Hao Zhang, Daniel Gildea, and David Chiang. Extracting synchronous grammar rules from word-level alignments in linear time. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 1081–1088. Association for Computational Linguistics, 2008.

Andreas Zollmann and Ashish Venugopal. Syntax augmented machine translation via chart parsing. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 138–141. Association for Computational Linguistics, 2006.