Automated Translation with Interlingual Word Representations

Oele, Dieke Merel

Automated Translation with Interlingual Word Representations

Dieke Oele

The work in this thesis has been carried out under the auspices of the Center for Language and Cognition Groningen (CLCG), affiliated with the University of Groningen.

Groningen Dissertations in Linguistics 123
ISBN: 978-94-034-0481-3 (printed version)
ISBN: 978-94-034-0480-6 (electronic version)

Document prepared with LaTeX 2$_\varepsilon$ and typeset by pdfTeX
Cover: *L'Apparition*, René Magritte. Paris, 1928

rijksuniversiteit
groningen

Automated Translation with Interlingual Word Representations

Proefschrift

ter verkrijging van de graad van doctor aan de
Rijksuniversiteit Groningen
op gezag van de
rector magnificus prof. dr. E. Sterken
en volgens besluit van het College voor Promoties.

De openbare verdediging zal plaatsvinden op

vrijdag 15 maart 2018 om 12.45 uur

door

Dieke Merel Oele

geboren op 20 december 1989
te Goes

Acknowledgements

I am grateful to many people for having been able to write this thesis, and I would like to try to express my gratitude to all of them.

In particular I wish to thank my supervisor. Gertjan, it has been an honor working with you. Thanks for giving me the chance to start this endeavor, but mostly, for not giving up on me in the process. You were there every week for discussions and suggestions that enabled me to my finish my project. Next to work, I will remember our frequent talks about suitable hiking locations and its flora and fauna. More interesting of course, was actually going on hikes, spotting dolphins and watching a big flight of gannets along the Portuguese coasts.

Special thanks goes to the other members of the QTLeap project. You have provided me with an interesting framework to write this thesis. I have greatly enjoyed the regular meetings which have been a very good learning environment for me.

I would also like to thank the reading committee of my thesis: Prof. Jack Hoeksema, Prof. Piek Vossen and Prof. Andy Way. Thanks for your critical and helpful comments. Although I did not have time to address them all, they attributed greatly to the final version of this thesis.

My time at the University of Groningen was made an unforgettable one by my colleagues at the Alfa-Informatica group, Antonio, Barbara, George, Gertjan, Gosse, Gregory, Hessel, Johan, John, Lasha, Leonie, Malvina, Martijn, Masha, Rik, Rob and Tommaso.

I would further like to thank my fellow PhD students and office mates. Anna, Anna, Jakolien, Johannes, Hessel, Fabrizio, Harm, Hessel, Kilian, Noortje, Rik, Rob, Steven, Masha, Pauline, Simon, Valerio, Yinxing and Yiping, thanks for the great time in the office and for the occasional Friday afternoon drinks.

Team Zardoz and team 1311.*, it has been great fun (trying) to win pub quizzes with you guys. I hope you will be able continue this winning streak (remember: when in doubt, it is most probably Michael Bublé).

Thanks to Lotte and Anna, who agreed to be my paranymphs and for having tea and prosecco with me in the process.

In the last couple of years I have had the pleasure to reunite with some old friends here in Groningen and to make new ones at the same time. Thanks *Koala Koffie Club*, for welcoming me into your group of friends. Leander and Sietske, thanks for teaching me how to row (well), and providing me with a daily escape route on the canals.

To my best friends in Groningen, Lotte and Laurens, thanks for always being there for me when I needed to have a chat and for all hours on the water and glasses of red wine on the main land when I didn't. I could not have imagined to find better company for my time here.

Even though Groningen is situated at the end of the world, I am very pleased that some of the friendships on the other side remained intact. Annabelle and Mattias, thanks for always having a place for me in Antwerp and for visiting me now and then. Rainer, Marleen, Daan, Lavina, Cora, Sander, I hope we can keep meeting each other at least once a year in the future.

Last but not the least, I would like to thank my family. Everyone in Groningen, Lientje, Frans, Bien, Hans, Josje, Rik, Pim, Jelmer, Frederike, Marijn, Veerle and Teun thanks for your support, the nice dinners and spontaneous trips. It has been a pleasure to see you guys more often. Betteke, Cees, Jesse, Esther, Jaap, Hans, Tanja, Anne, Peter Jan, Jorick, Zita, Ieneke, Mansje and Ken thanks for your equally important support from the other side of the Netherlands, the US and Canada.
Groningen,

February 13, 2018

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

One of the main challenges in machine translation (MT) is that words can have several meanings depending on their context, a problem known as lexical ambiguity. Example 1 shows such ambiguity for the words “drive” and “port”.

(1) a. British tourists could soon be allowed to **drive** on the left in the **port** of Calais.

b. An external **drive** typically includes a bridging device between its interface to a USB interface **port**.

Here, (1a) depicts the meaning of the respective words in the sense of transportation: a journey in a vehicle in a place where people and merchandise can enter or leave a country. The same words in sentence 1b, on the other hand, refer to entities from computer science: a device that writes data onto or reads data from a storage medium and a computer circuit.

The most straightforward approach to MT is to directly map each word in a sentence to a corresponding word in the target language. It is clear that this is not a very feasible approach, especially if one would want to use the MT system in multiple domains, as demonstrated in example 1. When translating sentence 1a and 1b, to, for example, Dutch, the English words "drive" and "port" should be translated to "rijden" (drive in a vehicle) and "haven" (harbour) and, "schijf" (disk) and "poort" (gateway) respectively. Thus, understanding of the meaning of the words and the meaning of the overall sentence appears to be crucial for MT performance.

For a successful translation, the meaning of a text in a source language sentence should be transferred to an equivalent meaning in the target language sentence. For this, knowledge is required on the meaning of such words, that can have different translations for different senses. As an alternative to MT systems that directly map each word in a source sentence to a target word, interlingual and transfer-based systems perform further analysis on the input sentences.

Interlingual systems first identify an interlingual, or meaning, representation for a concept expressed in the source language (Resnik, 2006). Analysis is performed on sentences in the source language resulting in interlingual representations that, in turn, can be mapped to a target language sentences. Although the idea of using an interlingual representation sounds promising, many issues remain. For example, when no transfer phase is used, the effort that goes into creating the analysis and generation modules increases. Also, many stylistic elements are usually lost in this process. As the representation is independent of syntax, the generated target text tends to read more like a paraphrase. Transfer-based systems, on the other hand, rely on a language-specific transfer of the source language structure to a target language structure.

To combine both the strengths of both transfer based and interlingual systems, we propose a transfer-based MT system with interlingual representations of words. In the analysis phase of such an MT system, for each word it is determined to which sense it belongs. For example, given an input sentence such as sentence 1b, an ideal system first determines that the word "drive" depicts a computer device and "port" a computer circuit. This information is stored as an interlingual representation of this sense in order to preserve the meaning of a word during transfer.

On the other side of the pipeline, in the generation phase, lemmas need to be selected from the meaning representations that were the result of analysis. As these representations are rather abstract, several words can be used to express its meaning. They, however, do not fit equally well in the target context. For instance, in example 1, a Dutch translation of sentence 1b, for the word "drive" in the computer device sense, the optional target lemmas are {'disk', 'magneetschijf', 'schijf'} while, for "port" it can choose from {"interface", "poort"}. As opposed to the lemmas "schijf" and "poort" that would fit this context well, the other ones, although they have a similar meaning, would result in a less fluent sentence in this context. Therefore, in the generation phase, a module is used that selects that lemma that best fits the target context.

(2) Een externe {"disk", "magneetschijf", "schijf"} bevat meestal een overbruggingsapparaat tussen de interface naar een USB {"interface", "poort"}.
(An external {disk, magnetic disc, drive} typically includes a bridging device between its interface to a USB {"interface", "port"}.)

About this Thesis

In this thesis, we investigate the use of transfer-based MT systems with interlingual representations of words. This way, we approach the problem of lexical ambiguity in machine translation as two separate tasks: word sense disambiguation (WSD) and lexical choice. First, the words in the source language are disambiguated according to their sense, resulting in interlingual representations of words. Then, for the selection of target words from the interlingual word representations, a lexical choice module is used that selects the most appropriate word in the target language. The current framework can be applied to any language pair, for which the required datasets are available, but we focus on Dutch and English throughout this thesis.

The research presented in this thesis was undertaken in the context of the European Project "QTLeap": Quality Translation by Deep Language Engineering Approaches. [1] Nowadays, neural machine translation (NMT) is the state of the art in MT. However, at the time of the start of the project, statistical machine translation (SMT) was the predominant approach to MT. Although SMT was yielding good results, it was felt that further improvements were hard to obtain. Therefore, an alternative approach was advocated on the assumption that further improvements are possible by using linguistic and semantic analysis of the utterances to be translated. The goal of the project was, therefore, to improve MT by examining the merits of the exploitation of linguistic and semantic analysis, in combination with statistical methods. Furthermore, the use of linguistic features was explored by experimenting with methods that are based on syntactic and semantic representations of natural language utterances.

[1] `http://qtleap.eu/`

As the majority of the work in this thesis was done as a part of the QTLeap project, it does not contain any further research into SMT nor NMT. The breakthrough in neural methods, however, did not go unnoticed of course. In Part 2 of this thesis, on novel methods in word sense disambiguation, we exploit such neural methods in a novel extension of the Lesk algorithm (Lesk, 1986) which crucially depends on embeddings constructed by these neural methods.

This thesis is organized in three parts. The first part focuses on machine translation and describes work that was undertaken in the QTLeap project for the Dutch–English language pair. In Chapter 2, we first describe machine translation, and give an overview of previous work. Chapters 3 and 4 are devoted to the development and evaluation of the MT systems for the Dutch–English language pairs.

In Chapter 3 we describe the tools that are used to create a transfer-based MT system for NL→EN and EN→NL. The goal of this chapter is to give an overview of the the development of the Dutch–English MT systems within the QTLeap project. It provides the background for further work in parts 2 and 3 of the thesis. Most of the work in Chapter 3 was carried out by project partners and is described here in detail to ensure that the later parts of the thesis can be properly understood in the context. In particular, several modules for translating between English and Dutch where developed by project colleagues at the Charles University in Prague.

Chapter 4, then, describes the evaluation of the MT systems developed in QTLeap. Both automatic evaluation measures were applied and a task-based evaluation with human subjects was performed. In addition, a qualitative evaluation using linguistic categories was performed.

Firstly, the SMT baselines for NL -> EN and EN -> NL were developed by the author of this thesis using existing tools. Secondly, together with project partners from Higher Functions, a real usage scenario was developed and several experiments were undertaken with human subjects in order to estimate the quality of the various translation systems (Gaudio et al., 2016). For this evaluation, the writer of this thesis was responsible for the Dutch components of the evaluation. This included the localization into Dutch of the experimental framework, and the recruitment and instruction of the volunteers. Thirdly, a detailed linguistic analysis of the translation errors by the various translation systems was performed by the author of this thesis, using the framework developed by project partners at the DFKI.

In the second part of this thesis, we consider the analysis component of our interlingual setup and present work on word sense disambiguation. We propose a WSD method that is based on a combination of WordNet and sense extended word embeddings. In Chapter 5, we describe previous work and present our method, as well as experimental results. We show that our method performs better than state-of-the-art WSD methods that only make use of a knowledge base. Additional experiments demonstrate the added value of the use of sense embeddings and confirm that our system works well consistently in different domains without using any domain specific data. We, furthermore, show that our method can be improved with other known extensions.

In Chapter 6, we further test our WSD system by participating in the SemEval-2017 Task 7 for the subtasks of homographic pun location and homographic pun interpretation. We describe experiments on different methods of splitting the context of the target pun and compare our method to several baselines. We show that our WSD method can be used successfully for pun identification and that performance improves when the context is split before disambiguating the target word. The WSD system used in this chapter is the one that was created by the author of this thesis, as described in the previous chapter. Adjustments to the system to make it fit for the pun location and interpretation tasks, the development of the development data and the evaluation of the results was shared work with Kilian Evang.

The third, and final, part of this thesis addresses the task of selecting lexical items from interlingual representations of words in the framework of MT. We propose a new lexical choice module that is part of a generation pipeline where the input representations are mapped to target sentences. In Chapter 7, we first describe the task of lexical choice from interlingual representations of words. It furthermore contains some preliminary experiments that explore what is necessary for good performance.

The outcomes of Chapter 7 are used in Chapter 8, where we describe a more sophisticated model for lexical choice. Our model considers dependency trees with word senses as hidden Markov tree models (HMTMs) Crouse et al. (1996); Durand et al. (2004); Žabokrtský and Popel (2009). We evaluate our model for lexical choice by comparing it to a statistical transfer component in an MT system for English to Dutch. In this set-up, the senses of the words are determined in English analysis, and then our model is used to select the best Dutch lemma for the given word sense. We show that our model works well as it outperforms the most frequent baseline. Also, the manual evaluations confirm that our model chooses better lemmas. Furthermore, when using the algorithm only on out of vocabulary (OOV) items it slightly improves a system that does not use lexical choice in generation.

In Chapter 9, we conclude the thesis and discuss its main contributions.

Contributions

In this thesis, we make the following scientific contributions:

A detailed description of the development and evaluation of the MT systems for Dutch–English in the QTLeap project
We present the Dutch–English MT systems that are a result of the combination of the TectoMT system and the Alpino parser (van Noord, 2006) and generator (De Kok, 2010; De Kok et al., 2011; De Kok, 2013). Also, we provide a description of the evaluation of these MT systems where several approaches were used ranging from automatic measures, human error annotation and task-based evaluation. The created MT framework provides the background for the experiments in parts II and III.

A knowledge-based word sense disambiguation method
We propose a WSD method that is similar to the classic Lesk algorithm (Lesk, 1986) as it exploits the idea that shared words between the context of a word and each definition of its senses provides information on its meaning. However, instead of counting the number of words that overlap, we use sense-extended word embeddings to compute the cosine similarity between the gloss of a sense and the context. Our method performs well compared to state-of-the-art knowledge based WSD methods. It also only requires a knowledge base and large amounts of text.

WSD for homographic pun interpretation
We present a sense and word embedding-based approach to pun interpretation. As our WSD method outputs all potential senses and a score, it can be used for pun interpretation, a task that requires two output senses. Furthermore, we propose to split the

context that surrounds the target pun as we expect some words to be more informative for either one of the respective senses.

WSD for homographic pun location
We present a sense and word embedding-based approach to pun location. For this, we use the output of our pun interpretation system for pun location. As we expect the two meanings of a pun to be very dissimilar, we locate puns by selecting the polysemous word with the most dissimilar two senses. We compute sense embedding cosine distances for each sense-pair and select the word that has the highest distance.

A lexical choice module
We propose a new model for lexical choice that selects lemmas given WordNet synsets in the abstract representations that are the input for generation. In order to determine the most appropriate lemma in its context, we map underspecified dependency trees to Hidden Markov Trees that take into account the probability of a lemma given its governing lemma, as well as the probability of a word sense given a lemma. A tree-modified Viterbi algorithm is then utilized to find the most probable hidden tree containing the most appropriate lemmas in the given context. Similar to our WSD system, our model does not require any domain specific parallel data.

Publications

Several chapters in this thesis are adapted versions of peer-reviewed publications:

Chapter 5:

- Oele, D. and van Noord, G. (2017). Distributional Lesk: Effective Knowledge-Based Word Sense Disambiguation. In *International Conference on Computational Semantics (IWCS),* Montpellier, France

- Oele, D. and van Noord, G. (2018). Simple Embedding-Based Word Sense Disambiguation. In *Global Wordnet Conference 2018,* Singapore, Singapore

Chapter 6:

- Oele, D. and Evang, K. (2017). BuzzSaw at SemEval-2017 Task 7: Global vs. Local Context for Interpreting and Locating Homographic English Puns with Sense Embeddings. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017),* pages 444–448, Vancouver, Canada. Association for Computational Linguistics

Chapter 8:

- Oele, D. and van Noord, G. (2016). Choosing lemmas from Wordnet synsets in Abstract Dependency Trees. In *Workshop on Deep Language Processing for Quality Machine Translation (DeepLP4QMT),* Varna, Bulgaria

- Oele, D. and van Noord, G. (2015). Lexical choice in Abstract Dependency Trees. In *First Deep Machine Translation Workshop,* page 73, Prague, Czech Republic

PART I

Machine Translation

CHAPTER 2

Background: Machine Translation

2.1 Introduction

In machine translation (MT), a text is automatically translated from a source language to a target language. The source of information for an MT system can be either rules and dictionaries, data or both. While the first approach is linguistically motivated, the second one only relies on information from large amounts of text.

Since the 1960s, countless efforts have been made to create MT systems based on syntactic and semantic analysis. These first MT systems were primarily based on linguistic rules. The advantage of the use of such systems is that no parallel data is required and they tend to be domain independent. However, translating automatically, solely on the basis of rules can be very difficult and time consuming.

This is due to the fact that, if one wants to reach a broader coverage of a language or create a system for a new language pair, a lot of manual effort and language expertise is required.

The main disadvantage of rule-based systems is that they do not reach the same quality of output compared to data driven approaches that do not require manually annotated data. Such methods make use of large amounts of parallel data and therefore do not require linguistically informed resources. A popular data driven approach is statistical machine translation (SMT). Although early SMT systems were rather simplistic and assumed a word-for-word correspondence between the source and target language, phrase-based SMT (PB-SMT) can produce higher quality translations. A third paradigm, hybrid MT, combines both rule-based and corpus-based methods.

In this chapter, we first describe the characteristics of rule-based systems in Section 2.2. We then give a brief overview of data driven approaches in Section 2.3. In Section 2.4, we describe possible approaches to creating such hybrid MT systems.

2.2 Rule-based Machine Translation

Rule-based machine translation (RBMT) systems typically consist of rules on the source and the target language that are based on linguistic knowledge. This knowledge can either be derived from dictionaries and handcrafted grammars that cover semantic, morphological, and syntactic regularities in both languages or by annotation by trained linguists. Examples of rule based MT systems are: Systran (Toma, 1977), Apertium (Forcada et al., 2011) and Grammatical Framework (GF); (Ranta, 2011)).

Currently, the development of purely rule based MT systems is quite rare. This is not surprising as the addition of new language pairs requires a large amount of work by trained linguists which makes it very expensive, especially in comparison with data driven methods that only require parallel data. To the best of our knowledge, Apertium (Forcada et al., 2011) and GF (Ranta, 2011) are the only purely RBMT systems in active development. Apertium is a free, open source project, providing tools for potential developers to help create new resources and build systems for new language pairs. It currently supports translation between 43 language pairs, with a focus on small and under resourced languages. On the other hand, GF, is a type theoretical grammar formalism (Martin-Löf, 1982), that can be used for MT applications, more specifically for interlingua based translation systems (see Section 2.2.3).

In RBMT, the words in a source sentence can be translated directly to target language words or the sentence can be mapped to an intermediate representation. An intermediate representation is a linguistic account of the input sentence that is more abstract, on the basis of which a target language sentence can be created. Generally, in an RBMT system, an input sentence is first analyzed morphologically, syntactically and semantically. This information is then encoded into the intermediate representation, after which a sentence is generated in the target language on the basis of this analysis.

RBMT approaches can be categorized on the basis of the nature of the linguistic knowledge used, and by the level of abstraction of the intermediate representation between the source and target sentence. This categorization can be illustrated with the Vauquois triangle (Vauquois, 1968), which can be found in Figure 2.1. In this triangle, the arrows that point from left to right, represent the translation pro-

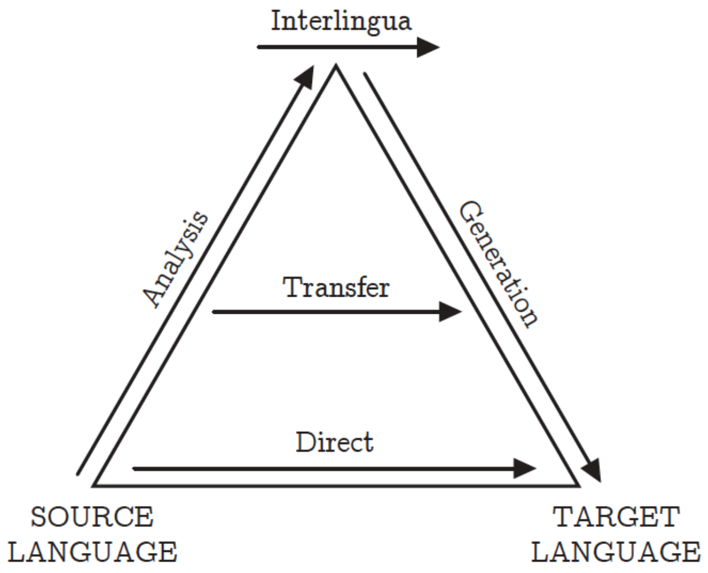

Figure 2.1 | *The Vauquois triangle (Vauquois, 1968)*

cess. On the left side, a source language sentence is received as input and, on the right side, a target sentence is produced as output.

The closer one moves to the top of the triangle, the more abstract the intermediate representation becomes, and the more language processing is required. The idea is that, at a higher level of abstraction, or higher in the triangle, more linguistic analysis is applied and the intermediate representations become more abstract. In this way, less language specific characteristics remain which could facilitate translation

On the basis of the nature of their intermediate representation, RBMT systems can be further categorized into three different categories that proceed from the bottom to the top of the triangle, namely:

direct systems, transfer-based systems, and interlingual systems. A description of each of these systems is provided in the following sections.

2.2.1 Direct Systems

Direct systems operate on the bottom of the Vauquois triangle and are often considered to be the simplest MT systems. They require only one single transformation without analysis of the source language sentence and without generation of the target language sentence. Instead, as direct systems are primarily based on dictionary entries, they translate a sentence word-by-word, without the use of an intermediate representation. These systems are usually designed for a specific source and target language pair. In addition to the use of bilingual dictionaries, some basic prepossessing of the sentence can be applied such as morphological analysis, lemmatization or both.

2.2.2 Transfer-based Systems

When moving upwards in the triangle, more linguistic processing is used. This is the case for transfer-based systems, systems that make use of morphological and syntactic analysis. They can be further divided into surface-transfer systems, where only shallow parsing, chunking or both is performed, or systems where a full representation is built for each sentence. Both types of systems proceed along the following three phases: analysis, transfer, and generation.

In the analysis phase, the source language sentence is analyzed into a representation that includes information on the grammar and semantics of a sentence. This usually involves morphological analysis and disambiguation, part-of-speech tagging, and parsing. Then, in the transfer phase, the representation of the source sentence, created in the analysis phase, is transformed into a corresponding target language representation. In the final step, the generation phase, a translation of the input sentence is created on the basis of the target language representation.

2.2.3 Interlingual Systems

As opposed to transfer-based systems, that rely on a language specific transfer phase, interlingual systems typically make use of an intermediate representation that is independent of any specific language. The intermediate representation, or interlingua, consists of a single underlying representation of a text in both the source and the target language. The motivation behind these systems is that, when more linguistic analysis is applied and the intermediate representations become more abstract, less language-specific characteristics remain which could facilitate translation.

In interlingual systems, the translation process does not include a transfer phase. Instead, during analysis, the meaning of the input sentence is encoded into an interlingua from which, in the generation phase, an output sentence can be derived directly. They could, therefore, be more easily used to create between multiple languages at once as it suffices to build analysis and generation modules for each of them independently.

Although the idea of using an interlingual representation sounds promising, many issues remain. For example, when no transfer phase is used, the effort that goes into creating the analysis and generation modules increases. Also, many stylistic elements are usually lost in this process. As the representation is independent of syntax, the generated target text tends to read more like a paraphrase. Due to its many complexities, only one interlingual MT system has ever been made operational in a commercial setting, namely the KANT system (H. Nyberg III and Mitamura, 1992; Mitamura and Nyberg, 1995). Another interlingual system that is still in active development is Grammatical Framework (Ranta, 2011).

2.2.4 Interlingual Representations of Words

In this thesis, we combine a transfer-based MT systems that includes interlingual representations of words An advantage of this approach is that one can translate from word meaning to word meaning. In this way, an incorrect translation of a word can be avoided if it is written in a similar way to other words. For this purpose, several multilingual ontologies are available, such as EuroWordNet (Vossen, 1998).

If one wants to translate words on the basis of meaning, first the correct sense of the word in its context needs to be found. The process of finding the correct sense is called word sense disambiguation (WSD). Experiments on this process are described in Part II. Then, in the target language, one needs to generate words from these representations, a subtask that is known as lexical choice. We describe experiments on this in Part III.

2.3 Data Driven Machine Translation

As an alternative to the effort of creating handmade rules for the translation of one language into another, data driven MT methods make use of parallel corpora. Those corpora consist of translations, that have previously been made by humans, of which the translated sentences are aligned to each other.

There are three main paradigms that make use of this approach. The first is example-based machine translation (EBMT), which we describe in (Section 2.3.1). Then, we describe SMT in Section 2.3.2, which can again be subdivided on the basis of translation units: word-based systems, phrase-based systems and syntax-based systems. A final, relatively new approach is neural machine translation (NMT), which we describe in (Section 2.3.3).

2.3.1 Example-based Machine Translation

EBMT systems use parallel corpora to automatically extract translation examples and reuse them as the basis for new translations (Somers, 1999). The idea of translating on the basis of examples was first suggested by Nagao (1984) for the English-Japanese language pair. The process of an EBMT system can be broken down into three stages:

1. **Match** fragments of the input text with segments in a database of real examples,
2. **Identify** corresponding translation fragments and align them,
3. **Recombine** the aligned texts to produce a target text.

When an exact match of the input sentence is found in the data, the full translation in the parallel corpus would be directly used with no further processing. Although EBMT can perform well on a small scale in some specific domain, it becomes less reliable when it is used in more general domains. As it is based on examples of larger sequences, it is impossible to predict every possible variation. Methods that are based on statistics, which we describe in the following section, are therefore used more often.

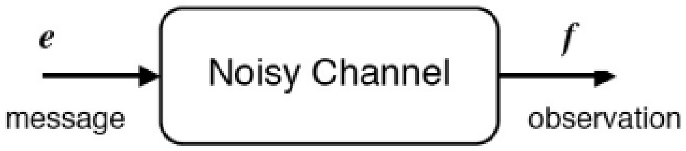

Figure 2.2 | *The noisy channel model for machine translation*

2.3.2 Statistical Machine Translation

As opposed to EBMT systems that use parallel corpora for direct translation analogy, SMT systems use parallel corpora to learn statistical models. Instead of focusing on the translation process itself, this approach starts with modeling the desired output. In SMT, given a source language text, the system models the probability of any target language text being its translation.

SMT systems model the problem of finding the best target translation for a source sentence as a noisy channel, as shown in Figure 2.3.2. Here, it is unknown which sentence is at the source of a target sentence. Suppose we are translating from French to English. When using the noisy channel model for MT, a document is translated according to the probability distribution $p(f|e)$ that an English target language string e is the translation of an observed French source language string f.

The source sentence f is considered to be a "corruption" of the target sentence e. However, we can look for sentences that maximize

$P(f|e)$ by applying Bayes′ theorem:

$$\begin{aligned}\hat{e} &= \arg\max_{e} P(e \mid f) \\ &= \arg\max_{e} \frac{P(f \mid e)P(e)}{P(f)} \\ &= \arg\max_{e} P(f \mid e)P(e)\end{aligned} \tag{2.1}$$

An advantage of applying the noisy channel approach to MT is that the problem is broken down into two smaller problems. This way, simpler problems can be solved separately because the estimations and model definitions are independent of each other. The two components of the model are:

1. **Translation Model**
   In SMT, the conditional probability of a source text given a target text $P(f|e)$, is modeled by the translation model (TM). The TM uses factorization into probabilities of each target word or phrase being the translation of the corresponding source word or phrase. Such probabilities can be estimated from word-aligned parallel training data.

2. **Language Model**
   The language model (LM) is used to model the probability of the target sentence $P(e)$ and to score the sentences generated by the TM. An LM is usually based on word n-grams containing estimations of the probabilities of their occurrences in monolingual training data.

Given these two components of the model, the output of the translation model on a new target sentence f is:

$$\hat{e} = \arg\max_{e} P(f \mid e) \times P(e) \tag{2.2}$$

Thus, the score for a potential translation e is the product of two scores:

1. The translation model score $p(f|e)$, which indicates how likely we are to see the source sentence f as a translation of e.

2. The language model score $p(e)$, which gives a prior distribution over which sentences are likely to be the target language.

While the first SMT systems were word-based (Section 2.3.2), phrase-based systems (Section 2.3.2) are more widely used nowadays.

Word-based Systems

Word-based SMT started with the IBM Models 1-5 and the Hidden Markov Model (HMM) for word alignment (Vogel et al., 1996). While the SMT approach was originally presented in Brown et al. (1988a,b) and in Brown et al. (1990), the five models are described in detail in Brown et al. (1993).

In the IBM models, each word in a source sentence is aligned to one or more target words (in IBM Model 1 and 3 respectively). Reordering of the words is then handled either on the basis of the absolute position of a word in a sentence, as in the IBM Model 2, or on the basis of the previous word translations in the source sentence, as in the IBM Models 4 and 5. In the HMM word-based alignment model, a translation is based only on the previous word's translation (Vogel et al., 1996).

Although the focus of SMT moved to phrase-based models (Section 2.3.2), the original IBM models and the HMM alignment model are still used as the base approach to word-alignment that precedes phrase-based SMT. We describe such systems in the following section.

Phrase-Based Systems

In phrase-based SMT systems (PB-SMT) (Koehn et al., 2003), parallel sentences are first segmented into pairs of parallel phrases. These phrases, or word n-grams, are then stored in a phrase table with corresponding frequencies of their occurrence in the training data. The resulting translation model can then be used to predict the probability of translating the phrases, rather than predicting translations of individual words.

Most current PB-SMT approaches are based on the method of Koehn et al. (2003), that led to the creation of Moses (Koehn et al., 2007), an SMT toolkit that allows for automatically training translation models for any language pair. Moses is currently one of the most widely used MT systems. Instead of using the basic noisy channel model for PB-SMT, Moses uses a discriminative log-linear model (Och and Ney, 2002), which allows the scores from the translation model and the language model to be combined with various features, including information, for example, in the context of a word or the grammatical structure of the sentence.

Several extensions that allow the inclusion of linguistic features to the phrase-based approach have been proposed, for example:

- **Factored models** (Koehn and Hoang, 2007) can be used to incorporate linguistic features, called factors, into a PB-SMT system. They are equivalent to the models used in phrase-based systems, except that they are not limited to the use of surface word forms but can also employ other word-level factors such as lemmas or part-of-speech tags. In addition, generation steps can be added that model the probability of obtaining one factor given another. For this purpose, language models can be built for specific factors.

The translation and generation steps are incorporated into the log-linear model for different layers of linguistic information. Each translation step learns how to map some factors from one language to the other. Similar to phrase-based models, factored models are typically learned from word aligned data.

- **Hierarchical models** can be used in PBSMT with syntax-based translation (Section 2.3.2). The hierarchical approach was pioneered by Chiang (2005) in their Hiero system. This process is inspired by phrase-structure syntax as the descriptions are formalized as synchronous context-free grammars that are learned from parallel text without syntactic annotations. The structures used in hierarchical translation systems, however, are not based on explicit linguistic annotations.

Syntax-based Systems

In syntax-based SMT, syntactic units are translated instead of single words or phrases. It is based on the recursive structure of language, often with the use of synchronous context free grammars (Hajič et al., 2004). These models may or may not make use of explicit linguistic annotations such as phrase structure constituency labels or labeled syntactic dependencies. Syntax-based SMT approaches make use of linguistically motivated structures on the source side (tree-to-string), on the target side (string-to-tree), or both the source side and the target side (tree-to-tree).

In tree-to-string translation, the syntactic structure of the source sentence is used to guide the SMT decoder in generating the target sentence. Such systems use rich source language representations to translate into unannotated word sequences in the target language.

A prominent example of such a tree-to-string translation approach is syntax-directed translation, introduced by Huang et al. (2006).

The inverse approach, string-to-tree translation is less frequently used. Here, a source sentence is used to generate a syntactic tree or several tree fragments of the target sentence to support grammatical coherent output and ground restructuring in syntactic properties.

Finally, in tree-to-tree translation, the maximum of available syntactic annotation is exploited. One such approach is synchronous tree substitution grammar (STSG), introduced into MT by Hajič et al. (2004), and formalized by Eisner (2003). It is based on the assumption that a valid translation of an input sentence can be obtained by local structural changes of the input syntactic tree and translation of node labels while there exists a derivation process common to both languages.

2.3.3 Neural Machine Translation

Although in recent years, SMT had been the dominant approach, a relatively new approach to based on deep neural networks has emerged (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014). This approach is inspired by the recent trend of deep representational learning. As opposed to PB-SMT, NMT does not consist of many subcomponents that need to be tuned separately. Instead, it simultaneously builds and trains a single, large neural network that reads a sentence and outputs its translation.

Most of the proposed NMT models employ encoders and decoders (Sutskever et al., 2014; Cho et al., 2014b), or use a language-specific encoder applied to each sentence whose outputs are then compared. An encoder neural network reads and encodes a source sentence into a vector used by the decoder to output a target language sentence.

The encoder and decoder are trained simultaneously in order to maximize the probability of a correct target sentence given a source sentence.

Neural machine translation models require only a fraction of the memory needed by traditional SMT models which makes it appealing in practice (Cho et al., 2014a). Also, it has already shown promising results as it achieves state-of-the-art performances for various language pairs (Hill et al., 2014; Jean et al., 2015; Luong et al., 2015a,b; Sennrich et al., 2016). In 2016, the popular MT platform Google Translate, implemented their first NMT systems for Chinese-English (Wu et al., 2016).

Although we are aware of the recent advances in MT guided by NMT, they were not very successful at the beginning of the project described in this work. Such systems, are therefore not explored in this thesis. However, in Part II, we make use of word embeddings that are created using neural networks. Distributed representations for words were proposed by Rumelhart (Rumelhart et al., 1986) and have been successfully used in language models (Bengio et al., 2006) and many natural language processing tasks, such as word representation learning (Mikolov et al., 2013b), named entity recognition (Turian et al., 2010), parsing and tagging (Socher et al., 2011).

Word embeddings represent the meaning of a word as contextual feature vectors in a high-dimensional space or some embedding, learned from unannotated corpora. This way, word vectors are built by mapping words to points in space that encode semantic and syntactic meaning despite ignoring word order information. A great advantage of word vectors is that they exhibited certain algebraic relations and can, therefore, be used for meaningful semantic operations such as computing word similarity (Turney, 2006), and exposing lexical relationships (Mikolov et al., 2013c).

We make use of word embeddings that are created with Word2Vec (Mikolov et al., 2013a). Its models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. It takes a large corpus as input and produces a vector space with the words in the corpus each corresponding to its own vector.

2.4 Hybrid Machine Translation

The main aim of the creation of hybrid machine translation systems is to take advantage of the strengths of different MT paradigms. Typically, hybrid approaches integrate both rule-based and data driven techniques in the development of MT systems. For instance, rules can be learned automatically from corpora, whereas corpus-based approaches are increasingly incorporating linguistic information. Following Thurmair (2009), we categorize hybrid systems into two main groups: multi-engine hybridization (Section 2.4.1) and single-engine hybridization (Section 2.4.2).

2.4.1 Multi-engine Hybridization

In multi-engine MT (MEMT), two or more existing systems are combined in order to get the best of both. This can be either done by selecting the best output from the MT systems used as they are, or by using parts of output hypotheses and recombining them into sentences. An example of the first method is Hildebrand and Vogel (2008). They search for the best n-grams in all output hypotheses available and then select the best hypothesis from the candidate list. An example of the second approach is proposed by Rosti et al. (2007) and extracts sentence-specific phrase translation tables from system outputs with alignments to the source sentence and runs a phrasal decoder with the newly created translation tables.

2.4.2 Single-engine Hybridization

In case of single-engine hybrid systems, a single MT approach is extended with a different MT approach. Modifications are possible in various stages of the pipeline. They can be classified into RBMT systems that are extended with data driven methods or the opposite. In the first option, data information and statistical techniques are integrated into a rule-based architecture (Section 2.4.2) while, in the latter case linguistic rules are added to a data driven architecture (Section 2.4.2).

Statistical Machine Translation with Rule-based Modules

The extension of SMT systems with linguistic information can be implemented by either using linguistic rules for pre-editing or modifying its core system. Pre-editing can be used to prepare data before training an SMT system. For example, data sparseness can be a major issue in SMT, especially for morphologically rich languages. To solve such issues, morphological preprocessing can be used to reduce data sparsity using tokenization, lemmatization, part-of-speech tagging or both.

Also, syntactic information can be used for preprocessing or decoding. The word order problems can be tackled by implementing syntax-based transformations that apply transformation rules to the source language parse tree to make the order of the source sentence closer to the target sentence. System core modifications can be carried out by adding RBMT information to the phrase tables (See Section 2.3.2) and by using factored translation schemes (See Section 2.3.2).

Rule-based Machine Translation with Statistical Modules

Rule-based systems can be extended with data driven techniques at several levels. A first, very straightforward, way to use statistical techniques in rule-based MT is to pre-edit the language resources it relies on, such as its dictionaries and grammar rules. New dictionary entries, for instance, can be learned by automatically extracting them from monolingual or parallel corpora. In this case, monolingual corpora can find missing entries in the dictionaries while parallel corpora can find translation candidates. An example of such a system is generation-heavy MT (GHMT) (Habash et al., 2009) that is pre-edited by enriching the dictionary with phrases from an SMT system. Besides dictionary entries, the systems′ grammar rules can by automatically extracted from corpora.

A final approach for using data driven techniques in rule-based systems includes the modification of its core. This is usually done by adding probabilistic information to the various phases in the translation pipeline. For instance, the TectoMT system (Žabokrtský et al., 2008) is such a hybrid MT system.

In TectoMT, it is possible to integrate elements of both statistical and rule-based MT into a modular framework that can be adapted to include various NLP tasks in a single pipeline. Similar to rule-based MT, the system handles translation over three phases: analysis, transfer, and synthesis. The analysis and generation phases are primarily modular, allowing for independent, statistical and rule-based NLP tools and processes to be implemented into the pipeline. The transfer phase that links the analysis and generation modules is primarily statistical.

In the QTLeap project, to examine the merits of further linguistic processing, hybrid MT systems were developed that combine linguistic rules with statistical methods. For this, our base approach to a translation system was the TectoMT system. In the next chapter, we describe the hybrid TectoMT systems that were created for the Dutch-English language pair within the QTLeap project.

CHAPTER 3

Machine Translation for Dutch and English in the QTLeap Project

The goal of this chapter is to give an overview of the the development of the Dutch–English MT systems within the QTLeap project. It provides the background for further work in parts 2 and 3 of the thesis. Most of the work in Chapter 3 was carried out by project partners and is described here in detail to ensure that the later parts of the thesis can be properly understood in the context. In particular, several modules for translating between English and Dutch where developed by project colleagues at the Charles University in Prague. Special thanks go to Ondřej Dušek who contributed most to the creation and integration of the different modules.

3.1 Introduction

In this chapter, the MT systems that were developed in the QTLeap project for the Dutch–English language pairs are presented. The project aimed to improve MT by way of including linguistic information while employing the strengths of statistical approaches. To examine the merits of further linguistic processing, hybrid MT systems were developed that combine linguistic rules with statistical methods. During the project, experiments were carried out following a processing pipeline of analysis, transfer, and generation while using both statistical and rule-based components.

Our base approach to a translation system was the TectoMT system (Žabokrtský et al., 2008). An advantage of using TectoMT is that it is language-independent in many aspects. It uses multilingual standards for morphology and syntax annotation and language-independent base rules, which facilitates the implementation of new languages. Furthermore, it is modular, enabling a smooth incorporation of various pre-existing language-specific tools.

In QTLeap, new MT systems have been developed by combining existing modules from TectoMT (English analysis, generation, and transfer), and Alpino (van Noord, 2006) (Dutch analysis and generation) and creating new ones where necessary. The existing modules for Dutch are described in Section 3.3 and the general architecture of TectoMT are outlined in Section 3.2. Then, in Section 3.4 and Section 3.5, the language specific components for the Dutch–English MT systems are described.

3.2 TectoMT

TectoMT employs a hybrid approach to MT containing both statistical and rule-based components implemented within the Treex NLP framework (Popel and Žabokrtský, 2010). It contains modules for several NLP tasks, such as sentence segmentation, tokenization, morphological analysis, POS tagging, parsing, named entity recognition, anaphora resolution, tree-to-tree translation, natural language generation, word-level alignment of parallel corpora, and transfer. The system follows an analysis-transfer-generation pipeline, as can be seen in the general architecture shown in Figure 3.1.

TectoMT makes use of four layers of structural description: raw text or word layer (w-layer), the morphological layer (m-layer), the analytical layer (a-layer), a surface syntax layer containing dependency trees, and the tectogrammatical layer (t-layer), a syntactic layer which describes the linguistic structure of the sentence (Sgall et al., 1986). In the MT pipeline of analysis-transfer-generation, the input text is gradually converted from one layer to another. The general process, which is mostly language independent, will be described in the following sections. After that, in sections 3.4 and 3.5, we describe the implementation of Alpino in TectoMT and the modules that were created in QTLeap to facilitate this process.

3.2.1 Analysis

In the analysis phase of TectoMT, the input sentence is converted to an a-layer tree that is subsequently converted into a t-layer tree. An example of such an a-layer tree (a-tree) and a t-layer tree (t-tree) can be found in Figure 3.2. In TectoMT, first, a dependency parser is used to

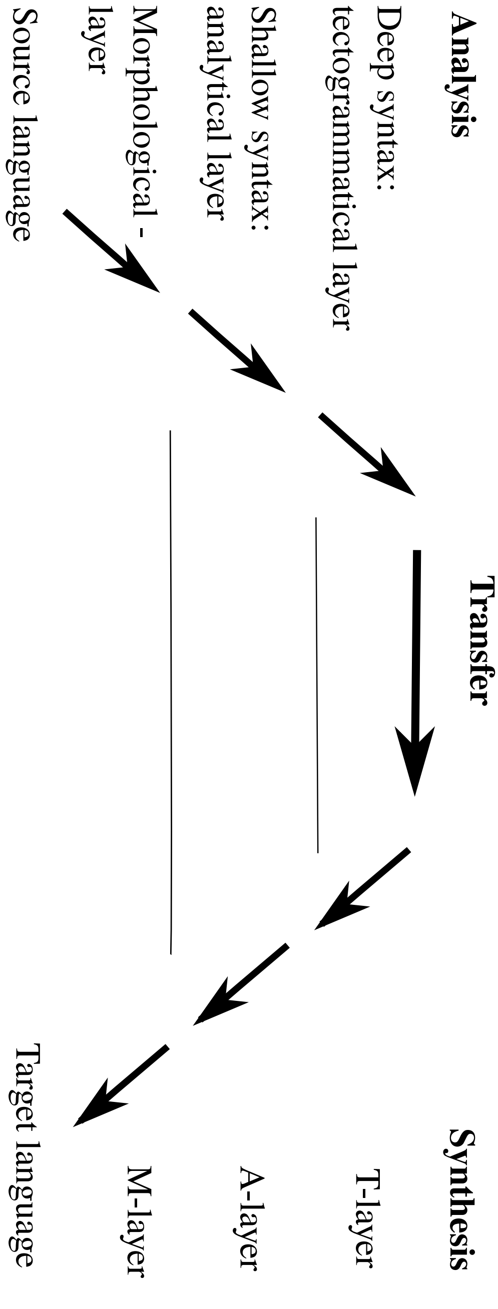

Figure 3.1 | *TectoMT architecture*

build an a-tree after which a t-layer analysis is created by a rule-based conversion of the a-tree.

The a-layer is a surface syntax layer, which includes all tokens of the sentence, organized as nodes into a labeled dependency tree. Before a-layer parsing, preprocessing steps include sentence segmentation, tokenization, lemmatization, and morphological tagging. The a-layer parsing itself can then be performed by a dependency parser so that each a-layer node is annotated, among others, with the following types of information:

- The **inflected word form** as it appears in the original sentence, including capitalization,
- The **lemma**, the base form of the word, for instance, infinitive for verbs, nominative singular for nouns,
- A **morphological description of the word form**: all morphology information describing the word form used in the sentence,
- A **surface dependency label** corresponding to commonly known syntactic functions such as subject, predicate, object, and attribute.

While, in the a-tree, each token of the input sentence corresponds to a node, its dependency tree (t-tree) only contains nodes for words bearing lexical content (main verbs, nouns, adjectives and adverbs) and coordinating conjunctions. Each t-tree node has a lemma (t-lemma), a semantic role label (functor), and a set of attributes expressing grammatical meaning (grammatemes). The a-tree is gradually transformed into a t-tree by modules that perform the following tasks:

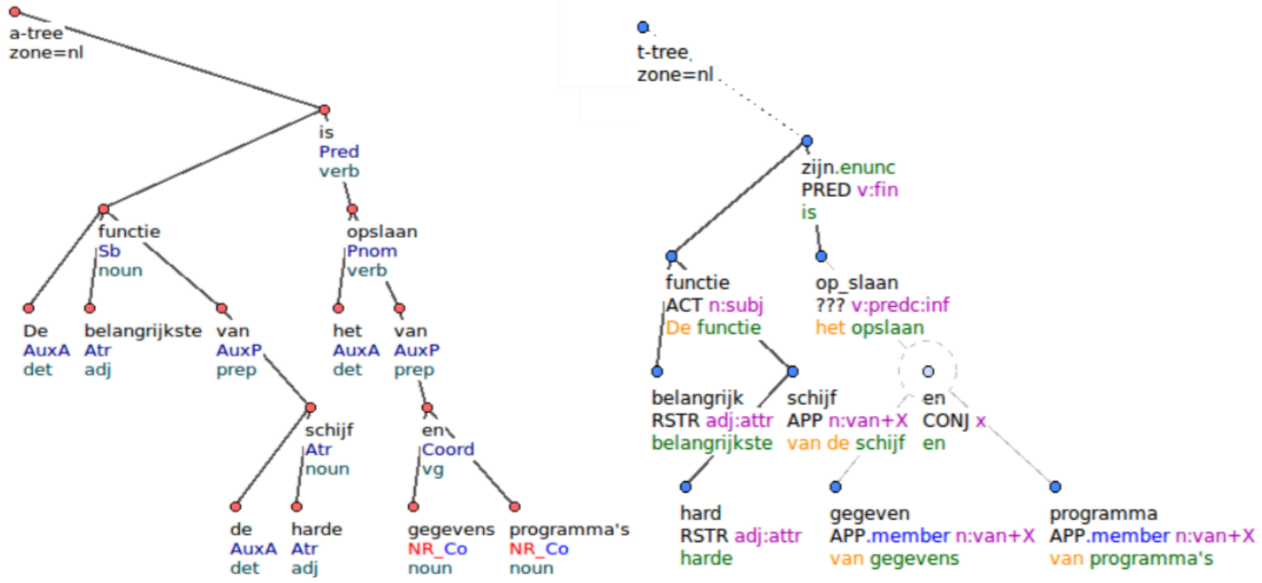

Figure 3.2 | *Example of an a-tree (left) and a t-tree (right) for the Dutch sentence: "De belangrijkste functie van de harde schijf is het opslaan van gegevens en programma's" (The main function of the hard-disk is storing data and programs). In the nodes of the t-tree, the first description line shows the lemma, the second line contains morpho-syntactic information (purple) and the third line lists the corresponding surface tokens.*

- **Removal of auxiliary words:**
  In the first step, nodes of auxiliary words are removed from the tree so that only content words have their own nodes on the t-layer. Links are retained in t-nodes to which they relate (e.g., prepositions are linked to nouns, auxiliary verbs to the lexical verb).

- **Surface lemma to t-lemma conversion:**
  A lemma in the t-layer is usually identical to the surface lemma but can be merged (e.g., personal pronouns or possessive adjectives derived from nouns) or modified. For example, lemmas of personal pronouns are substituted with the tag #PersPron while reflexiva tantum verbs, separable and phrasal verbs as well as multi-word surnames are combined (e.g., screw_up).

- **Formeme assignment:**
  TectoMT's t-trees include *formemes*. They consist of a concise description of morpho-syntactics of each node (Dušek et al., 2012). They are composed of coarse-grained part-of-speech tags, prepositions or subordinate conjunctions, and a coarse-grained syntactic form. This adds up to a simple human-readable string, such as *v:to+inf* for infinitive verbs or *n:into+X* for a prepositional phrase.

- **Functor assignment:**
  Here, functors (semantic roles) are detected and marked for each node. A t-node can contain over 60 different functors, or semantic role labels, such as ACT (actor/experiencer), PAT (patient/deep object), TWHEN (time adverbial), RSTR (modifying attribute) and so on.

- **Grammateme assignment:**
  Grammatemes are semantically indispensable morphological categories. They belong to a set of linguistic features relevant to the meaning of a sentence (e.g., semantic part-of-speech, number for semantic nouns, grade for semantic adjectives and adverbs, or person, tense, and modality for semantic verbs).

- **Actor reconstruction:**
  In cases where the subject or actor personal pronoun is not present on the surface, they are reconstructed for example for pro-dropped subjects, imperatives and passive clauses where the actor is not expressed explicitly.

- **Coreference resolution:**
  Coreference links are introduced to connect anaphora with their antecedents.

The result of the aforementioned steps is a syntactic and semantic representation of the input sentence (t-layer) that can now be converted to an equivalent representation of the target sentence. We describe this process in the following section.

3.2.2 Transfer

Using the t-layer representation in MT allows for the separation of the problem of translating a sentence into three relatively independent simpler subtasks: the translation of t-lemmas, formemes, and grammatemes (Žabokrtský, 2010).

The transfer phase of t-lemmas and formemes is further divided in two steps. In the first step, maximum entropy context-sensitive translation models (Mareček et al., 2010) produce translation variants for each t-tree node's t-lemma and formeme. For each t-lemma and each formeme in a source t-tree, the translation model assigns a score to all possible translations on the basis of observations in the training data. This score is a probability estimate of the translation variant given the source t-lemma, its formeme and its context. It is calculated as a linear combination of a discriminative TM, where prediction is based on features extracted from the source tree, and a dictionary TM, comprising a dictionary of possible translations with relative frequencies without context information. In the second step, hidden markov tree models (HMTM) (Žabokrtský and Popel, 2009) are used to combine the translation model's predictions in a Viterbi search. The hidden Markov tree models are similar to standard (chain) hidden Markov models but operate on trees (see Section 8 for a further description of such models).

The translation of grammatemes is much simpler than the translation of t-lemmas and formemes since abstract linguistic categories such as tense and number are usually paralleled in the translation. Therefore, a set of relatively simple language-specific rules (with a list of exceptions) is sufficient for this task.

The main TectoMT translation models that handle the transfer of source to target t-layers are statistical and can be trained for any language. Training the statistical components requires training data. In QTLeap parallel treebanks were created by using automatic annotation up to the t-layer on both languages. For this, MT analysis tools are used. The annotation pipeline starts with a parallel corpus and ends with a parallel treebank containing pairs of t-trees aligned on the level of t-nodes. The analysis phase of the pipeline mimics the one used in a translation process, including tokenization, lemmatization, morphological tagging, dependency parsing to a-layer and a conversion to t-trees.

In the word-alignment stage, pairs of t-trees are constructed. First, words are aligned using GIZA++ (Och and Ney, 2000) after which the alignments are projected to the corresponding nodes in the t-trees. Then, additional heuristic rules are used to align t-nodes that have no counterparts on the surface. [1]

3.2.3 Generation

In the generation phase (referred to as "synthesis" in TectoMT), rule based components gradually convert the target language representation into a shallow one, which is used to generate text. The generation modules in the pipeline are language specific and in general, include solving the following problems:

- Word ordering imposed by the syntax of the target language,
- Morphological agreement,
- Addition of prepositions and conjunctions,
- Compound verb forms,
- Addition of function words,
- Addition of interpunction,
- Inflection of word forms based on morphological information from the context,

[1] Note that once a parallel treebank for a given language pair has been constructed, it can be used for training translation models in both translation directions.

- Capitalization of words that start a sentence.

3.3 Alpino

Alpino (van Noord, 2006) is a collection of tools and programs for parsing Dutch sentences into dependency structures, and for the generation of Dutch sentences on the basis of an abstraction of dependency structures.

Dependency structures contain information about the grammatical relations (arcs) between a word and other words (nodes) with which it can form a constituent.

In Alpino, dependency structures are represented by attribute-value structures including information of the head word (hd) of that dependency structure, as well as attributes such as subject (su), direct object (obj1), secondary object (obj2), modifier (mod) and determiner (det) for each of its dependents. Each word has one or more head words and zero or more dependents. In addition, each word has a part-of-speech tag and a marker for its begin and end positions in the sentence associated with it. An example of such a dependency tree for a Dutch sentence is given in Figure 3.3.

A description of Alpino dependency structures can be found in van Noord et al. (2011, 2013) and Van Eerten (2007) where the dependency structures are described that are derived from the CGN and Lassy Dutch corpora.

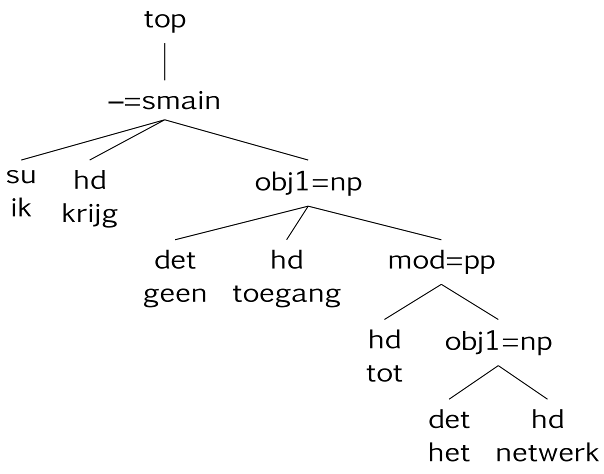

Figure 3.3 | *Dependency Tree of the Dutch sentence "Ik krijg geen toegang tot het netwerk" (I cannot get permission to enter the network)*

3.3.1 The Alpino Parser

The Alpino parser is an implementation of a stochastic attribute value grammar (van Noord, 2006). It includes an attribute-value grammar inspired by head-driven phrase structure grammar (HPSG) (Pollard and Sag, 1994), a large lexicon, and a maximum entropy disambiguation component.

The grammar contains over 800 grammatical rules, organized in an inheritance network, expressed in the attribute value grammar notation. It takes a constructional approach, with rich lexical representations and a large number of detailed, construction specific rules. A very large lexicon (over 300,000 entries) combined with a large set of heuristics to recognize named entities as well as unknown words and word sequences provides attribute value structures for the words in the input. To judge the quality of (partial) parses, the algorithm refers to a maximum entropy disambiguation model which was trained on a gold-standard treebank containing approximately 10,000 sentences.

3.3.2 Abstract Dependency Trees

To accommodate processing in an MT pipeline, the dependency structures that are input for generation should be less specific. As they contain less information than regular dependency trees, we refer to them as abstract dependency trees (ADTs). Similar to normal dependency trees, ADTs contain a syntactical representation of a sentence in the form of a tree. They model the grammatical relations between lexical items and categories built from lexical items. Each node in the tree contains information about its main word and each dependent can, by itself, contain a dependency structure. An ADT is, therefore, an abstraction of the dependency structure format of Alpino. It can also be constructed from a dependency tree that is the output of the Alpino parser by removing or reducing certain information.

The most important difference between CGN and Lassy dependency structures (Van Eerten, 2007; van Noord et al., 2011, 2013) and ADTs is that ADTs do not contain information about word order. While in the parsing output, words are annotated by their sentence position to identify the word in the sentence, in generation, this information would not be used. It is the task of the generator to find realizations of the dependency structure that are grammatical and fluent.

Another difference with regular parsing output is that words are not represented by their inflected form used in the sentence but instead by their stem and lexical information. The inflected words are omitted and will be constructed by the generator.

Some Dutch verbs have particles that can be connected to or be separated from that verb. An example of such a verb is “weggaan" in Example 3a and 3b.

(3) a. de daders zouden ongemerkt kunnen weggaan
(The perpetrators could go unnoticed)

b. de daders zouden ongemerkt weg kunnen gaan
(The perpetrators could go unnoticed)

If the particle is separated from the verb, it is also a separate dependent of the verb, so that every word that occurred in a sentence has a relation in the dependency structure. The choice of connecting a particle to a verb or not is usually a matter of fluency, and should, therefore, be made by the generator.

A final aspect that is not specified in the abstract dependency structure is punctuation. Its specification would not provide any useful information to the generator and is therefore omitted.

Similar to normal dependency structures, ADTs have edges between their nodes with a dependency label. All nodes are associated with a category while the leaf nodes represent the words. The latter are lexical nodes representing words as a root, a sense, a part of speech tag, and a set of attribute/value pairs. An example of such a lexical node is:

```
adt_lex(np,suggestie,suggestie,noun,[rnum=sg])
```

Here the *category* (np), *root* (suggestie), *sense* (suggestie), *part-of-speech tag* (noun), and *attributes* (rnum=sg) of a lexical item are noted. The sense of a lexical item includes the root form of the word and can have additional information about a lexical item to select for a particular reading. For instance, for words with separate particles, the particles are frequently listed in the sense (e.g., the sense of “rood aanlopen" *(turn red)* is *rood-loop_aan*). ADTs allow for the underspecification of some of the attributes. For instance, the omission of number attributes can result in a singular or plural noun.

3.3.3 The Alpino Generator

The ADTs, that were described in the previous section, can be used as input to the Alpino generator (De Kok, 2010; De Kok et al., 2011; De Kok, 2013). In the generation process, the same attribute-value grammar is used as for parsing (although some rules are excluded which allow the analysis of fragmentary or unexpected input). It maps the input ADTs to surface strings, taking care of word order, agreement, inflection, and punctuation.

The Alpino generator makes use of chart generation(Shieber, 1988; Kay, 1996). Generally, a given dependency structure can be realized by more than a single sentence. Therefore, in the Alpino generator, a statistical fluency component based on maximum entropy is used to select the most natural and fluent realization for a given input structure. More details on this process can be found in De Kok (2013).

The generator has been developed from a monolingual point of view. During the QTLeap project, it was therefore tested against real-world input before implementing it in the MT system. A number of problems came to the surface and to make it suitable for machine translation, subtle adaptations and improvements were necessary. A description of these tests and adaptations are described in Section 7.3.

3.4 TectoMT for English to Dutch

In this section, we describe the TectoMT system that translates English to Dutch. As can be seen in Figure 3.4, the pipeline starts with the analysis of the English sentence (Section 3.4.1). Then follows the English to Dutch transfer phase (Section 3.4.2) after which Dutch sentences are generated (Section 3.4.3). In TectoMT, modules for English analysis were readily available while the transfer remains largely language independent. For the generation of the Dutch output sentences, however, the Alpino generator was used. Therefore, new modules had to be developed that ensure that the TectoMT trees are converted to abstract dependency trees that are input to the Alpino generator (ADTs).

3.4.1 Analysis of English Input Sentences

The English analysis follows the annotation pipeline used for the CzEng 1.6 parallel corpus (Bojar et al., 2016b), using a rule-based tokenizer, a statistical part-of-speech tagger (Spoustová et al., 2007) and dependency parser to a-trees (McDonald et al., 2005), followed by mostly rule-based post-processing. The t-layer conversion starts from the a-tree and closely follows the process outlined in Section 3.2.1. Language dependent modules for English include:

- The assignment of t-lemmas focuses on phrasal verbs, personal pronouns, and the negation particle "no",

- Formeme assignment reuses language-independent code with the help of the Interset morphology abstraction layer (Zeman, 2008). Formemes that are specific for English nouns include their syntactic position (subject, direct object, indirect object, attribute, possessive),

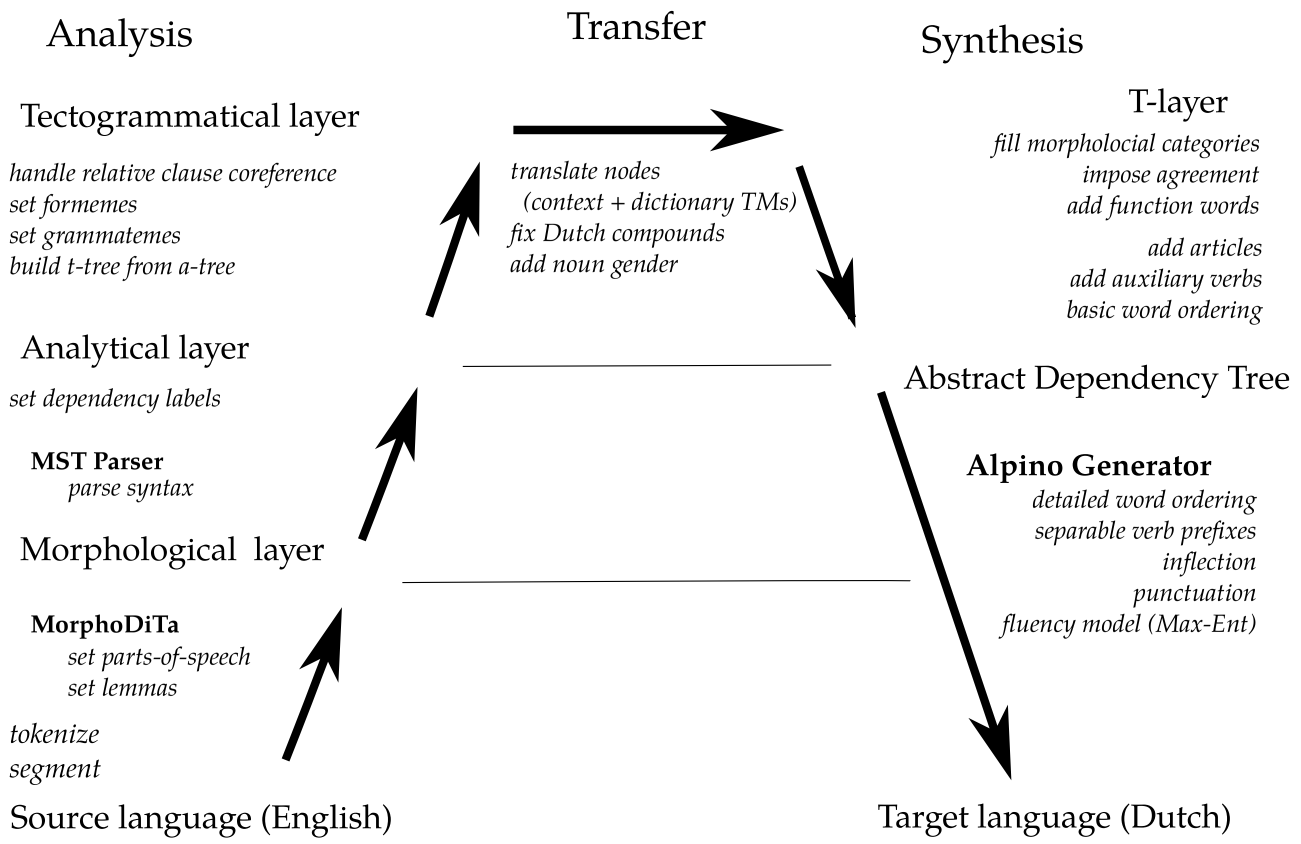

Figure 3.4 | *TectoMT architecture of EN→NL translation*

- Functors are assigned by rule-based modules using auxiliary words, lemmas, and formemes,
- Grammateme assignment for English uses language-specific rules which are mainly based on surface morphology and presence of auxiliaries,
- Generated actor (subject) nodes are added in imperative clauses and in control constructions (with a verb that governs an infinitive clause). In the latter case, the added subjects of infinitive clauses have a coreference link to the subject or the object of the governing verb, based on the type of the control construction.
- Simple modules for coreference of relative and possessive pronouns.

3.4.2 Transfer from English to Dutch

In the transfer phase, where English t-trees are converted to Dutch t-trees, some language specific rule-based transfer modules were added in QTLeap. For example, a module is added that handles the translation of English noun groups into a single Dutch compound. Here, the post-processing module finds cases where decisions made by the translation model result in redundancies in the Dutch sentence correcting typical translation mistakes in the case of compounds. For example, words such as "*web page*" translated to "*webpagina pagina*" and "*programming language*" was translated as "*programmeertaal taal*".

The translation of English into Dutch relative clauses is handled by a module that forces them to be translated as relative clauses. The translation model operating on formemes often maps a relative clause onto a plain finite clause, which can cause errors in word order. Examples of such improvements, as compared to a system that does not use this module, can be found in example 4. The translated sentence is more fluent than the translation by a base system.

(4) a. *English original:*
Open the file, application or message where you want to paste the text.

b. *No module:*
Open het bestand de toepassing of de boodschap die wil je waar de tekst geplakt.
(lit. *'. . . which you want where the text pasted'*)

c. *Mapping module:*
Open het bestand, de applicatie of het berichtwaar je de tekst wil plakken.
(lit. *'. . . where you want to paste the text'*)

A further improvement on the transfer level ensures that the linguistic features are checked after transfer. This is necessary because, if a t-tree node is translated to a different part-of-speech, some of them can become invalid. This could, for instance, cause nouns to be inflected as verbs, and so on.

3.4.3 Generation of Dutch Output Sentences

The Dutch sentences are generated by the Alpino generator. To make it fit into TectoMT system, first the TectoMT trees are adapted to appropriate input for the generator. In the Dutch generation component, the t-tree resulting from the transfer phase is first converted into an ADT with rule-based modules. This process is shown in Figure 3.4. The ADT is then passed to the Alpino generator (De Kok, 2013), which handles the creation of the target sentence including inflected word forms.

A number of rule-based blocks in TectoMT take the output of transfer, t-trees, and convert them to ADTs (Section 3.3.3). The TectoMT conversion part of the pipeline consists of many language-independent modules, including morphology initialization and agreements (subject-predicate and attribute-noun), insertion of prepositions and conjunctions based on formemes and insertion of punctuation. Language-specific modules for Dutch include:

- Insertion of infinitive particles (e.g., om-te) and reflexive particles (e.g., zich),
- Insertion of articles,
- Insertion of auxiliary verbs based on grammatemes (syntactic features),
- Basic word ordering (moving predicates to the end of the clause).

At the end of this first stage, the shape of the trees is adjusted to conform to the ADT format. The Alpino generator then takes the ADT and generates target sentences as described in Section 3.3.3.

Before implementing it in TectoMT, the Alpino generator has been made more robust to handle noisy input ADTs from incorrect parses and translation model decisions. In QTLeap, a number of heuristics is implemented, which ensure that if a dependency structure cannot be realized by the generation component, it splits the structure into multiple parts to facilitate generation. A general back-off strategy is applied in the generation algorithm to the parts of the input structure in case no full generation is possible.

In QTLeap a number of heuristics are used that rearrange the input ADTs to match the expectations implicit in the Alpino grammar and lexicon. Currently, there are 391 of such transformations, ranging from fairly generic ones (typically for particular syntactic constructions, e.g., to ensure that subject control is specified in the correct way for subject control and raising verbs), to transformations for particular English expressions (e.g., "if so" should not be translated as "als het is" but is now translated as "zo ja"), transformations for context-sensitive translations (e.g., "check for" must be translated as "controleren op" and not "controleren voor"), correcting neuter and non-neuter determiners, adding prepositions in particular cases ("press X" should not be translated as "druk X", but as "druk op X" in Dutch), and domain-specific translations ("driver" should not be translated as "chauffeur" in the context of computer software).

3.5 TectoMT for Dutch to English

In this section, the TectoMT system for the opposite translation direction, from Dutch to English, is described. Figure 3.5 shows the Dutch to English translation pipeline. Similar to the English to Dutch translation scheme, the pipeline starts with the analysis of the Dutch sentence (Section 3.5.1 followed by the Dutch to English transfer phase (Section 3.5.2). Finally, Dutch sentences are generated (Section 3.5.3)

In QTLeap, the Alpino parser was implemented within the TectoMT system. To facilitate this, several modules were developed that convert the output of the parser to the TectoMT structures. The modules for English generation were already available.

3.5.1 Analysis of Dutch Input Sentences

For Dutch analysis, the Alpino parser (see Section 3.3.1) is used. In the Dutch-English TectoMT system (Figure 3.5), first, rule-based modules convert the Alpino-parsed dependency trees in order to make them similar to the TectoMT surface dependency trees (a-trees). This involves decoding Alpino parts-of-speech to Interset (Zeman, 2008) with a converter that was created for the Alpino part-of-speech tagset.

The remaining conversion to t-trees is similar to the TectoMT systems of the other languages. When possible, language-independent blocks are used. This is the case for the initial construction of t-trees, handling of relative clauses and reflexive pronoun co-reference, and modules assigning grammatical features.

Elsewhere, language-specific modules were created for Dutch analysis. These modules include, for example, the selection of meaning-bearing elements that need to be included in the t-tree, *functor* (semantic role) detection, handling of negation, personal and reflexive pronouns, handling of multi-word surnames, the assignment of formemes and information on the inflection of words. These language-specific modules reuse language-independent code where appropriate.

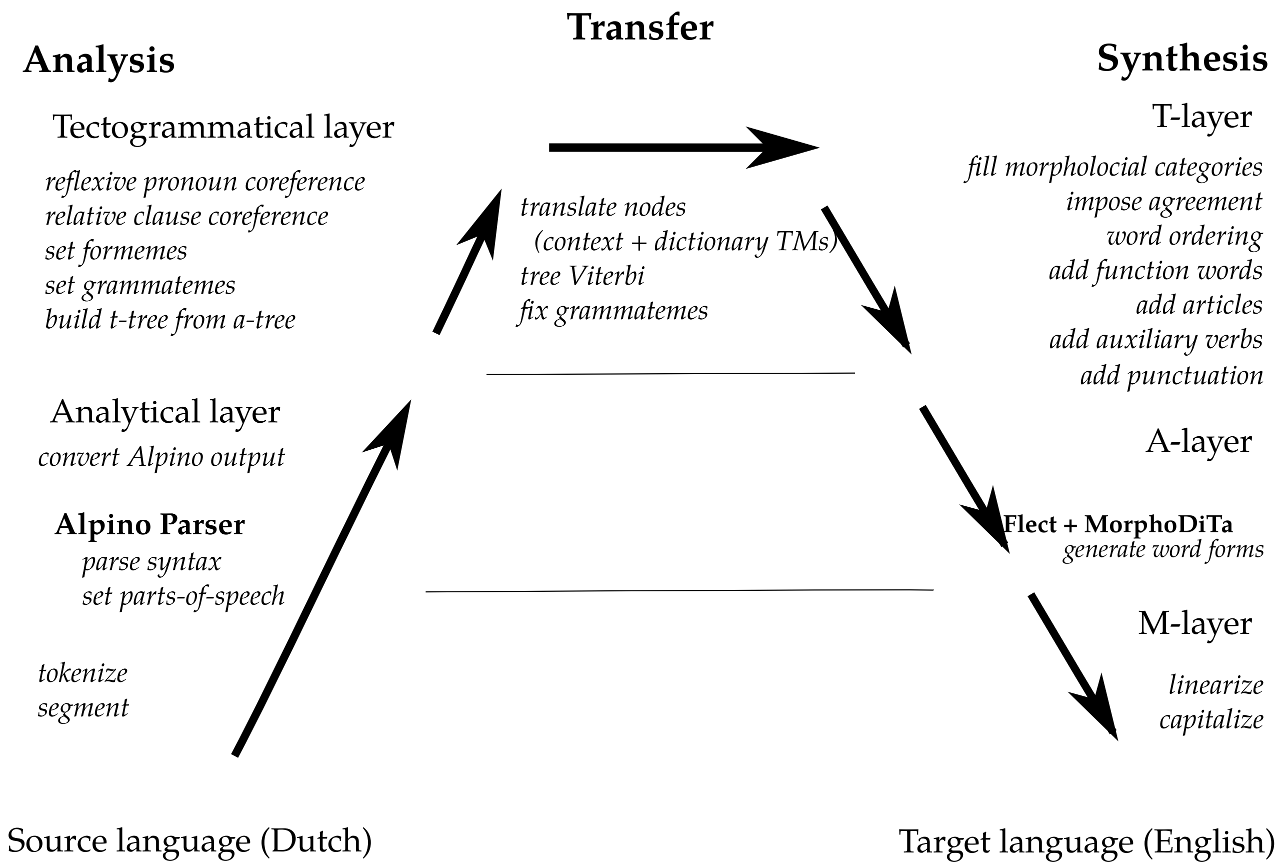

Figure 3.5 | *TectoMT architecture of NL→EN translation*

3.5.2 Transfer from Dutch to English

In the transfer phase in which Dutch t-trees are converted to their English equivalent, probabilistic lexical choice models and dictionary translation models are used as described in Section 3.2.2 and Section 3.4.2.

3.5.3 Generation of English Output Sentences

The English generation pipeline was already available in TectoMT and globally conforms to the general setup presented in Section 3.2.3. The (rule-based) parts that were specifically designed for English involve, ordered as they appear in the generation pipeline:

- A rule-based word ordering module that enforces the SVO order in indicative clauses, as well as the vSVO order in interrogative clauses,
- Detection of surface morphology based on grammatemes, including the enforcement of subject-predicate agreement,
- Modules that insert English-specific auxiliary words, infinitive and phrasal verb particles, possessive markers ('s), and articles,
- Insertion of auxiliary verbs based on grammatemes,
- Heuristic rules for the necessary English punctuation (clause-initial punctuation and punctuation for some dependent clauses),
- Word form generation,
- Rules for indefinite article phonetics (distinguishing "a" and "an").

3.6 Evaluation

This chapter provided a description of the MT systems for Dutch and English that were developed in the context of the QTLeap project. The MT systems are a result of the combination of the TectoMT system and the Alpino parser and Generator and new modules. To measure the performance of these systems, an extensive evaluation was performed with the other partners of the QTLeap project. We describe this evaluation and its outcomes in the following chapter.

The MT systems provide a good framework for further experimentation with transfer-based MT with interlingual representations of words. We therefore use them for experiments on lexical choice in Chapter8.

CHAPTER 4

Evaluation of Hybrid Machine Translation Systems for Dutch–English

This chapter describes the evaluation of the MT systems developed in QTLeap. Both automatic evaluation measures were applied and a task-based evaluation with human subjects was performed. Firstly, the SMT baselines for NL→EN and EN→NL were developed by the author of this thesis using existing tools.

Secondly, together with project partners from Higher Functions, a real usage scenario was developed and several experiments were undertaken with human subjects in order to estimate the quality of the various translation systems. Gaudio et al. (2016) For this evaluation, the writer of this thesis was responsible for the Dutch components of the evaluation. This included the localization into Dutch of the experimental framework, and the recruitment and instruction of the volunteers.

Thirdly, a detailed linguistic analysis of the translation errors by the various translation systems was performed by the author of this thesis, using the framework developed by project partners at the DFKI.

4.1 Introduction

In the previous chapter, we described the MT systems for Dutch and English that were developed in the QTLeap project. The MT systems are a result of the combination of the TectoMT system and the Alpino parser and Generator. For the evaluation of these MT systems, several approaches were used ranging from automatic measures, human error annotation and task-based evaluation.

We describe the evaluation methods used in Section 4.3. For the training and evaluation the MT systems, parallel corpora are required. The datasets that were used for the Dutch and English language pair are described in Section 4.2.

In order to ensure the comparability of the results of our system for Dutch and English, we compare it with an SMT system that does not use any linguistic processing. Although NMT is currently the state-of-the-art, the creation of baseline systems was one of the first steps in the project and, at the time, SMT was still state-of-the-art. Also, as it was the goal of the project to measure the effects of combining linguistic with statistical information, an SMT system is a good reference point. The creation of this system for the Dutch–English language pair is described in Section 4.4.

We describe the settings that were used for training TectoMT in Section 4.6. In addition, in the QTLeap project, an extra system was developed combining a linguistically motivated MT system (TectoMT, Section 3.2) with a phrase-based SMT system (Moses (Koehn et al., 2007), Section 4.4.1), in an effort to improve upon both of them. The creation of the Chimera system and the automatic evaluation of the MT systems was done by other project members and has been added here for completeness. Details on this Chimera system (Bojar et al., 2013; Bojar and Tamchyna, 2015) can be found in Section 4.7. In Section 4.8 of this chapter, the results of the different types of evaluation for the MT systems can be found.

4.2 Parallel Corpora for Dutch and English

The creation of MT systems requires parallel data: collections of aligned sentences. In such corpora, each sentence in one language is matched with its corresponding sentence in the other language. For the development, training and testing of the MT systems that are described in Chapter 3, we made use of the following parallel corpora:

- **The Dutch Parallel Corpus** (Macken et al., 2007) is a parallel corpus for Dutch, French and English consisting of more than 10 million words. It contains five different text types, namely: literature texts, journalistic texts, instructive texts, administrative texts and external communication texts and is balanced with respect to text type and translation direction. All texts in the corpus are sentence-aligned and further enhanced with basic linguistic annotations (lemmas and word class information). For Dutch-English 180,000 parallel sentences are available.

- **Europarl** (Koehn, 2005) is a large and well known publicly available parallel corpus. It consists of proceedings of the European Parliament, dating back to 1996. Altogether, the corpus contains about 30 million words in the 21 official languages of the

European Union. For Dutch-English 2 million parallel sentence pairs are available.

- **The OPUS corpus** (Tiedemann, 2012)[1] is a growing collection of publicly available parallel corpora. Here, free on-line data is collected, converted and aligned. The corpora are based on open source products. From the OPUS corpus several subcorpora have been considered for the creation of the Dutch-English MT systems:

  - **The EU-Constitution corpus** is a parallel corpus containing texts from the European Constitution, including 21 languages. For Dutch-English translation, 10,000 sentence pairs are available.

  - **The KDE4 Localization data corpus** contains the localization files of KDE, an open source software community, and supports more than 80 languages. As the corpus contains in-domain data for the informatics genre, it could be useful for the translation of our test data (a part of the QTLeap corpus described below), which belongs to the same domain. However, most messages are very short and mainly consist of only one sentence or just a term or phrase. For Dutch-English 192,000 parallel sentences were derived.

  - The sub-corpus of **PHP manuals** is derived from the HTML version of the online documentation of the scripting language PHP. It contains around 3.5 million words in a total of 21 languages. For the Dutch-English baselines 32,000 sentence pairs were used.

[1] `http://opus.lingfil.uu.se/`

- **The QTLeap corpus** (Osenova et al., 2015) consists of translations of customer data of the Higher Functions company in its project languages (Basque, Bulgarian, Czech, Dutch, German, Portuguese and Spanish). In this IT helpdesk scenario, a user query in one language is translated into English, an answer is found in the English database, translated into the target language and sent back to the user.

  As the data was collected through a real usage application (the original Portuguese interactions were first translated into English to serve as a starting point for the QTLeap project), it is composed of naturally occurring utterances that were produced by users while interacting with the service. The corpus is composed of 4,000 question and answer pairs, divided into four 1,000 pair batches, in the domain of computer and IT troubleshooting for both hardware and software.

- **The QTLeap News Corpus** is a sample extracted from the corpus made available by the annual workshop on SMT [2] from the news domain. To this end, 1104 English sentences and their corresponding human translations into Czech, German and Spanish from WMT 2012 and WMT 2013 translation tasks were taken as the basis. The English sentences were then professionally translated to Bulgarian, Dutch, Portuguese and Basque via a subcontract from QTLeap.

[2] WMT, see `http://www.statmt.org/`

4.3 Evaluation Types

A straightforward and commonly used evaluation method in MT is the application of automatic metrics. In Section 4.3.1, we describe the automatic measures that we used in the project. A second evaluation was performed manually by identifying systematically occurring translation errors that are related to linguistic phenomena. The results of this evaluation were used to locate existing problems during the development of the MT systems. In Section 4.3.2 we elaborate on this process.

The focus of the third evaluation scheme is to verify the usefulness of the translations. This is done by assessing the added value in terms of their impact on the performance of a question answering (QA) system. For this purpose, the constructed MT systems are embedded in a multilingual call center. This is a real usage scenario where high-quality machine translation is used to support efficiency and economy of scale, thus serving as a real usage test for the evaluation of the results of the MT systems. We describe this real usage scenario in Section 4.3.3

4.3.1 Automatic Evaluation

In automatic evaluation, heuristic methods are used to compare the output of an MT system with a reference translation. The most widely used evaluation metric is the bilingual evaluation understudy (BLEU) (Papineni et al., 2002). The BLEU method computes n-gram matches that are computed sentence by sentence. BLEU uses a modified precision score to measure the overlap between candidate translations produced by the system and reference translations. Usually, the geometric mean for scores up to 4-grams are reported. BLEU scores range from 0 to 1, where 1 is the highest reachable score. This number, however, can only be reached if all its substrings can be located in one of the reference texts. It should, furthermore, be calculated on a large test set with several reference translations. A disadvantage is that such large test sets are not always available for each language pair.

First, the maximum number of times a word occurs in the reference translations is counted. The candidate counts then are clipped by their corresponding reference maximum value and added for all the candidate sentences. These counts are then divided by the number of candidate n-grams in the translated text to compute p_n which is defined as:

$$p_n = \frac{\sum_{C \in Candidates} \sum_{n-gram \in C} \text{count}_{clip}(\text{n-gram})}{\sum_{C' \in Candidates} \sum_{n-gram' \in C'} \text{count}(\text{n-gram}')} \tag{4.1}$$

where C runs over the entire set of candidate translations, and count_{clip} returns the number of n-grams that match in the reference translations.

The metric then computes the similarity between MT output and its reference translation as follows:

$$BLEU = BP \times exp\left(\frac{1}{N}\sum_{n=1}^{N} \log p_n\right) \tag{4.2}$$

Which is adjusted by a brevity penalty BP:

$$BP = \begin{cases} 1 & \text{c} > \text{r} \\ e^{(1-r)/c} & \text{c} \leq \text{r} \end{cases}$$

where c is the length of the MT output, r is the length of the reference translation.

The BLEU metric has been widely used in MT, mainly because manual evaluation is too expensive and because it was known to be correlating more to human judgment as compared to other automatic measures. However, it has also been the subject of much criticism. The main point of critique is that it may not correlate well with human judgment. For example, Callison-Burch and Osborne (2006) have compared BLEU's correlation with various SMT systems and a rule-based system and found poor correlation with human judgments.

Additionally, Doddington (2002) and Turian et al. (2006) have reported on experiments that show that the best correlation with human judgments was found with just a single reference translation per test sentence and that the addition of more references does not help. This goes entirely against the rationale behind having multiple references which should capture natural variation in word choice and phrase construction. Next to poor human judgment correlation, Ananthakrishnan et al. (2007) have identified further problems. First of all, a good translation can contain synonyms. However, if the translation modules do not share the same lexicon with the reference text, it receives a lower BLEU score. The BLEU method measures direct word-by-word similarity and looks to match and measure the extent to which word clusters in two documents are identical. A good translation that uses different words can therefore score badly if there is no match in the human reference.

Although BLEU allows for too little variation in vocabulary, it allows for too much variation in other aspects. This causes meaningless and syntactically incorrect variations to score the same as good variations. There are typically thousands of variations on a hypothesis translation, of which a vast majority of them are both semantically and syntactically incorrect, but which, nevertheless, receive the same BLEU score.

Keeping in mind the previously mentioned critiques, we use additional automatic evaluation metrics. A second metric is the US National Institute of Standards and Technology metric (NIST) (Doddington, 2002), which is an adaptation of the BLEU metric. Instead of only calculating n-gram precision with equal weights, NIST first computes the relevance of a specific n-gram. In this way, less frequent n-grams are given more weight as compared to more frequent ones. The NIST method also differs from BLEU in its calculation of the brevity penalty, as small variations in translation length have less impact on the overall score.

A final automatic metric we use is F-Measure (Melamed et al., 2003), which is defined as the harmonic mean of precision and recall: Since the system should not output errors, we compute the number of correct words generated by the system in terms of precision:

$$\text{precision} = \frac{\text{correct}}{\text{output} - \text{length correct}} \tag{4.3}$$

On the other hand, to ensure that nothing is overlooked , the number of words that a system should generate correctly is computed in terms of recall:

$$\text{recall} = \frac{\text{correct}}{\text{reference} - \text{length}} \tag{4.4}$$

F-measure, then, is the harmonic mean of the two metrics:

$$\text{f-measure} = \frac{\text{precision} \times \text{recall}}{(\text{precision} + \text{recall})/2} \tag{4.5}$$

For MT, this can be reformulated as:

$$\text{f-measure} = \frac{\text{correct}}{(\text{output-length} + \text{reference-length})/2} \tag{4.6}$$

4.3.2 Manual Evaluation

A first manual evaluation was performed with a qualitative goal. It was performed by identifying systematically occurring translation errors that are related to certain linguistic phenomena so they could be handled in future systems. For each of these errors, up to 100 segments in the source language were extracted.

Linguistic phenomena that were sensitive to erroneous translation in the QTLeap data were: imperatives, compounds, menu item separators, quotation marks, verbs, and terminology. Subsequently, the total occurrences of these linguistic phenomena were counted in both the source sentence and in the outputs of the MT systems. For each of the linguistic phenomena, 600 English source segments were extracted from the QTLeap corpus, a corpus that was developed within the QTLeap project containing questions and answers regarding the IT domain. In those source segments, 2,015 instances of the different phenomena were found overall, as it was often the case that more than one instance occurred per segment. The accuracy of the MT outputs was then measured by dividing the overall number of correctly translated instances of the phenomena by the overall number of instances in the source segments.

4.3.3 Task-based Evaluation in a Real Usage Scenario

Task-based or extrinsic evaluation can be used in real usage scenarios where the text is used by humans to make decisions or perform actions. In a final evaluation, we use such an evaluation method and tested the suitability of the MT systems in a real usage scenario, namely in an IT help desk QA system provided by Higher Functions – Intelligent Information Systems Limited (HF), a Portuguese small to

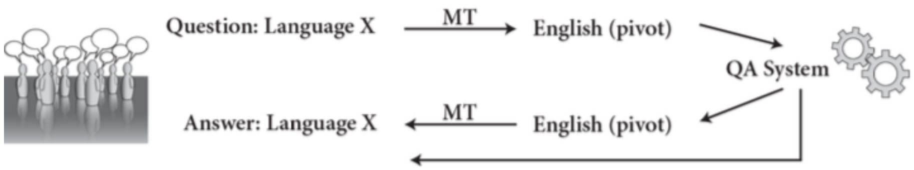

Figure 4.1 | *IT help desk workflow with MT services*

medium enterprise. This company aims to help users to solve problems with their IT devices. They can ask a question through a chat channel and receive an answer immediately.

Applying an MT system to these questions and answers extends this support service because users can ask a question in their own language. Their question is translated into English, the language in which questions and answers are stored in the database. The retrieved answers can, in turn, be translated back into the user's language. The automatic process of translating the questions and the corresponding answers could help minimize human operations, which only become necessary if there is no similar question-answer pair already available in the database. The received answer may not be perfect, but its quality should be sufficient to resolve the question.

The evaluation in the real usage scenario was composed of two distinct parts. Both parts were carried out using an online platform designed by HF for this purpose, namely a chat-based PC help desk scenario. Figure 4.1 shows the help desk with the embedded MT services. The translation direction X→EN was aimed at supporting information retrieval from the database whose question and answer pairs were recorded in the pivot language, that is, English. The first part of the evaluation focused on how the translation of the question into the pivot language affected the answer retrieval component of the QA system (Section 4.3.3). The second part focused on the outbound translation, aiming to evaluate to what extent the translated answer was clear and understandable to the final customers (Section 4.3.3).

Evaluation of the Answer Retrieval Step

The first part of the task-based evaluation focused on how the inbound translation affected the answer retrieval component of the QA system. The main idea of this evaluation step was to compare the effects when an English question was used by the QA system, with the result obtained when the same question was translated into English from a different language by the MT service.

The effects of the retrieval component were measured by means of the confidence scores of the answers. A user posted a question to the QA system, where it was matched against question-answer pairs that were previously generated by humans. Depending on the matching result of these new and old question-answer pairs, further steps were provided. An answer was displayed to the client without human intervention only if it had a confidence score above 95 points out of 100.

Ideally, the answer satisfied the end users needs and helped to solve the problem or to provide the requested information. If no answers reached this score, the top five results with a confidence score above 75 were shown to an operator, who could choose to adopt one of them, or accept none of them and provide a new answer. Eventually, the user and operator would further interact via chat. If no answer scored above 75 points, the question would be answered by a human operator with no help from the system. This would be the worst case scenario as the end user would need to contact the operator, which would consume a considerable amount of time and money on one single case.

Evaluation of the Outbound Translation Step

The second part of the task-based evaluation focuses on outbound translation, aiming to evaluate to what extent it delivers a clear and understandable answer to final customers without the intervention of a human operator.

The second part of the task-based evaluation focused on outbound translation, aiming to evaluate to what extent it delivered a clear and understandable answer to final customers without the intervention of a human operator. For this, we recruited testing subjects who matched as closely as possible the profile of typical HF users: non-experts in computer-related topics of mixed age groups. In the helpdesk web interface, they were presented with the reference answer and the translations of the systems (anonymized and put in a random order, so that the evaluation was blind) and were asked to assess which translation would be most helpful.

4.4 Statistical Machine Translation Baseline

In order to evaluate the progress of the project, a reference point was necessary that could represent the state of the art in MT. Therefore, a statistical machine translation (see Section 2.3.2) baseline was developed by training a phrase-based SMT system for each translation direction. The following account discusses the data and configurations used for these baseline systems for Dutch-English.

4.4.1 Moses

The SMT baseline systems were created using the Moses software (Koehn et al., 2007), a leading PB-SMT toolkit. It is a state-of-the-art open system that makes it easy to compare different systems. The software is a realization of a statistical method to MT providing tools to train and tune SMT systems as well as a decoder to run them. In addition to phrase-based SMT, it supports the creation of tree-based and factored models.

In Moses, creating a translation system from the training data contains various stages using different packages. In order to word align the parallel training corpus, Moses is commonly used together with GIZA++ (Och and Ney, 2000), which implements the IBM Models 1-5 and HMM models discussed in Section 2.3.2 with several extensions, improvements, and optimizations. The two main components in Moses are the training component and the decoder, which are described below.

Training

In Moses, the training pipeline consists of a collection of various tools, which take the raw data, both parallel and monolingual. Word alignments are used to extract phrase to phrase translations and corpus-wide statistics. It takes the parallel data as input and uses co-occurrences of words and phrases to deduce translation correspondences and estimate probabilities to create phrase tables containing statistical descriptions of the corpora. During the training process, the phrase table functions as a dictionary between the source and target languages. The final training step is tuning, where the different statistical models are weighted against each other to produce the best possible translations. Here, the optimal weights are found for the statistical by maximizing translation performance on a small subset of a parallel corpus.

Decoding

The second component in Moses is the decoder which, given a trained machine translation model and a source sentence, translates the sentence into a target sentence. The decoder in Moses uses the translation model that was created during training to search for the highest scoring sentences in the target language corresponding to the source sentence.

An important part of the decoder is the language model, a statistical account of one language that includes the frequencies of token-based n-gram occurrences in a corpus. The language model is trained on a large monolingual corpus or the monolingual part of a parallel corpus of the target side. In decoding, Moses uses the language model to select the most probable target language sentence from a large set of possible translations. These translations are generated by use of the phrase table and a reordering table that is used to reorder the output sentence.

4.5 Statistical Machine Translation Baseline System Settings

For the SMT baselines both training and testing data are preprocessed by tokenization. Sentences longer than 80 words and empty sentences are removed as they can cause problems with the training pipeline. Words are aligned with GIZA++ (Och and Ney, 2000). Tuning was undertaken on an excluded sample of 1% of the training data with minimum error rate training (MERT) (Och, 2003). The heuristics for the baselines were set to "grow-diag-final-and" alignment and "msd-bidirectional-fe" reordering. For the creation of the language models, IRSTLM (Federico and Cettolo, 2007) was used to train a 5-gram language model with Kneser-Ney smoothing(Kneser and Ney, 1995) on the target side of the training corpora.

4.6 TectoMT Settings

The data used for development and evaluation were domain-specific, namely the IT-domain. Therefore, a significant improvement could be gained in the transfer phase by using in-domain data. The transfer components from and to Dutch, included the combination of general-domain discriminative and dictionary translation models with in-domain models (Rosa et al., 2015).

For both English to Dutch and Dutch to English, discriminative (context-sensitive) and dictionary translation models (see Section 3.2.2 and Section 3.4.2) were trained. For training, a combination of the DPC corpus (Macken et al., 2007) and the KDE corpus (Tiedemann, 2012) were used (see Section 4.2 for more information on Dutch datasets).

4.7 Chimera Systems

In addition to the TectMT systems for Dutch and English, a hybrid MT system, was developed, namely Chimera. (Bojar et al., 2013; Bojar and Tamchyna, 2015)[3] This is one of several approaches combining a linguistically motivated MT system (TectoMT, Section 3.2) with a phrase-based SMT system (Moses, Section 4.4.1), trying to improve

[3] The creation of the Chimera system was undertaken by other project members and has been included here for completeness.

upon both of them (see Rosa et al. (2016) for an overview of related approaches).

The basis of Chimera is a Moses PB-SMT system that was trained with two phrase tables (and two sets of parameters for MERT training) instead of one. For this, TectoMT was used to provide additional training data for Moses. First, sentences were translated by TectoMT to create a synthetic parallel corpus. From this corpus, a secondary phrase table was extracted, which could be used together with the primary phrase table, extracted from the training data, to train Moses. The resulting Moses system then translated the input. This setup enabled Moses to use parts of the TectoMT translations that it considered good, while still using a large phrase table as a basis. The use of this additional phrase model could have a positive effect, for example, in choosing the correct inflection of a word when the language model encounters unknown contexts, or in generating translations for words that are not in the Moses phrase table.

Chimera was successfully used for the WMT 2016 IT-domain translation task (Bojar et al., 2016a). This domain-adaptation improved the BLEU scores for all tested languages (Czech, Spanish, Dutch and Portuguese). Here, for the creation of the Chimera systems, the first phrase table was extracted from the (big, general-domain) parallel corpora, used for training the baseline SMT system, while the second phrase table was extracted from much smaller, but in-domain training data, namely the answer part of the QTLeap corpus with TectoMT translations. Except for the additional, in-domain phrase table, the Chimera setup (tokenization, true-casing, reordering models, language models etc.) is the same as for the baseline SMT system.

4.8 Results

From the QTLeap corpus (Section 4.2), two out of four data sets containing 1,000 interactions question-answer pairs were used for the development (Batch 1) and testing of the baseline systems (Batch 2). Those datasets were further used for development of the TectoMT systems. During the project, Batch 3 was used for testing the intermediate systems while the main test corpus for the evaluation of the final systems was QTLeap Batch 4.

It is important to note that, in the QTLeap project the most important translation direction, in terms of quality, was the translation from English to another language. The Dutch to English translation direction was aimed at supporting information retrieval from the QA database where the question and answer pairs were recorded in English while the English to Dutch direction was aimed at supporting outbound translation thus supporting the delivery of the answer retrieved in the user's language. Therefore, although systems for both translation directions were established, both development and evaluation were primarily focused on the English to Dutch translation direction. For the same reason, a Chimera system was only developed for this direction.

The TectoMT and Chimera systems were compared to the SMT baseline (Section 4.4) using both the automated metrics and manual evaluations. We first give results on an experiment where different parallel corpora are used for the creation of the SMT baseline. We discuss the results of the automatic evaluation of all MT systems in Section 4.8.2 and the results of the manual evaluations in Section 4.8.3. The results of the task-based evaluation in the real usage scenario can be found in Section 4.8.4.

EN→NL	EP	DPC	EP+DPC	KDE	DPC+KDE	All
Self	21.38	37.52	38.39	34.86	**38.51**	38.29
HFdev	12.96	18.94	17.03	19.49	**19.95**	19.06
Test					28.16	

Table 4.1 | *Results of the comparison of different training sets for the SMT baseline in terms of BLEU score for English to Dutch. The used corpora are: Europarl (EP), Dutch Parallel Corpus (DPC), KDE, PHP and EU constitutions. In the third, fifth and final column the corpora are combined.*

4.8.1 Performance of the SMT Baseline Systems

In this section, the use of different datasets for the training of the SMT baseline systems is described. The systems incorporated successful settings from other language pairs and development efforts focused on the data used. Before creating the final SMT baseline systems, we tested which corpus, or which combination of corpora, was most effective for the Dutch-English language pair.

We created several SMT systems for both translation directions. Table 4.1 and 4.2 show the BLEU scores for each of them. Each model was tested on 1% of the corpus (self) and on the HF-development data (Batch 2). As we were comparing systems that use the same setting but different datasets to each other, we only used BLEU scores for the SMT baseline evaluation. Models trained on corpora that yielded better scores in terms of BLEU scores were then combined for the creation of a new system to see if this would further increase the scores.

The Dutch Parallel Corpus in combination with KDE yielded the best result on the development data for both translation directions and was therefore used for the final SMT baseline system. The addition of the

NL→EN	EP	DPC	EP+DPC	KDE	DPC+KDE	All
Self	24.46	41.63	**43.87**	36.48	42.15	43.82
HFdev	18.07	25.09	24.56	22.60	**25.59**	25.45
Test					33.47	

Table 4.2 | *Results of the comparison of different training sets for the SMT baseline in terms of BLEU score for Dutch to English. The used corpora are: Europarl (EP), Dutch Parallel Corpus (DPC), KDE, PHP and EU constitutions. In the third, fifth and final column the corpora are combined.*

smaller corpora to the Europarl and Dutch Parallel Corpus combination did not improve the system in comparison to DPC.

The models that yielded best results on the development data for both translation directions were used to translate the test data (Batch 3 of the QTLeap corpus). The final results of this system are shown in the third row of each language pair in Table 4.1 and 4.1. In the following sections, the best scoring systems were used as SMT baselines and were compared to the MT systems that were created in QTLeap.

4.8.2 Results of the Automatic Evaluation

For the automatic evaluations, scores were computed using the BLEU and NIST scripts from the Moses toolkit (See Section 4.3.1) and the F-MEASURE script from (Popović, 2012). For BLEU and F-MEASURE, $p < 0.05$ significance has been assessed using bootstrap re-sampling (Koehn, 2004). For NIST, no significance test was carried out. The, best system is marked in bold if it is at least 0.1 better that the second-best system.

EN→NL	BLEU	F-measure	NIST
SMT baseline	25.42	31.00	6.7286
TectoMT	22.35	29.02	6.6271
Chimera	**26.65**	**32.12**	**6.9139**

Table 4.3 | *Performance of the SMT baseline, TectoMT and Chimera for Dutch–English translation of the Batch4a (EN→NLanswers) part of the QTLeap Corpus.*

Table 4.3 shows the results of the performance of the final MT systems in terms of BLEU score for and English to Dutch on the answer part of Batch 4. The table shows that for the translation from English to Dutch, he SMT baseline was outperformed according to the automatic measures. TectoMT performed worse in comparison with the SMT baseline in BLEU (25.42 vs. 22.35). The Chimera system (26.65) was the best one. This shows that even if TectoMT was more than three BLEU points worse than Moses, the scores improved when the systems were combined.

Table 4.4 shows the results of the performance of the final MT systems in terms of BLEU score for Dutch to English on the question part of Batch 4. Here, the TectoMT systems performed better compared to the SMT baseline for all metrics. Note that the Chimera system was only developed in the English to Dutch translation direction as this was the focus of the project.

NL→EN	BLEU	F-measure	NIST
SMT baseline	27.89	34.30	6.8487
TectoMT	**30.34**	**36.03**	**7.1891**

Table 4.4 | *Performance of the SMT baseline, TectoMT and Chimera for Dutch–English translation of the Batch4q (NL→ENquestions) part of the QTLeap Corpus.*

4.8.3 Results of the Manual Evaluation

Results of the first manual evaluation, that consisted of a qualitative evaluation (Section 4.3.2), can be found in Table 4.5. As can be seen in this table, while Chimera outperformed the SMT baseline in all categories of the phenomena listed here, TectoMT was only worse in the placement of quotation marks. Overall, both Chimera and TectoMT performed better in comparison with the SMT baseline.

When looking at the output, for example, TectoMT and Chimera performed better in translating imperatives, as can be seen in example A. Here, the SMT baseline chose to translate the imperative with an infinitive while the other systems translated it correctly. Example B, then, provides an illustration of a case where Chimera and TectoMT outperformed the SMT baseline in the translation of terminology. With the SMT baseline system, the term key was translated with the Dutch word "sleutel", which refers to a tool that can be used to open a door. Chimera and TectoMT, however, correctly translated this word with "toets", referring to a key on a keyboard.

	#	SMT baseline	TectoMT	Chimera
imperatives	80	74%	90%	94%
compounds	37	65%	73%	76%
">" separators	40	90%	100%	100%
quotation marks	86	88%	80%	97%
verbs	110	70%	87%	95%
terminology	164	80%	91%	95%
sum	517			
average		78%	88%	94%

Table 4.5 | *Translation accuracy on manually evaluated sentences in Dutch focusing on particular phenomena. Test sets consist of hand-picked source sentences of Batch 2 that include the respective phenomenon.*

(A)	Source:	In the Insert menu, select Table.
	SMT baseline:	In het menu Invoegen Tabel selecteren.
	TectoMT:	Selecteer Tabel in het Invoegen menu.
	Chimera:	In de Invoegen menu, selecteer Tabel.
	Reference:	In het menu Invoegen kiest u de optie "Tabel"
(B)	Source:	Try pressing the F11 key.
	SMT baseline:	Probeer de toets F11 sleutel.
	TectoMT:	Probeer de F11 toets in te drukken.
	Chimera:	Probeer pressing de F11 toets.
	Reference:	Probeer op F11 te drukken.

Score	EN	SMT baseline	TectoMT
≥95	59.3%	18.2%	19.3%
75–94	35.8%	22.3%	26.3%
≥75	**40.5%**	**37.5%**	**45.6%**
50–74	4.5%	45.8%	44.0%
25–49	0.4%	12.4%	10.3%

Table 4.6 | *Percentage of the answers delivered by the QA system and their scores for English-only questions (EN) and the Dutch to English MT systems*

4.8.4 Results of the Task-based Evaluation

This section describes the results of the task-based evaluation that was performed in the context of a real usage scenario (Section 4.3.3).

Results of the Answer Retrieval Step

As mentioned in Section 4.3.3, this first evaluation step aimed to compare the results of receiving an English question (the default language used in the automated answering) or a question translated into English from Dutch in the real usage scenario. In this scenario, the user of the QA system received a direct answer without consulting a human operator if the confidence score of the translation matching was above 95 points. If this score was not reached by any of the candidate answers, the operator saw the top five answers that gained a confidence score above 75. The operator could then choose to either accept one of them or to decline them all and provide a new answer. In case there was no answer that scored above the 75 points threshold, the human operator would answer directly without consulting the system.

Table 4.6 presents the percentage of questions the QA algorithm was able to find a candidate answer for, within a certain confidence score interval, using the SMT baseline or TectoMT for Dutch to English. When no translation was used, the English QA system could automatically answer a question without human intervention in 59% of the cases. In 36% cases, it provided help for operators supporting them in finding the right answer. Only in about 4.9% of the cases was the operator left without any help. When the translation services were used for Dutch, the performance of the QA system was lower although, in most cases, an answer could be found by the system.

Results of the Outbound Translation Step

For the second evaluation step, test subjects were recruited who matched the profile of typical users of the QA system as closely as possible: non-experts in computer-related topics of mixed age groups. They were presented with the reference answer and the translations of the MT systems and were asked to rank these three alternative answers against the reference answer.

Table 4.7 and Figure 4.2, a graphical representation of Table 4.7, show the results of the manual evaluation of the English to Dutch translation direction for the TectoMT and Chimera systems. In Table 4.7, row *g* shows whether the difference between the SMT baseline and either TectoMT or Chimera was significant according to McNemar's test (McNemar, 1947) at the 95% confidence level.

The SMT baseline was outperformed in terms of manual intrinsic evaluation by both TectoMT and Chimera. Figure 4.2 clearly shows that the subjects have a preference for TectoMT over the SMT baseline especially when ignoring ties. We can see that both MT systems are at least as good as the respective SMT baseline, in Figure 4.2, the yellow bar reaches over 50%, that is, the systems produced a better

	TectoMT	**Chimera**
a. SMT-B better	31%	15%
b. Better than SMT-B	39%	26%
c. Equally good	12%	27%
d. Equally bad	19%	32%
e. SMT-B Equally good (c+d)	31%	59%
f. Better ignoring ties	56%	64%
g. Significant-McNemar	no	yes

Table 4.7 | *Comparison between the SMT baseline (SMT-B) and TectoMT translations and Chimera results*

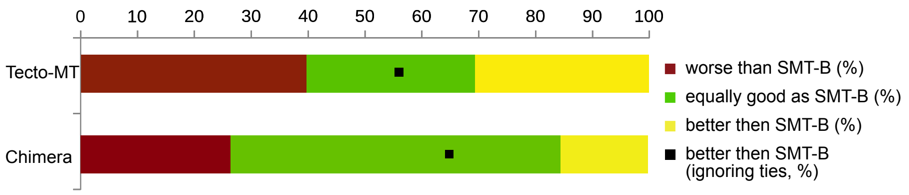

Figure 4.2 | *Comparison between the SMT baseline (SMT-B) and TectoMT translations and Chimera results*

or equally good translation for more than 50% of sentences). In the case of Chimera, the better ignoring ties score, which is plotted as a dark black box, is significantly higher than 50%.

4.9 Conclusions

In this chapter, the evaluation of the MT systems for the Dutch-English language pair that were created in the QTLeap project was described. The aim of the evaluation was to include more linguistic processing methods into the MT pipeline. Depending on the availability of training material, it may be difficult to improve on an SMT baseline by using a more complex architecture given the limitation in the available resources, such as parallel corpora. However, as the scores in Table 4.4 indicate, the TectoMT system did outperform the SMT baseline in the Dutch to English translation direction.

In addition to the TectoMT system, a further hybrid MT system was created for English to Dutch by the other project members. This Chimera system consists of a combination of TectoMT and Moses. In terms of BLEU score (Table 4.3), Chimera significantly improved upon both the SMT baseline and the TectoMT system, thus achieving the best results for English to Dutch. A quantitative manual evaluation provided some linguistic insights into the translation outputs. For most linguistic phenomena that have been found to be prone to error, both TectoMT and Chimera outperformed the SMT baseline.

A second manual evaluation, performed by volunteer subjects without thorough knowledge of the IT domain, focused on the impact of the translation on a helpdesk application. In general, the focus of this evaluation was to assess the added value of the translations in terms of their impact on the performance of the QA system of the helpdesk. We measured the percentage of questions where the QA algorithm was able to find a candidate answer within a certain confidence score interval. Better results were obtained with TectoMT translations than with the SMT baseline system.

The second part of the evaluation in a real usage scenario targeted the publication of the answer obtained by the QA system translated into the users language (Dutch) and was carried out by human evaluators. In this step, the English to Dutch translation direction was taken into account and the probability to call an operator was assessed. If we considered only the level where the probability of calling an operator was low, better results were obtained when TectoMT or Chimera was used.

In this chapter, we described the evaluation of MT systems that make use of both linguistic as well as statistical information. For the Dutch to English translation system, the baseline SMT system has been significantly, outperformed by TectoMT. In the opposite translation direction, Chimera outperforms both TectoMT and the SMT baseline. This demonstrates that improvements in terms of quality can be gained by combining more linguistically informed systems with statistical information.

We further explore the benefits of the combination of statistics and linguistics in the remainders of this thesis. In, Part II and III, additional linguistically and statistically motivated modules that can be implemented in the transfer-based MT framework are proposed.

PART II

From Words to Word Representations

CHAPTER 5

Embedding-Based Word Sense Disambiguation

5.1 Introduction

Automatically assigning the intended meaning of a word in its context, also known as Word Sense Disambiguation (WSD), has proven to be beneficial in several tasks such as parsing (Agirre et al., 2008), information retrieval (Agirre et al., 2009b) and cross-lingual information retrieval (Vossen et al., 2006). However, the usefulness of the task of disambiguating word senses truly becomes apparent in machine translation where one word can have multiple translations depending on its meaning. A correct translation of a word in a sentence may easily be wrong when it is used in a different context.

Consider, for instance, the Dutch word “vorst” in Example 5, which, in English, can be translated with “monarch”, “frost” and “ridge”. For speakers of Dutch, it is very easy to differentiate the meaning of this word in sentences 5a, 5b and even in the less common sentence, 5c.

(5) a. Het land wordt geregeerd door een strenge **vorst**.
*(The country is ruled by a strict **monarch**.)*

b. De landbouw wordt benadeeld door strenge **vorst**.
*(Agriculture is negatively affected by severe **frost**.)*

c. De vogel zit op de **vorst** van het dak.
*(The bird sits on the **ridge** of the roof.)*

Using our knowledge of the world, we understand that a monarch is someone who rules a country while frost is a weather condition that is disadvantageous for growing fruit or vegetables. While it is fairly simple for humans to identify the meaning of a word in a sentence, WSD is a difficult task for computers and thus requires a sophisticated approach.

Approaches to WSD may be supervised, when machine learning techniques are used to train a system from labeled training sets, or unsupervised, when the meaning of a word is inferred without access to such manually tagged corpora. Systems that are based on supervised learning methods gain best results (Snyder and Palmer, 2004; Pradhan et al., 2007; Navigli and Lapata, 2007; Navigli, 2009; Zhong and Ng, 2010). However, supervised learning requires a large amount of manually annotated data for training. Also, even if such a supervised system obtains good results in a certain domain, it is not readily portable to other domains (Escudero et al., 2000). Creating more and larger hand-tagged corpora for more different domains would be a solution to both problems. The process of creating such data sets, however, is very costly and time-consuming.

Sense	Gloss
vorst:noun:1	Hoofd van een land *(Head of a country)*
vorst:noun:2	Weer met temperatuur onder nul *(Weather-type with a temperature below zero)*
vorst:noun:3	Nok van een dak *(Ridge of a roof)*

Table 5.1 | *Glosses for the three senses of the Dutch word "vorst" (frost/monarch/ridge)*

5.2 Knowledge-based Word Sense Disambiguation

As an alternative to supervised systems, knowledge-based systems do not require manually tagged data and have proven to be applicable to new domains (Agirre et al., 2009a). To find the sense of a word in its context, they make use of information from lexical knowledge bases, such as dictionaries, thesauri and ontologies. An example of such a knowledge base is WordNet (Fellbaum, 1998), a semantic lexicon for a language. It can be seen as a dictionary containing definitions for each sense of a word and information on the various semantic relations between them, such as synonymy and antonymy.

A WordNet groups words in synsets, groups of words that are roughly synonymous in one of their meanings. Each synset also contains a *gloss*, which is a short description of the sense. Examples of glosses for the senses of the Dutch word "vorst" from example 5, can be found in table 5.1.

Two categories of knowledge-based algorithms are widely used: overlap- and graph-based methods. While graph-based methods make use of the structural properties of the graph that underlies a particular knowledge base, overlap-based methods exploit the overlap between the definition of a word and the words in its context. An example of such a system is the Lesk algorithm (Lesk, 1986). This method exploits the idea that the overlap between the definition of a word and the definitions of the words in its context can provide information about its meaning. It only requires two types of information: a set of dictionary entries with definitions (hereafter referred to as glosses) for each possible word meaning, and the context in which the word occurs.

The idea of the original algorithm is simple: given two words (w_i, w_j), it selects the sense whose definitions have the maximum overlap, the highest number of common words, in the definitions of the senses. For each sense s of the target word w_i the gloss $G(w_i)$ is taken. The context word w_j is represented in the same way as $G(w_j)$ For two words, the following score computes the overlap between each gloss of the ambiguous word and the glosses of its context words:

$$\text{Score}(w_i, w_j) = \text{overlap}(G(w_i), G(w_j)) \tag{5.1}$$

A popular variant of the algorithm is the simplified Lesk algorithm (Kilgarriff and Rosenzweig, 2000), which disambiguates one word at a time by comparing each of its glosses to the context in which the word is found. This variant avoids the combinatorial explosion of word sense combinations the original version suffers from when trying to disambiguate multiple words in a text. Given an ambiguous target word, the sense s whose gloss has the maximum overlap with its context $C(w)$ is selected. Given a target word w, the following score is computed for each of its senses S:

$$\text{Score}(s, w) = \text{overlap}(C(w), G(s)) \tag{5.2}$$

A problem with both versions of Lesk is that the lexical overlap can be very small as it is merely based on the gloss of the sense. Therefore, the original algorithm was adapted so that it exploits glosses of related meanings in WordNet, a version called extended Lesk (Banerjee and Pedersen, 2002). The glosses are extended with words from related senses (such as hypernyms and hyponyms). For each sense S of a target word w its score is estimated as:

$$\text{Score}(w_i, w_j) = \text{overlap}(C(w), G(s')) \tag{5.3}$$

Here $G(s)$ is, similar to C(w), the bag of words in the gloss of a sense s. It is, however, extended with the words of the glosses of related senses. A combination of the simplified and the extended methods was proposed by Ponzetto and Navigli (2010) into:

$$\text{Score}(s, w) = \text{overlap}(C(w) \mid G(s')) \tag{5.4}$$

Despite the extension of Banerjee and Pedersen (2002), the lexical overlap remains very small. In the next section, some distributional approaches that aim to overcome this problem are described.

5.2.1 Distributional Approaches

The aim of WSD systems that make use of distributional methods is to resolve word ambiguity without the use of annotated corpora. One example of representations of words is word embeddings (See Section 2.3.3). A disadvantage of word embeddings, however, is that they assign a single embedding to each word, thus ignoring the possibility that words may have more than one meaning. This problem can be addressed by associating each word with a number of sense specific embeddings. In supervised WSD, features based on such embeddings, learned using a combination of large corpora and a sense inventory, have been shown to achieve state-of-the-art results (Rothe and Schütze, 2015; Jauhar et al., 2015; Taghipour and Ng, 2015).

Several approaches to the creation of sense embeddings have been proposed in recent work. For example, in Reisinger and Mooney (2010) and Huang et al. (2012), a fixed number of senses has been learned for each word that has multiple meanings by first clustering the contexts of each token, and subsequently relabeling each word token with the clustered sense before learning embeddings. Rothe and Schütze (2015) have created additional embeddings for senses from WordNet on the basis of word embeddings. For this, an auto-encoder, called AutoExtend, was used that relies on the relations present in WordNet to learn embeddings for senses and lexemes. To create these embeddings, a neural network containing lexemes and sense layers was built, while the WordNet relations were used to create links between each layer. Their method takes any set of word embeddings and any lexical database as input and produces embeddings of senses and lexemes, without requiring further resources.

In the remainder of this chapter, a knowledge-based WSD method, inspired by Lesk, that combines a lexical knowledge base with distributed representations for words is proposed. In Section 5.3, the method and experiments undertaken to test this proposal are described. Then, in Section 5.4, our method is further extended with other approaches that aim to improve Lesk.

5.3 Lesk++

We proposed a knowledge-based WSD method that exploits word and sense embeddings. Our approach was loosely based on the simplified extended Lesk algorithm (Ponzetto and Navigli, 2010), that also uses a knowledge base to find the correct sense of a word in context. A problem with the aforementioned method, however, is that, when a gloss is matched against the context of a word, in most cases the lexical overlap is very small. In order to solve this problem, our WSD method computed the similarity between embeddings of the gloss of a sense and the embedding of the context of the word, instead of counting the number of words that overlap. Additionally, we used the similarity between the context and the lexeme, the word-sense combination. The sense that has the highest combined similarity was selected as the best sense.

Our system made use of a combination of sense embeddings, context embeddings, and gloss embeddings. Similar approaches have been proposed by Chen et al. (2014) and Pelevina et al. (2016). The main difference in our approach was that instead of automatically inducing sense embeddings and finding the best sense by comparing them to context embeddings, we used WordNet-based senses and added gloss embeddings for better performance. Also, such systems are not readily applicable for applications that rely on WordNet-based senses, such as machine translation and information retrieval and extraction systems (see Morato et al. (2004) for examples of such systems).

A similar setup has been proposed by Basile et al. (2014) who used a distributional approach to representing definitions and the context of the target word. Here, semantic vectors for glosses and contexts were used to compute similarity of the gloss and the context of a target word, while we also computed the similarity of a sense and its context directly using sense embeddings. Brody and Lapata (2008) have used distributional similarity to automatically annotate a corpus for training a supervised method. Each target word in the corpus was paired with a list of neighboring words, selected via distributional similarity. Those neighbors were linked to a sense in WordNet and were then used for annotation. Miller et al. (2012) have exploited a distributional thesaurus to expand glosses and context before using the simplified Lesk algorithm. Inkpen and Hirst (2003) have applied gloss and context vectors to the disambiguation of near-synonyms in dictionary entries.

5.3.1 Method

Our WSD algorithm takes sentences as input and outputs a preferred sense for each polysemous word. Given a sentence $w_1 \dots w_i$ of i words, we retrieve a set of word senses from the sense inventory for each word w. Then, for each sense s of each word w, we consider the similarity of its lexeme (the combination of a word and one of its senses (Rothe and Schütze, 2015)) with the context and the similarity of the gloss with the context. For this, our method requires lexeme embeddings $L_{s,w}$ for each sense s. We create such embeddings with AutoExtend (Rothe and Schütze, 2015) from WordNet on the basis of word embeddings.

For each word w we need a vector for the context C_w, and for each sense s of word w we need a gloss vector G_s. The context vector C_w is defined as the mean of all the content word representations in the sentence: if a word in the context has already been disambiguated, we use the corresponding sense embedding; otherwise, we use the word embedding. For each sense s, we take its gloss as provided in WordNet. In line with Banerjee and Pedersen (2002), we expand this gloss with the glosses of related meanings, excluding antonyms. Similar to the creation of the context vectors, the gloss vectors G_s are created by averaging the word embeddings of all the content words in the glosses.

For each potential sense s of word w, the cosine similarity is computed between its gloss vector G_s and its context vector C_w and between the context vector C_w and the lexeme vector $L_{s,w}$. The score of a given word w and sense s is thus defined as follows:

$$\text{Score}(s, w) = cos(G_s, C_w) + cos(L_{s,w}, C_w) \tag{5.5}$$

The sense with the highest score is chosen. When no gloss is found for a given sense, only the second part of the equation is used.

Prior to the disambiguation process, we sort the words by the number of senses it has, in order that the word with the fewest senses will be considered first. As our method relies on the words in the context, which may themselves be ambiguous, words that have fewer senses can be easier to disambiguate (Chen et al., 2014). If words in the context have been disambiguated already, this information can be used for the disambiguation of words that follow. We therefore use the resulting sense of each word for the disambiguation of the following words starting with the "easiest" words that have fewest senses.

5.3.2 Experiments

We tested our method on both Dutch and English. Our sense inventory for Dutch was Cornetto (Vossen et al., 2012b) and for English, we used WordNet 1.7.1 (Fellbaum, 1998) as this version matches our sense embeddings. In Cornetto, only 51.0% of the senses have glosses while, in the Princeton WordNet, almost all of them do.

For Dutch, we build 300-dimensional word embeddings on 500 million words from the SoNaR corpus (Oostdijk et al., 2013) using word2vec CBOW (Mikolov et al., 2013a), and created sense- and lexeme embeddings with AutoExtend. For English, we used the embeddings from Rothe and Schütze (2015).[1] They lie within the same vector space as the pre-trained word embeddings by Mikolov et al. (2013a)[2],

[1] http://www.cis.lmu.de/~sascha/AutoExtend/
[2] https://code.google.com/p/word2vec/

trained on part of the Google News dataset, which contains about 100 billion words. This model (similar to the Dutch model) contains 300-dimensional vectors for 3 million words and phrases.

We evaluated our method by comparing it with a random baseline and simplified extended Lesk (SE-Lesk) (Kilgarriff and Rosenzweig, 2000; Banerjee and Pedersen, 2002; Ponzetto and Navigli, 2010). Additionally, we compared our system to a state-of-the-art knowledge-based WSD system, UKB (Agirre and Soroa, 2009), that, similar to our method, does not require any manually tagged data. The UKB system can be used for graph-based WSD using a knowledge base. It applies random walks, for example, Personalized PageRank, on the Knowledge Base graph to rank the vertices according to the context. We used UKBs Personalized PageRank method word-by-word with WordNet 1.7 and eXtended WordNet for English, as this setup yielded the best results in Agirre and Soroa (2009).

We did not compare our system to the initial results of AutoExtend (Rothe and Schütze, 2015) as they tested it in a supervised setup using sense embeddings as features. However, as is customary in WSD evaluation, we compared our system to the most frequent sense baseline, which is notoriously dicult to improve upon due to the highly distorted distribution of word senses (Agirre and Edmonds, 2006). As this baseline relies on manually annotated data, which our system aims to avoid, we considered this baseline to be semi-supervised.

To find the most frequent sense of each target word, for English, we used Semcor (Fellbaum, 1998), a sense tagged corpus comprising 250,000 words of text (mostly taken from the Brown Corpus(Francis and Kucera, 1979)) in which all content words have been tagged manually with word senses. For Dutch, we use the DutchSemCor corpus (Vossen et al., 2013b) containing 1 million sense annotations, for approximately 3,000 lemmas, of which 250,000 have been made manually. The manually annotated part of the corpus is balanced per sense, which means that an equal number of examples for each sense is annotated. It is therefore not a reliable source for computing the most frequent sense. As an alternative, drawing on the example of Vossen et al. (2013a), we derived sense frequencies by using the automatically annotated counts in DutchSemCor. [3]

The most frequent sense baseline for Dutch was, therefore, lower, compared to the English, where the most frequent sense of a word was fully based on manual annotation.

Test data was acquired by taking a subset of 5,000 annotated sentences from both SemCor and DutchSemCor. For Senseval-2 (SE-2) and Senseval-3 we use the WSD evaluation framework of Raganato et al. (2017), which provides evaluation datasets and output of other knowledge-based WSD systems. From those systems we reported on the extended Lesk version (DSM) of Basile et al. (2014),[4] which is most similar to our approach.

5.3.3 Results

The results of the evaluation of our method (Lesk++) for both Dutch and English can be found in Table 5.2. Accuracy was calculated by dividing the number of words that were disambiguated correctly, as compared to the sense tagged corpus, by the total amount of polysemous words. Results are in bold when statistically significant over the other WSD systems at $p < 0.05$.

For both Dutch and English, our method performed significantly better than SE-Lesk and the random baseline for all tasks. Also,

[3] In DutchSemCor senses are annotated with an SVM, trained on the manually annotated part of the corpus. See Vossen et al. (2013a) for more details.

[4] We use `https://github.com/pippokill/lesk-wsd-dsm` without sense frequency for comparability.

	DSC	SC	SE-2	SE-3
Lesk++	**45.9%**	55.1%	54.9%	**59.3%**
SE-Lesk	28.1%	53.2	52.1%	50.1%
UKB	38.9%	**57.6%**	**56.0%**	51.8%
DSM	-	-	51.2%	42.3%
Random	26.5%	33.6%	39.9%	34.9%
MFS	*36.0%*	*70.9%*	*65.6%*	*66.2%*

Table 5.2 | *Performance in terms of accuracy of Lesk++ compared to the baselines on DutchSemCor (DSC), SemCor (SC) Senseval-2 (SE-2) and Senseval3 (SE-3)*

our system performed better than UKB on both DutchSemCor and Senseval-2. On DutchSemCor, it outperformed the most frequent baseline.

Effects of Each Module

The main idea behind our method was a simple combination of two cosine similarity scores. In a second experiment, we evaluated the effects of both of these scores by using them separately. Additionally, we examined the use of sorting the words by its number of senses before disambiguation.

We compared our final results with a system where similarity is only computed between the context and gloss vector and with a system that only computes the cosine distance between the context and the lexeme (only the first and the second part of Equation 5.5 respectively). Both systems were tested without and with sorting the target word sequence. The results of this third experiment on the sense tagged corpora for Dutch (DSC) and English (SC) can be found in Table 5.3.

	Lesk++	Lexemes	+Sorting	Gloss	+Sorting
DSC	50.3%	42.7%	44.7%	45.9%	46.4%
SC	55.1%	44.7%	45.2%	47.0%	52.7%

Table 5.3 | *Effects of lexemes, glosses and sorting. The second and the fourth column show results of a system that only uses the lexeme (Lexemes) or gloss vectors (Gloss) respectively. In the third and last column sorting (+Sorting) is added.*

For Dutch, the results indicated that sorting the word by its number of senses by itself is not very effective compared to the system that does not use this module. The use of glosses, on the other hand, seemed to be very effective, while the combination of both measures yielded the best results. The effect of the gloss vectors was even stronger for English, which can be explained by the fact that the English WordNet has a higher gloss coverage. Also, for English, although both sorting and glosses were effective, the combination performed better.

Comparison of Different Domains

To examine whether our results were domain specific, we evaluated our system for Dutch on four parts of DutchSemCor. The results of this experiment for the all-words task can be found in Table 5.4. On every subsection of the DSC dataset, our method outperformed SE-Lesk, the random baseline and UKB. Furthermore, our method outperformed the most frequent baseline on three subsections. The newspapers subsection formed an exception, probably because it belongs to a more general domain, which is known to be more difficult for knowledge-based WSD (Agirre et al., 2014).

	dl	st	wp	np
Lesk++	**35.1%**	**37.1%**	**45.2%**	41.4%
SE-Lesk	27.7%	30.4%	28.8%	29.4%
UKB	30.5%	32.1%	37.3%	33.8%
Random	24.2%	23.5%	28.2%	25.7%
MFS	30.6%	33.3%	35.8%	**42.9%**

Table 5.4 | *Results for the Dutch all-words task on a random subset of each of the four largest datasets from DutchSemCor: discussion lists (dl), subtitles (st), Wikipedia (wp) and newspapers (ns).*

5.3.4 Intermediate Conclusions

The difference in results for Dutch and English could be explained by the difference in the datasets. The Cornetto coverage compared to Princeton WordNet was only about 60% and the average polysemy was 1.07 for nouns, 1.56 for verbs and 1.05 for adjectives, while for English it was 1.24 for nouns, 2.17 for verbs and 1.40 for adjectives.

Furthermore, in Cornetto, less than half of the synsets have glosses, while in the English WordNet almost all of them do. The lack of glosses had a large impact on the Lesk algorithm since it is very sensitive to their exact wording, so the absence of a certain word can radically change the results. The algorithm determines overlap only among the glosses of the senses being considered. Since dictionary glosses can be absent or very short, they cannot always provide enough words to distinguish between two senses. In our method, if no gloss was available the system could fall back on the similarity of the sense and the context. Yet, the results of our experiments demonstrated that the performance of our algorithm also suffered somewhat from this problem. Our model achieved better performance compared to both the simplified extended Lesk algorithm and a random baseline for Dutch

As both the Lesk algorithm and our extension rely on the definition of the words and the words that surround it, it would be interesting to see whether adapting both sources of information would improve either of them. Taking these results into account, there are two possibilities: expansion or reduction. For the first option, the existing words of the context and glosses could be expanded with additional words that have similar meanings. For example, Miller et al. (2012) have used a distributional thesaurus, that is computed from a large parsed corpus to lexically expand the context and glosses with related words. They have shown that, using such expanded context and glosses, improves the simplified extended version of Lesk.

When reducing the amount of words in either the context or the target word's sense, methods are required that prevent the loss of informative words. Vasilescu et al. (2004) have shown that a preselection of words in the context of the target word improves simplified Lesk. In the next section, we describe experiments where both methods are used in combination with ours.

5.4 Lexical Expansion and Lexical Selection

We used the method of Miller et al. (2012) to expand the glosses and the contexts of the target words before using our adaptation of the Lesk system.[5] For each content word we retrieved the 30 most similar terms from the distributional thesaurus and added them to the context or gloss while occurrences of the target word were removed.

[5] We use the distributional thesaurus downloaded from: `www.lt.informatik.tu-darmstadt.de/de/data/distributional-thesauri`.

For each content word we retrieve the 30 most similar terms from the distributional thesaurus and add them to the context or gloss while occurrences of the target word are removed.

For the selection of context words, we used the lexical chaining technique, as applied in Vasilescu et al. (2004)), that uses the idea of creating lexical chains from Hirst and St-Onge (1998). Lexical chains are sequences of words that are semantically related. Drawing on Vasilescu et al. (2004), we used the synonymy and hypernymy relations in WordNet in combination with a similarity measure (Jaccard formula (Jaccard, 1901)), to verify whether a context word was a member of such a lexical chain. For both the target word *w* and each context word *c* in its context, we retrieved a set of sense definitions of all the synonyms and hypernyms of *w* according to the WordNet hierarchy. A context word was added to the context if the similarity score for the set of *w* and the set of *c* was greater than an experimental threshold.

5.4.1 Experiments

The experimental setup was the same as the one described in Section 5.3.2. However, on this occasion, as well as UKB, we also reported on the extended Lesk version of Basile et al. (2014), (DSM) without sense frequency for comparability which is most similar to our approach.

	DSC	SC	SE-2	SE-3
SE-Lesk	28.1%	53.2	52.1%	50.1%
+LE	**29.6%**	**56.5%**	**51.0%**	**49.3%**
+LS	16.0%	40.7%	48.1%	54.3%
+LE,LS	25.2%	40.6%	46.2%	46.0%
Lesk++	45.9%	55.1%	54.9%	59.3%
+LE	42.5%	47.8%	43.8%	46.2%
+LS	**47.3%**	**67.2%**	**58.4%**	**59.4%**
+LE,LS	41.0%	66.9%	49.1%	43.5%

Table 5.5 | *Results for DutchSemCor (DSC), SemCor (SC), Senseval-2 (SE-2) and Senseval3 (SE-3) for simplified extended Lesk (SE-Lesk) and Lesk++. The following columns use lexical selection (LS), lexical extension (LE) and both extension and selection (LE,LS).*

5.4.2 Results

Table 5.5 shows the results of both SE-Lesk and our method (Lesk++) with lexically extended (LE) and selected (LS) context and gloss vectors. The use of word and sense embeddings yielded overall better results as compared to SE-Lesk. Remarkably, lexical extension, which was very beneficial for SE-Lesk, caused serious harm to our method. Selecting words in the context, on the other hand, improved our method and worsened the performance of SE-Lesk.

Table 5.6 shows the results of the best performing combinations, SE-Lesk with lexical extension and Lesk++ with lexical selection, compared to three baselines. Our system, when used in combination with the lexical selection method, performed better than the other purely knowledge-based methods. The different performance of the extensions to SE-Lesk and Lesk++ demonstrated that both algorithms capture different types of information and therefore require a different

	DSC	SC	SE-2	SE-3
Lesk++LS	**47.3%**	**67.2%**	**58.4%**	**59.4%**
SE-Lesk,LE	29.6%	56.5%	51.0%	49.3%
UKB	38.9%	57.6%	56.0%	51.8%
DSM	-	-	51.2%	42.3%
Random	26.5%	33.6%	39.9%	34.9%
MFS	*36.0%*	*70.9%*	*65.6%*	*66.2%*

Table 5.6 | *Results for simplified extended Lesk (SE-Lesk) with lexical extension (LE) and Lesk++ with lexical selection (LS), UKB, DSM, a random and a most frequent sense baseline*

type of input. Since SE-Lesk counts on the direct overlap of words, it is highly dependent on a larger amount number of words. Lesk++ on the other hand, overcame this problem and clearly benefits from more "quality" information in the contexts.

5.5 Conclusions

Our model achieved better performance compared to both the simple extended Lesk algorithm and a random baseline for Dutch and English. The evaluation, where our method is compared to a random baseline, a first sense baseline and a Lesk algorithm, demonstrated that it outperformed both the Lesk algorithm and the random baseline for Dutch and English. Furthermore, it gave promising results for some datasets for outperforming the most frequent sense baseline. A second experiment confirmed the effects of gloss vectors, while the results of a final experiment indicated that our method works well in different domains.

Although our method worked well on its own, its simplicity allowed us to explore whether other extensions to the Lesk algorithm that have proven to be successful could improve it further. We therefore compared several extensions to the Lesk algorithm with an adaptation that uses sense, gloss and context embeddings to compute the similarity of word senses to the context in which the words occur. When using a selection scheme before creating context vectors, its performance was better than our knowledge-based baselines.

The main advantage of our method is its simplicity which makes it fast and easy to apply to other languages if a knowledge base is available. Moreover, it only requires unlabeled text and the definitions of senses, and does not rely on any manually annotated data, which makes our system an attractive alternative for supervised WSD. In the next chapter, we use our WSD model for the location and interpretation of puns to further test its performance.

CHAPTER 6

Word Sense Disambiguation for Pun Location and Interpretation

The WSD system used in this chapter is the one that was created by the author of this thesis, as described in the previous chapter. Adjustments to the system to make it fit for the pun location and interpretation tasks, the development of the development data and the evaluation of the results was shared work with Kilian Evang.

6.1 Introduction

In this chapter we apply our WSD system in a somewhat different setting. We use our system to participate in the SemEval-2017 Task 7,[1] for the subtasks of homographic pun location and homographic pun interpretation. A pun is a word used in a context to evoke two or more distinct senses for humorous effect. For example, in the 1987

[1] `http://alt.qcri.org/semeval2017/task7/`

movie *The Running Man,* Arnold Schwarzenegger's character cuts his enemy Buzzsaw in half with a chainsaw, then announces: "He had to split." The verb *split* is the pun here, evoking two senses in the context: that of leaving, and that of disintegrating into two parts.

Recognizing and appreciating puns requires sophisticated feats of intelligence currently unique to humans. In a recently proposed set of artificial intelligence tasks Miller et al. (2017a) have challenged computers to try their hand at it: *pun detection* (tell whether or not a text contains a pun), *pun location* (given a text with a pun, tell which word is the pun) and *pun interpretation* (given a pun in context, tell which senses it evokes).

Pun interpretation is closely related to the task of Word Sense Disambiguation. A typical WSD system chooses that sense of a word which fits best in the context the word appears in. A pun interpretation system, however, should return not one but two different senses of a word. Miller and Turkovi (2016) suggest a straightforward extension of the WSD approach to pun interpretation: choose the best and second-best scoring sense for the word in its context. However, this approach does not take into account the specific structure of pun-based jokes. In most cases, such jokes can be divided into two parts, where, in the first part, cues for one sense are concentrated, and in the second part, cues for another sense appear. Figure 6.1 shows examples of such cases.

A pun interpretation system could exploit this two-part structure by splitting the global context of the entire joke into two local contexts and performing WSD separately for each local context, choosing the best sense for each of the two. As this process makes each context more informative for the respective sense, we hypothesized that it leads to more accurate pun interpretation than the simple approach which uses the top-scoring two senses according to the global context.

The first time he put the horses on the carriage, it went without a **hitch**.
If a priest is called a white collar worker, a nun would be a creature of **habit**.
Old math teachers never die, they just become **irrational**.
In the winter my dog wears his coat, but in the summer he wears his coat and **pants**.

Figure 6.1 | *Examples of pun-based jokes. The pun is typeset in boldface. Words that we judge to be cues for one sense are marked with a dashed underline, words and n-grams that we judge to be cues for the other sense are marked with a dotted underline. Note that the cues for the two senses tend to divide the jokes into two non-overlapping parts.*

Additionally, we believe that we can use the output of our pun interpretation system for pun location. We hypothesize that the two senses of a pun are typically very dissimilar, as this is important for the joke to be recognizable. We therefore attempted to locate puns by selecting the polysemous word with the most dissimilar two senses.

6.2 Method

Drawing on Miller and Gurevych (2015) we used a knowledge-based WSD system and applied it to pun annotation. For this, we used our word and sense-based WSD system as described in Section 5.3.1.

6.2.1 Pun Interpretation: The Context Had to Split

Our WSD method returned the sense with the highest score taking into account the whole context. For pun interpretation, we could simply adapt it to return the best and second-best sense. However, to exploit the two-part structure of pun-based jokes, we instead split the context into two local contexts, run WSD for each local context and then returned the best sense according to each of the two.

Ideally, we wanted to split the context so that all cues for one sense are in one local context, and all cues for the other sense are in the other. Consider, for example, the pun "channel" in example 6.2.1 for which we wanted to find two senses. In order to find the sense of the television channel, the context should look like context 1. On the other hand, we also aimed to find the sense of the part of the Atlantic Ocean that divides the UK from the mainland of Europe, therefore, the second context should look more like the one in context 2.

(6) Television sets in Britain have to cross the English **Channel**.

full context: {Television sets, Britain, cross, English}

context 1: {Television, sets}

context 2: {Britain, cross, English}

We, therefore, split the context so as to maximize the semantic dissimilarity between both parts. For each possible split of the text into two contiguous parts, we created a vector for each part by taking the mean of all content words, as described earlier, and computed the cosine distance between both vectors. The pair of parts with the highest distance were used as local contexts for the WSD system. For each polysemous word in the sentence, the WSD system was applied twice, using a different part of the context. The highest scoring sense for each run was chosen. As both runs could assign the same sense to the word, the second-best sense of the first run was chosen in this case.

6.2.2 Pun Location: Attracting Opposites

We, therefore, split the context so as to maximize the semantic dissimilarity between both parts. For each possible split of the text into two contiguous parts, we created a vector for each part by taking the mean of all content words, as described earlier, and computed the cosine distance between both vectors. The pair of parts with the highest distance were used as local contexts for the WSD system. For each polysemous word in the sentence, the WSD system was applied twice, using a different part of the context. The highest scoring sense for each run was chosen. As both runs could assign the same sense to the word, the second-best sense of the first run was chosen in this case.

6.3 Experiments

For WSD, we used the same settings and datasets as described in section 5.3.2. Although a pun can have two or more different part of speech tags, our method did not account for this. Instead, we assumed that both potential senses of a pun have the part-of-speech that was assigned to the pun by the Stanford part-of-speech tagger (Toutanova et al., 2003).

6.3.1 Development Data

For the development of our system, we gathered and annotated a small dataset of 91 puns from the website "Pun of the Day". [2] We used instances that have the same characteristics as the data we considered for the subtasks (one pun per text, one content word per pun,

[2] http://www.punoftheday.com/

target exists in WordNet 3.1, pun is homographic). From a small set of downloaded texts, we first independently selected the texts that met all of these criteria. This was followed by a round of adjudication by discussion to determine the texts to use. We then used a similar process to annotate each pun with its two senses.

6.3.2 Pun Interpretation

We compared our pun interpretation method to three baselines: a random baseline, a most frequent sense baseline and a WSD system that does not use context splitting. The latter was modified to return the two senses with the highest score instead of one.

In addition, we compared different ways of splitting the context of the pun. Next to splitting on the basis of the maximal cosine distance between two possible parts of the context we also ran the WSD system with contexts that were split in half and with contexts that were split at the first punctuation symbol.

6.3.3 Pun Location

For pun location, we compared our system's performance to two baselines. One baseline randomly selects one content word from the text as the pun, and the other baseline always selects the *last* content word in the text as the pun. In addition, we used the output of all experimental setups of pun interpretation to assess the influence of the quality of assigned senses on pun location.

6.4 Results

Results of the experiments for pun interpretation can be found in Table 6.1. [3] Our system easily outperformed both the random baseline and the most frequent sense baseline. Also, if we split the context before disambiguating the target word, we gained higher scores as compared to a system that selects the two best scoring senses. We did not, however, gain higher scores when we split the contexts on the basis of maximum semantic dissimilarity. Instead we observed that a system that splits the context in half performs better.

The task of pun interpretation appears to be an extremely challenging problem This is confirmed by the random and most frequent sense baseline that are exceptionally low as compared to traditional word sense disambiguation. Still, in the Semeval competition, our system for interpreting puns ranked close second Miller et al. (2017b). Although most of the other systems also make use of some measure of lexical overlap, our system was the only one that made use of context splitting.

Table 6.2 shows results for pun location using the output of our system for pun interpretation. Our system scored well above our random baseline. However, the baseline selecting the last content word was much stronger, as the pun often appeared at the end of the joke in the data.

Compared to the other systems in the Semeval task, our system ranked last out of ten Miller et al. (2017b). The last word baseline turned out to be surprisingly hard to beat for this subtask This, was

[3] Lesk++ optimal split, was the submitted system. Numbers differ slightly with the reported numbers of Semeval due to a fixed inconsistency in how words are handled for which only one sense could be found.

System	Coverage	Precision
Random baseline	98.92	7.24
MFS baseline	98.92	11.21
Lesk++, no splitting	98.23	15.45
Lesk++, split in half	98.23	16.39
Lesk++, split by punctuation	98.23	15.53
Lesk++, optimal split	98.23	15.53

Table 6.1 | *Results for pun interpretation on the shared task test data.*

	Coverage	Precision
Random content word	100.00	13.20
Last content word	100.00	52.96
Lesk++, optimal split	100.00	27.69

Table 6.2 | *Results for pun location on the shared task test data.*

also the case for the other participants in the shared task Miller et al. (2017b).

Results of the experiments for pun location using the output of our system compared to all baselines for pun interpretation and different splitting setups are shown in Table 6.3. Using the output of our system, with or without context splitting, performed better compared to systems that use random or most frequent output. There was little difference in pun location in the output of systems that use splitting modules and the ones that do not.

Pun interpretation system	Coverage	Precision
Random baseline	100.00	17.24
MFS baseline	100.00	18.48
Lesk++, no splitting	100.00	27.75
Lesk++, split in half	100.00	27.75
Lesk++, split by punctuation	100.00	28.44
Lesk++, optimal split	100.00	27.69

Table 6.3 | *Results for pun location on the shared task test data with different splitting setups*

6.5 Discussion

Our method for pun interpretation did not yet deal with puns where each sense has a different part of speech. A solution to this would be to also use the senses of the words second-best option of a part-of-speech tagger. Our method also did not deal with phrasal verbs and multi-word expressions.

Our method for pun location worked better than chance, but much worse than a simple heuristic exploiting of the fact that puns typically appear at the end in the data. It would be interesting to see if both methods can be combined, for example, by using confidence scores and the heuristic as a fallback. It would also be interesting to see if the heuristic can be applied to other types of data, such as movie scripts.

6.6 Conclusions

We hypothesized that the idea that pun-based jokes can be divided into two parts, each containing information about the two distinct senses of the pun, can be exploited for pun interpretation. Experiments were undertaken splitting the context that was input into a WSD system into two parts, running WSD for each context and returning the best sense for pun interpretation. Results of our experiments demonstrated that on the pun interpretation task, systems that use such a module outperform a WSD system that returns the two best senses. Also, our system performed better compared to both the random and the most frequent baseline.

As we expected the two meanings of a pun to be very dissimilar, we used the output of pun interpretation for pun location. Computing cosine distances between each sense-pair and selecting the one that has the highest distance gained higher scores as compared to a system that randomly selects a content word to be the pun.

PART III

From Word Representations to Words

CHAPTER 7

Lexical Choice

7.1 Introduction

In the first part of this thesis, we described the creation and evaluation of a transfer-based machine translation system. Such systems transform a linguistic analysis of a source sentence into a linguistic representation that serves as a basis for the construction of target sentences. In this setup, a generation system is used to construct sentences on the basis of these representations.

An important part of the generation process is to find words in the target language that can correctly represent the meaning of the original sentence in the source language. One way to preserve word meaning is to map to, and generate from, representations of word meaning. Previously, in Part II, we described the task of word sense disambiguation in which, for a given word and its context a meaning representation in the form of a WordNet synset was found. In this chapter, we describe the opposite task, namely lexical choice. Here, we select a target word for a given Wordnet synset context.

The process began with a meaning representation for which an appropriate word needed to be selected that could both express the meaning of its source word and fit its target context. We therefore considered dependency structures that, instead of words, contain word senses in its leaf nodes. For the representation of word senses, we used the output of a WSD system, namely WordNet synsets. As a synset consists of a set of lemmas that are synonymous, they are well-suited for lexical choice. From the lemmas in this synsets, it was the task here to select a target lemma that fit the target context best.

A potential problem of the use of synsets as input for generation, however, is that not every lemma in a synset is a full synonym of its original word. It can, therefore, be problematic to select the most probable one without considering the context. Consider for instance the English synset: $\{''employment'',''work''\}$. Both lemmas in this synset have the meaning of "The occupation for which you are paid".

(7) a. He is looking for **employment**
He is looking for **work**

b. He is out of **employment***
He would like to terminate his **work*** agreement

In the sentences of Example 7a they are perfectly exchangeable while, in Example 7b, both sentences require a different lemma in this particular context.

In this chapter, we first discuss the task of natural language generation (NLG) from abstract representations, of which lexical choice is a subtask (Section 7.2). The second part of this chapter (Section 7.3), describes preliminary experiments on lexical choice in abstract dependency trees with word senses. We challenge the Alpino generator to prepare it for MT by choosing lemmas from word senses. When using some basic heuristics to choose lemmas, we are able to locate existing problems in the generator before using it in a multilingual setting. Furthermore, we can find out what information is necessary to create a more sophisticated model for lexical choice.

7.2 Natural Language Generation

Lexical choice is a part of the broader task of NLG. This task is responsible for the construction of natural language sentences from meaning representations. As Reiter and Dale (2000) have observed, the generation of sentences is different from other tasks in NLP in the sense that an enormous amount of complex choices need to be made. For example, when mapping representations of language to text, a large amount of one-to-many selections are to be resolved. Therefore, flexible systems are required that make use of linguistic features containing distinctions within the language. Grammar-based systems, or syntactic generation systems (van Noord, 1990), are a subcategory of such approaches that employ linguistic constraints where the relations between meaning and form are encoded in an externally specified grammar.

The process of syntactic generation can be divided into two subtasks. First, it should be decided "what to say", a task that is known as strategic generation. Here, the intended meaning of an utterance is determined and encoded in a meaning representation. In the second sub-task, then, it is decided "how to say it". Sentences are generated on the basis of the meaning representations that are the result of the previous task. In this thesis, as we consider MT systems that translate on the basis of an abstract analysis of a sentence, we assume that the meaning representations that are the input to our generation system are the output of the analysis phase of an MT system. We, therefore, expect the strategic part to be covered and focus on the task of tactical generation.

The Alpino generator, which we used for our experiments, uses the same grammatical and lexical knowledge base as the Alpino parser. The idea that a grammar can be used for generation and parsing originated with Kay (1975) and Appelt (1987). In practice, this means that a grammar defines a relation between strings and logical forms. Logical forms are meaning specifications that indicate how information is grouped to encode the representation of the meaning of a sentence (Shieber, 1988; Shieber et al., 1989). They do not specify the actual words or grammatical structures that will be used to communicate this information. During natural language understanding, the task is to arrive at a logical form that corresponds to the input string.

Generation can then be described as the problem of finding a corresponding string for an input logical form. In this light, the grammar should be such that the process is reversible. Whenever parsing produces an analysis tree with a particular root node, generation from this root node should be able to produce the original expression, and vice versa. Grammars for which a parser and a generator can be found, such that the requirement of reversibility is fulfilled are called reversible grammars.

Shieber (1988) had noted that not only a single grammar can be used for parsing and generation, but also the same language-processing architecture can be used for processing the grammar in both directions. He has suggested that charts can be a natural uniform architecture for efficient parsing and generation. This idea has been further refined by Kay (1996), who has noted that chart generation is similar to chart parsing with free word order.

Early generation systems typically depend on large-scale knowledge bases that are built by hand. To improve the robustness of such systems, Knight and Hatzivassiloglou (1995) have introduced a two-level architecture in which a statistical n-gram language model is used to rank the output of a knowledge-based generator. This way, when dealing with new constructions, the knowledge-based system can generate multiple hypotheses and use a language model for the final selection. Also, when faced with incomplete or underspecified input, the language model can help fill in the gaps. In the Alpino generator, in order to select the most fluent sentence from a set of candidate sentences, several statistical models are used. They are further described in De Kok et al. (2011) and De Kok (2013).

7.2.1 Sense Abstract Dependency Trees

One way to represent abstract linguistic input for the realization of sentences is dependency structure (Hays, 1964). A dependency structure represents a sentence as a set of relations connecting all the words in a sentence. These representations are more abstract in comparison to syntactic trees as they do not constrain or prescribe a particular word order. They are therefore more specific in terms of semantics and the notion of relations across words is more explicit. One of the first attempts to implement dependency structure in generation for machine translation has been made by Lin (2004), who has proposed a path-based transfer model, using a word-aligned parallel treebank.

The input to our system were Alpino abstract dependency trees that are described in more detail in Section 3.3.2. They contain words, part-of-speech tags, syntactic categories of constituents, and grammatical dependency relations between heads and dependents, and are therefore purely syntactic representations. Although this method gives a detailed syntactic description of a sentence, it is usually too specific to the grammar for further processing in an MT system. We, therefore, used an even more abstract version of such dependency trees where the lexical nodes do not consist of lemmas but of senses. We, therefore, term them sense abstract dependency trees (SADTs).

In the next section, we describe experiments in lexical choice using SADTs as input. A disadvantage of such input, however, is that our models assume SADTs with correct input senses, which is not always the case. Word Sense Disambiguation is a very difficult task that is still very prone to error. In such cases, where a wrong input sense is given, a lexical choice model is also very likely to fail.

7.3 Lexical Choice from Sense Abstract Dependency Trees

In the next sections, we describe preliminary experiments on lexical choice in SADTs that are the input for generation. We choose lexical items using some very basic techniques in a monolingual setting.

The aim of these experiments is twofold; first, we aimed to test the Alpino generator before implementing it into an MT system. The generator had been developed from a monolingual point of view and it was never tested against real-world input. For the generator to perform well, the lemmas in its input need to be consistent with the grammar. Using a lexical choice module mimics such real world input. The output of these experiments was therefore evaluated manually so that existing problems in the generator could be located. A number of problems emerged and, to make it suitable for MT, subtle adaptations and improvements were necessary, some of which are described in Chapter 3.

As the Alpino generator only uses knowledge from the abstract representations, it is possible that it is not able to generate a full sentence for some of them. This is usually due to words that are not known by the lexicon or to the fact that a chosen lemma can not combine with the other words in the SADT according to the Alpino grammar and/or its dictionary. For the first goal, we, therefore, measured the number of sentences the generator was able to create with the input ADTs.

The second aim of the following experiments was to examine whether the output of such experiments, where simple heuristics are used for lexical choice, could provide information on what is necessary to improve lexical choice. We, therefore, used the information from the conclusions of these experiments to create a more sophisticated method which is described in Chapter 8.

7.3.1 Three Methods for Lexical Choice

The process of lexical choice began with a SADTs that contained WordNet synsets. To select lemmas from each synset we used three different methods:

1. **First lemma**
   The first word in the set of synonyms was chosen. In Cornetto Vossen et al. (2013b), there is no specific order for lemmas in synsets, this heuristic is therefore similar to an arbitrary choice.

2. **Most frequent lemma**
   The lemma that has the highest frequency of all lemmas in the synset was chosen, as counted in a large corpus.

3. **Context information**
   Information on the grammatical context of the word in the SADT was consulted and the lemma that fits its target context best was chosen.

After selecting lemmas, the resulting ADTs contained lemmas instead of synsets and could be used as input to the Alpino generator to generate sentences.

7.3.2 Experiments

Our experiments were undertaken in a monolingual setting using Dutch input sentences. This meant that we analyzed Dutch sentences with the Alpino parser to ADTs from which we generated target sentences with the Alpino generator. However, our lexical choice setup assumed SADTs as input. Therefore, to mimic such SADTs, we first analyzed sentences with Alpino to ADTs and replaced all content words with their most frequent sense, [1] according to Cornetto, to obtain SADTs.

The input sentences were taken from Batch 2 of the QTLeap corpus (Osenova et al., 2015). Furthermore, for the second experiment, which required frequency counts, we used a list of lemmas and their part of speech tags that were derived from a large set of parsed sentences from the SoNaR corpus (Oostdijk et al., 2013).

In the third experiment, we used context information by matching the dependency structures of each lemma in the input synsets with previously seen dependency structures to verify whether they can appear in its context. As this required dependency information, we retrieved dependency triples from the parsed sentences of SoNaR.

[1] At the time of conducting these experiments, the WSD system described in Chapter 5 was not yet developed.

Those triples included a target word, a dependency, and a dependent word. For example, from the sentence "Ik wil een afbeelding invoegen" *(I want to insert an image)*, the following dependency triple would be extracted indicating that the noun *afbeelding* is an object of the main verb *invoegen*:

```
voeg_in, hd/obj1, afbeelding
```

From the resulting dependency triples relations that occur more than 50 times were extracted.

7.3.3 Evaluation

The aim of our experiments is 1. to measure effect on the performance of the generator, more specifically its robustness and 2. to measure the quality of the lexical choices to see what information is necessary for future models to perform well.

Although it was straightforward to evaluate the effects of lexical choice on robustness of the generator in terms of quantity, the evaluation of the quality of lexical choice, however, was rather difficult and subjective since there is no precise standard for evaluating them. We, therefore, assessed the output sentences manually. To evaluate the quality of an output sentence it was assumed that it could fall into three categories: *correct*, *doubtful* or *incorrect*.

An output sentence was considered correct if it conveyed the same meaning as the original utterance. For example, sentence 8a in Example 8 is regarded a good transformation. Although some of the output sentences that have been considered correct can sound rather odd, as they contain very formal or archaic language, they are still considered correct because they have the same literal meaning as compared to the input sentence and fit the grammatical context.

(8) a. Good: *Welke websites kan ik gebruiken om eigendommen → bezittingen te verkopen?*
(What websites can I use to sell ***belongings?****)*

b. Doubtful: De **grootste** → **aanzienlijkste** capaciteit is 4.7 gigabyte.
(The ***biggest*** *→* ***most significant*** *capacity is 4.7 gigabyte)*

c. Incorrect: Is het in Kladblok++ **mogelijk** → **potentiaal** om een blok regels commentaar te geven?
(In Kladblok++, is it ***possible*** *→* ***potential*** *to comment a block of rules?)*

We considered an output sentence doubtful if the meaning of these sentences was quite clear, but not completely the same as the one in the source sentence. An example of such a lexical choice is sentence 8b. A sentence, like the one in sentence 8c, is considered incorrect if it is incomprehensible, ungrammatical or if it has a different meaning.

7.3.4 Results

We first elaborated on the results of the robustness of the generator and showed some sentences that were difficult for the generator to generate. In the second section of the results, for all three experiments, some correct, doubtful and incorrect lexical were discussed.

Robustness of the Generator

In order to compare the robustness of the generator, we first generated the sentences without using lexical choice as a baseline. For this, we analyzed and generated sentences from ADTs to ADTs which can introduce some error. This resulted in 6.4% of sentences that were

First lemma	28.7%
Most frequent lemma	14.8%
Dependency information	8.8%
Generation without lexical choice	6.4%

Table 7.1 | *Performance of the generator after lexical choice, percentage of sentences that are not or not fully generated*

not fully generated. The percentages of sentences that the generator failed to generate using a lexical choice method are shown in Table 7.1.

Choosing the first lemma clearly challenged the generator, that now had issues with 28.7% of the sentences. When choosing the most frequent word, the generator performed better, now struggling with 14.8%. However, the same type of errors occurred mainly because the chosen lemmas could not be used in the contexts. The use of dependency information further improved the ability of the generator to create full sentences as it solved a large amount of such errors. Choosing lemmas guided by dependency information clearly improved its ability to generate full sentences, since it failed to do so with only 8.8%.

Some examples of choices where the generator is not able to generate a sentence because the lemma can not be used in the target context, can be found in Example 9. For instance, in Example 9a, the word *omgangsvorm* (behavior) was chosen which, unlike the original word *manier* (manner), cannot take a verbal complement. The sentence in 9b was not fully generated because *talrijke* (numerous) cannot appear as a determiner while *veel* (many), the original lemma can. In Example 9c, *testen* (to test) was used as an alternative for *focussen* (to focus) which, in the dictionary, cannot be combined with the prepositional complement headed by *op* (on).

(9) a. Is er een eenvoudige **manier** → **omgangsvorm** om bestanden te verplaatsen?
(Is there a safe way ***etiquette*** *to move files?)*

b. Ik heb **veel** → **talrijk** documenten.
(I have a lot of ***numerous*** *documents.)*

c. Ik probeer te **focussen** → **testen** op een afbeelding.
(I try to focus ***test*** *on an image)*

Manual Evaluation

Example 10 shows some sentences with lexical choices that we considered correct. The choice of a lemma did not change the meaning of the original sentences and the lemma fits its context.

(10) a. Hoe **vaak** → **frequent** moeten updates van mijn anti-virus worden uitgevoerd?
(How often do updates of my anti-virus have to be performed?)

b. Wat is een **hacker** → **computerkraker**?
(What is a hacker?)

c. Wat is een **moederbord** → **moederkaart?** *(What is a motherboard?)*

Some *doubtful* output can be found in Example 11. Example 11a has the same meaning as the original sentence. We considered it doubtful because instead of the Dutch word *veilig*, its English translation *safe* is chosen. In Example 11b, some very infrequent words are chosen resulting in a strange output sentence that is, nonetheless, understandable.

(11) a. Hoe herstart ik mijn computer in de **veilige modus** → **safe manier**?
*(How do I reboot my computer in the **safe mode** → **safe manner**?)*

b. Ik wil **graag** → **smachtend evenementen** → **gebeurens** in Google+ zien.
*(I would **love** → **long** to see **events** → **happenings** in Google+)*

The output sentences in Example 10 and 11, demonstrated that choosing the first lemma could yield some good or doubtful output sentences. More interesting for our purpose, however, were incorrect lexical choices. There are several reasons why such errors occur. For example, lexical choices can be correct in some contexts but can be wrong in another one. The domain of the input sentences can be very specific which can be problematic when using a general-domain knowledge base such as WordNet. Our input sentences belong to the IT-domain. Illustrations of such cases where a more general-domain target word is chosen can be found in Example 12a and 12b.

(12) a. Hoe deactiveer ik mijn **openbare** → **algemene profiel** → **zijaanzicht** op LinkedIn? *How do I deactivate my **public** → **general profile** → **side view** on LinkedIn?*

b. Het is een **programma** → **uitzending** voor het aanpassen van afbeeldingen. *It is a program [show] for the adjustment of images*

It should be noted that, most errors that were found during the manual evaluation were due to mistakes in the analysis of SADTs. The majority of such errors consisted of wrong input senses. The most frequent sense we used as input, as tested in Section 5.3.3, was only around 30-40% accurate. At the same time, errors could be caused by the wrong assignment of POS-tags. Since we assigned sense to a lemma with its tag, this could also cause a wrong choice of input sense.

When applying any lexical choice module to a wrong meaning representation, the resulting lemmas can look very strange, mostly funny, in its target contexts. Cases of lexical choice from wrong input senses can be found in example 13, including an example of a wrong input POS tag in Sentence 13d. Although such incorrect input senses contribute to a large number of errors, they are caused in the analysis phase and should, therefore, be solved there.

(13) a. Kan ik een 2.0 **stick** → **joint** in een 3.0 poort gebruiken?
*(Can I use a 2.0 **stick** → **joint** in a 3.0 port?)*

b. Hoe verander ik het formaat van de **tegel** → **tapijttegel**?
*(How do I change the size of a tile **carpet tile?**)*

c. Hoe update ik een **applicatie** → **applicatiewerk**?
*(How do I update an **application** → **handcraft**?)*

d. Selecteer de afbeelding en **pas** → **paspoort** aan met de pijltjestoesten.
*(Select the image and adapt **passport** with the arrow keys.)*

Most Frequent Lemma

Selecting the first word in a synset to choose lemmas in an abstract representation for generation was not a very good heuristic. An experiment where the most frequent lemma was chosen yielded more correct sentences.

However, although more correct sentences were being produced using the most frequent word, more errors arose that are domain specific. This was due to the fact that the data that were used to determine the most frequent lemma belonged to the general domain, while we tested on an IT-domain dataset. An example of this phenomenon can be found in Example 14.

(14) a. Selecteer **actieve** → **bezige** Downloads.
Select the ***active*** → ***busy*** *downloads.*

Here, the transformation does not sound very fluent in this domain, while in the general domain it could.

Dependency Information

The output of the previous experiments indicated that, for lexical choice, it could be beneficiary to look at the surrounding words. Choosing lemmas guided by dependency information, therefore, helped to avoid many errors.

In Example 15, some sentences are shown that were incorrect in the first two experiments and now, with the implementation of dependency information, were correct. For example, in sentence 15a *vensters* (windows) was not replaced by *lichten* (lights). The word does not only have a different meaning (due to a wrong sense input), it could also not appear as a direct object in this context. In sentence 15b, the first word of the fixed expression *high definition* was first replaced with *stoned* but since this lemma does not occur as a modifier of *definition* it was not chosen. Even though a chosen lemma fits the context, it can have a different meaning then the one that was intended. An example of such a lexical choice can be found in sentence15c, where the verb *aanmaken* (to create) was replaced with *aansteken* (to light up).

Here, both words were interchangeable. According to its dependency structures, the lexical choice makes no sense in this context.

(15) a. Hoe kan ik alle **vensters** → **lichten** in Windows minimaliseren?
*(How can I minimalize all **windows** → **lights** in Windows?)*

b. Selecteer HD (**high** → **stoned** definition).
*(Select (HD) **high** → **stoned** definition.))*

c. Hoe kan ik een nieuwe CSS map in Dreamweaver aansteken?
*(*How can I create a new CSS directory in Dreamweaver?)

7.4 Intermediate Conclusions

In this chapter, a preliminary study was discussed in which we challenge the robustness of the Alpino generator in order to prepare it for machine translation. The experiments described in this chapter had an exploratory function aiming to locate existing problems in the generator and to find out what information is necessary to do lexical choice.

The results of using the simple heuristic of choosing the first lemma in a synset demonstrated that lemmas in a synset are not necessarily interchangeable in contexts. Choosing more frequent words, according to large corpora, gave better results while including additional information on the grammatical context of the word improved the output even further. From this, we concluded that information about the frequency of the lemmas and their context was necessary to find appropriate lemmas.

The results of using the simple heuristic of choosing the first lemma in a synset show that lemmas in a synset are not necessarily interchangeable in contexts. Choosing more frequent words, according to large corpora, gives better results while including additional information on the grammatical context of the word improves the output even more. From this we can conclude that information about the frequency of the lemmas and their context is necessary to find appropriate lemmas.

The heuristics that we used in experiment 2 and 3, however, were very naive and did not sufficiently take into account both pieces of information simultaneously. Hence, a more sophisticated system needs to be developed that can find the most optimal lemma while using both frequency and context information simultaneously. In the next chapter, we describe such a model that takes into account both sources of information at the same time.

CHAPTER 8

Lexical Choice with Hidden Markov Tree Models

8.1 Introduction

In the previous chapter we described experiments on lexical choice from SADTs. From the results of these experiments, we concluded that, in order to find a correct word from a given abstract representation, a combination of information on its grammatical context and its frequency is required. Both types of information can easily be obtained: information on the context is available in large parsed corpora, while the frequency of the meaning of a word in particular contexts can be found in sense tagged corpora.

We propose a model that exploits both types of information for the task of lexical choice in SADTs. The output are ADTs, that have the same structure as the SADTs, with lemmas. As both context information and the frequency of the lemma and sense combination, need to be taken into account, we model the SADTs as hidden Markov tree models (HMTMs) Crouse et al. (1996); Durand et al. (2004); Žabokrtský and Popel (2009).

Figure 8.1 contains an example of a HMTM with an SADT on the left and a preferred output ADT on the right. In the SADT, the nodes that correspond to content words, consist of WordNet synsets that need to be mapped to target lemmas on the right. For example, from the synset $\{'poeder','pulver','stof','poeier'\}$, in Figure 8.1, the preferred lemma would be *stof*.

In this chapter, we first describe the model and some related work in Section 8.2. Then, in Section 8.3, we explain the use of HMTM for lexical choice from SADTs. We implement our method in a transfer-based MT system and perform various experiments on it. The description of these experiments and the results can be found in Section 8.4.1

8.2 Hidden Markov Tree Models

Both HMTMs and the well-known HMMs contain observed states with corresponding hidden states (Diligenti et al., 2003; Durand et al., 2004). However, instead of a linear chain of observations, HMTMs map over a tree of observations. Furthermore, analogously to regular HMMs, HMTMs rely on emission probabilities and transition probabilities.

Conditional dependence only along the tree edges is assumed in HMTMs, which corresponds to the intuition behind the linguistic dependency relations in dependency trees. The advantage of considering SADTs as HMTMs is that it allows us to use the tree Viterbi algorithm (Crouse et al., 1996; Durand et al., 2004; Žabokrtský and Popel, 2009) to find the optimal lemmas in their context. As can be seen in Figure 8.1, the process can be interpreted as revealing the hidden

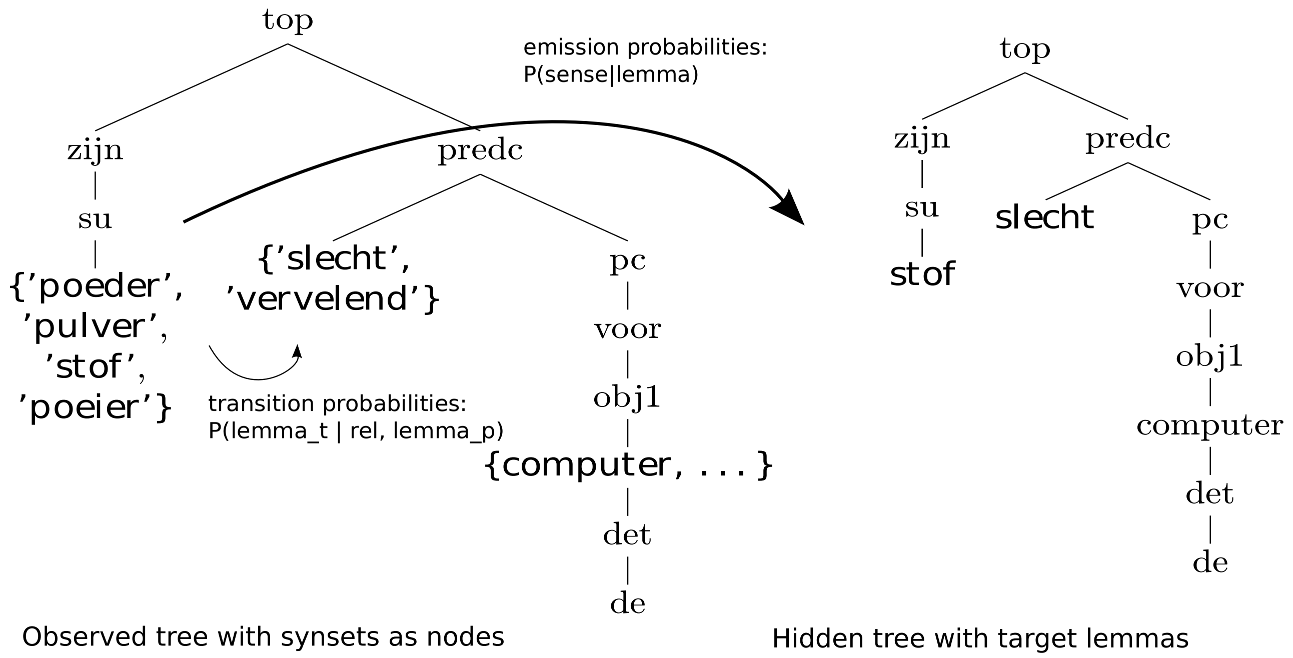
top
zijn
predc
su
{'poeder', 'pulver', 'stof', 'poeier'}
{'slecht', 'vervelend'}
pc
voor
obj1
{computer, . . . }
det
de
emission probabilities: P(sense|lemma)
transition probabilities: P(lemma_t | rel, lemma_p)
top
zijn
predc
su
stof
slecht
pc
voor
obj1
computer
det
de
Observed tree with synsets as nodes
Hidden tree with target lemmas

Figure 8.1 | *HMTM for the Dutch sentence: ′**Stof** is slecht voor de computer (Dust is bad for your computer). All ambiguous context words are represented as senses with optional target lemmas (in parenthesis)* states (the lemmas) in the tree nodes, given another observable labeling of the nodes of the same tree (the senses). The resulting ADTs should then comprise the correct lemmas given their context and can be used as input for generation.

The use of hidden Markov tree models for lexical choice in WordNet synsets is novel. Crouse et al. (1996) have introduced the adaptation of hidden Markov chains to tree models for signal processing. The corresponding adaptation of the classic Viterbi algorithm, used to restore the hidden state tree, has been introduced by Durand et al. (2004). Previous applications of such models are: image segmentation, signal classification, denoising and image document categorization (Durand et al., 2004).

In natural language processing, the use of HMTMs is relatively new and has been applied to word alignment (Kondo et al., 2013) and MT (Žabokrtský and Popel, 2009). Žabokrtský and Popel (2009) were the first to apply HMTMs to the task of lexical choice using a variant of the Viterbi algorithm in the transfer phase of a deep-syntax based machine translation system.

8.3 Method

In the Markov process for lexical choice, we assumed that we were given a directed labeled dependency tree. Its nodes corresponded to word senses, and its edges corresponded to dependency relations. When using HMTMs for lexical choice, the hidden states consisted of lemmas, whereas the observations were word senses (synsets). The hidden states in the tree nodes were revealed on the basis of an observed labeling of the nodes of the same tree.

Hidden Markov tree models are defined by an observed random tree, $S = s_1, \ldots, s_m$ and a hidden random tree, $T = t_1, \ldots, t_m$, that has the same structure as the observed tree where m is the size of the tree. In the HMTM model for lexical choice, the tree is defined by an observed dependency tree containing senses as nodes, and a hidden tree with target lemmas.

The function $\pi : 1, ..., N \rightarrow 0, ..., N$, $\pi(n)$ represents the unique parent of node n with 0 corresponding to the root of the tree l as a label indicating indicating the nature of the dependency. Such labels include subject, object, modifier, and so on.

Each node, except the root node, refers to a sense. We obtain the following distribution on pairs (S, T) of observed senses and target lemmas:

$$P(S,T) = \prod_{n \neq 1} P(t_n | t_{\pi(n)}) P(s_n | t_n) \tag{8.1}$$

In the formal description of the model we can see that the parameters describing HMTMs, are similar to those of regular HMMs, namely: transition probabilities:

$$P(t_n | t_\pi(n))$$

and emission probabilities:

$$P(s_n | t_n)$$

For lexical choice, we derive these probabilities with Bayes' theorem, using prior knowledge of the observed states, as described in the next section.

8.3.1 Probabilities

The required frequency counts for the emission probabilities of a sense given a target lemma can be obtained from sense annotated corpora or from the output of WSD-systems. The probability of an observed state (the sense), given the hidden state (the lemma) can be estimated as follows:

$$P(sense|lemma) \approx \frac{freq(sense, lemma)}{freq(lemma)} \tag{8.2}$$

Consider for instance the probability of the sense $\{lager, beer, ale, ...\}$ given the lemma *beer*. If the lemma *beer* is associated in the corpus with the $\{lager, beer, ale, ...\}$ sense in 89 out of a 100 cases, then the emission probability will be estimated as 0.89.

The transition probabilities of a target lemma $lemma_t$ given a dependency relation rel (such as subject, object or modifier) and its parent $lemma_p$, can be collected from large parsed corpora. For this we can use the following estimation:

$$P\left(lemma_t \mid rel, lemma_p\right) \approx \frac{freq(lemma_p, rel, lemma_t)}{freq(lemma_p, rel)} \tag{8.3}$$

If we want the probability of the lemma *beer* given a parent *drink* in the dependency relation *obj* we compute:

$$p(beer \mid obj, drink) \approx \frac{freq(drink, obj, beer)}{freq(drink, obj)} \tag{8.4}$$

For example, if *drink* occurs 40 times with an object, and in 20 cases that object is the lemma *beer*, then we estimate the probability as 0.5.

8.3.2 Tree Viterbi

With the combination of emission and transition probabilities, the most probable hidden tree labeling given the observed tree labeling can be restored by use of a modification of the traditional Viterbi algorithm for HMMs to a tree Viterbi algorithm for HMTMs. The modification of the original algorithm was introduced by Durand et al. (2004) and Diligenti et al. (2003).

Given the observed tree $S = s_1, \ldots, s_m$, our aim is to find the a hidden state tree, $T = t_1, \ldots, t_m$, maximizing $P(S, T)$. To restore the most probable hidden tree labeling $\hat{T}$ given the observed tree labeling S, using the formula from Equation 8.1, we can write:

$$\begin{aligned}
\hat{T} &= \arg\max_{x} P(T \mid S) \\
&= \arg\max_{x} P(T, S) \\
&= \arg\max_{x} \prod_{n \neq 1} P(t_n | t_{\pi(n)}) P(s_n | t_n)
\end{aligned} \tag{8.5}$$

The tree Viterbi algorithm starts at its leaf nodes and continues upwards. To retrieve the optimal state tree, it is necessary to store the optimal states that correspond to each of the children for each node and each state. Downward recursion is then used along the pointers from the optimal root state in order to retrieve the most probable hidden tree.

8.4 Experiments

Our method can be applied to various NLG problems, such as paraphrasing or summarization. However, as SADTs consist of a combination of dependency trees and word senses, both a dependency parser and a WSD system are required to create them. There are several ways in which we can obtain such trees. For example, we can use a dependency parser to create ADTs and substitute the lexical nodes with word senses. However, in a multilingual setup, SADTs can be obtained in a transfer-based MT system where WSD is performed on the source side of the MT-pipeline.

The advantage of this setup is that we had more realistic data compared to the monolingual setup we used in Chapter 7. Also, this allowed us to quantitatively evaluate our model in terms of BLEU score, as we had the source reference text that could be used to compare our output target text.

In addition, as we evaluated our system in an MT setup, we could also compare it with the original MT-output that did not use the tree-viterbi algorithm for lexical choice. In this way, we could use the lexical choice model only on out of vocabulary (OOV) items, that is, cases where the transfer model was not able to find a translation. We, therefore, carried out two experiments:

1. Lexical choice in an MT setup;

2. Lexical choice in an MT setup for 2. OOVs.

In our experiments, sentences were analyzed and translated from English to Dutch with Treex (see Section 3.2). To obtain senses, we applied WSD on the source side of the pipeline, stored them in the nodes of the dependency tree and retained them during transfer. For this, we used the UKB-WSD module (Agirre and Soroa, 2009) [1] which, at the time of the experiments, was the leading knowledge-based WSD system, in the English analysis phase of the MT pipeline. The English

[1] At the time of conducting these experiments, the WSD system described in Chapter 5 was not yet developed

synsets are then mapped to their Dutch equivalent, which is possible due to the fact fact that most senses are linked between languages. The resulting SADTs were then input to our model. Afterwards, we used the Alpino Generator to generate Dutch sentences on the basis of the output ADTs.

All experiments were carried out on the answers-part of Batch 1 of the QTLeap corpus (Osenova et al., 2015). In the final two experiments, the model was also tested on a broader domain test set. For this, we took 1000 sentences from the English-Dutch News Commentary data set (Tiedemann, 2012).

The emission probabilities were estimated on the basis of Dutch-SemCor (Vossen et al., 2012a). An important issue, however, was that it only contains counts for a limited number of lemmas (approximately 3,000), which could be problematic when estimating emission probabilities. If the target lemma did not appear in the corpus, it was not considered, possibly causing a less suitable lemma to be chosen. In our data, for example, for a synset containing the following senses: *{'bladzijde', 'pagina', 'zijde'}*, the lemma *pagina* would not be chosen as it did not appear in our sense-tagged corpus. We, therefore, assumed, in such cases, that all senses for the word are equally likely.

Transition probabilities were obtained from a large set of parsed sentences. For this, we took the SoNaR part of Lassy Large (van Noord et al., 2013), containing approximately 500 million words. From these parses, dependency relations and their counts were retrieved, resulting in a transition probability matrix that can be queried for each lemma, that appears at least 50 times in the corpus, given its parent lemma and their relationship.

	QTLeap	News
Most frequent	20.16	07.00
Tree Viterbi	21.93	07.45
Original transfer	23.02	08.52

Table 8.1 | *BLEU scores for English-Dutch translation using the tree Viterbi algorithm for lexical choice from SADTs*

8.4.1 Lexical Choice in a Machine Translation Setup

For the evaluation of our model, the output was assessed both manually and by using BLEU score. In WSD, a typical baseline consists of taking the most frequent sense of the target word. We applied the idea behind this baseline to lexical choice by looking at the frequency distribution of a lemma/synset combination in DutchSemCor. In addition, we compared our results to the output of the original transfer-based MT system. However, we did not consider this a fair comparison as the translation model in the MT system is trained on domain-specific data, which makes it easier to translate words correctly, while we used general-domain data. Furthermore, as these data are generated from a different type of input (regular ADTs), they did not have to deal with errors that were introduced by a WSD model.

Table 8.1 shows the results of the use of the tree-viterbi in an MT setup. The algorithm performed better compared to the most frequent baseline. This was confirmed by a manual comparison of the lemmas chosen by our model with the ones chosen by the baseline system.

For manual evaluation, we again randomly selected a subset of 100 sentences from the output of the QTLeap dataset and assessed whether the lexical choices were either correct or incorrect. If the

Lexical choice		Sense input	
Correct	56%		
Incorrect	44%	Correct	11%
		Incorrect	33%

Table 8.2 | *Results of manual evaluation of English-Dutch translation using the tree Viterbi algorithm for lexical choice from SADTs*

choice of lemma was incorrect it was determined whether this was either caused by the model itself or by a wrong input sense, as those errors were beyond the scope of these experiments. The results of this evaluation can be found in Table 8.2 From the lexical choices under consideration, 56% were correct. Of the remaining 44% incorrect choices, 33% are caused by a wrong input sense.

The algorithm outperformed the baseline in most cases. In Example 16,[2] for instance, the baseline system chose the incorrect adjective *bezig* (busy) instead of *actief* (active).

(16) Most frequent: Open je vriend profiel om te kijken of het account nog **bezig** (busy) is.

tree Viterbi: Open je vriend profiel om te kijken of het account nog **actief** (active) is.

Source: Open your friend's profile to see if the account is still **active**.

A similar case can be found in Example 17. Here, the most frequent lemmas were *sterke bedrag* (strong amount) while our model correctly selected *harde schijf* (hard disc).

[2] Note that, next to lexical choices, the examples also contain MT errors

(17) Most frequent: Probeer als je **sterke bedrag** (strong amount) twee USB uitvoeringen heeft om het te verbinden aan beiden.

Tree Viterbi: Probeer als je **harde schijf** (hard drive) twee USB uitvoeringen heeft om het te verbinden aan beiden.

Source: Check if your **hard drive** has two USB configurations to connect it to both.

Here, our model could choose between *bedrag* (amount), *schijf* (disc) and *som* (sum), with the latter having the highest emission probability. For the lemma *hard* it could choose between multiple adjectives, including *sterk*. The combined probabilities ensure that the correct lemmas were chosen by the algorithm.

8.4.2 Lexical Choice for Out of Vocabulary Items

In the MT setup, our model made improvements, in particular, in cases where the original translation model was not able to find a translation and therefore yielded the original, non-translated, word. Using the algorithm for lexical choice only on these OOVs could thus improve the output.

Table 8.3 shows the results[3] of the second experiment on OOVs. The scores for both test sets were similar to those of the system that did not use our model. However, when looking at the output manually, the algorithm did improve the original sentences in almost all cases, yielding more fluent and natural sounding sentences. Such improvements were, for instance, Examples 18 and 19.

(18) Zet je wachtwoord in **rectangle** $\rightarrow$ **rechthoek** en klik op log In.
Insert your password in the rectangle and click Log In

[3]The results of this experiment differ from the previous one as a more recent version of the MT system was used.

	QTLeap	News
Most frequent for OOV	23.00	08.59
Tree Viterbi for OOV	23.03	08.63
Original MT-output	23.02	08.52

Table 8.3 | *Results of English-Dutch translation using the tree Viterbi algorithm only on OOV's*

(19) Probeer de belangrijke combinatie van **brightness** → **helderheid** te controleren.
Try to check the important combination of brightness

In the test data, OOVs were not very common. Our model found a Dutch word for 132 of them, which explained the small difference in the BLEU score. On a random subset of a hundred of sentences, it was manually assessed that 7% of lexical choices were errors made by our model, 33% were errors caused by a wrong input sense, while 60% were good lexical choices.

When using the algorithm only for OOV items, the algorithm made better choices as compared to the most frequent baseline. An example of these better choices can be found in Example 20, where the algorithm correctly preferred the word *fragment* over *brok* (chunk) when indicating a piece of text.

(20) Most frequent: Selecteer het **brok** (chunk) dat je wilt kopiëren.
tree Viterbi: Selecteer het **fragment** (fragment) dat je wilt kopiëren.
Source: Select the **fragment** that you want to copy.

8.5 Discussion

The evaluation of our model shows that it is able to select an appropriate lemma if the correct input sense is available. It also becomes apparent that the algorithm for lexical choice is not very suitable in an MT system. Although the HMTM outperforms the most frequent baseline, the MT system only slightly improves when the algorithm is used on OOVs.

In the MT system, where the transfer model is trained on domain specific parallel data, our system is not able to significantly improve the translation output. The MT systems are trained on domain-specific data, namely the KDE corpus, which probably decreases the chance of translating a lemma incorrectly in the transfer-phase, since they are rarely incorrect. This leaves little room for improving the output by way of lexical choice, while still having the same chance of causing errors. However, as can be seen in Table 8.1 and Table 8.3, our model is able to yield a score similar to the one of the MT system without the use of domain specific parallel data. It could, therefore, be useful in settings where this data is not available.

Another problem that is highly likely to cause errors is mistakes in the analysis and/or the transfer phase. For example, errors in the assignment of part-of-speech tags or dependency relations could have negative effects on the outcome since it would not be possible to find correct transition probabilities in the transition matrix. These errors should be solved in the analysis and/or transfer phase of the MT-pipeline and are therefore beyond the scope of this research.

One option to enhance the use of the module in an MT-system would be to join multiple synsets before our model is used. For example, the UKB system returns confident scores for the different senses.

In cases when the scores for top synsets are close to each other, all the lemmas that belong to more than one synset could be used for the selection. Also, although the use of WordNet as a dictionary source is natural, other dictionaries could be added for better coverage, both mono- and bilingual.

8.6 Conclusions

In this chapter, we aimed to tackle the problem of lexical choice by using HMTMs. A dependency structure over senses is mapped to a dependency structure over lemmas while taking into account both the information of the context and the frequency of the lemma and sense combination.

Evaluation of our model shows that it outperforms the most frequent baseline. Also, the manual evaluations confirm that our model chooses better lemmas as compared to the most frequent baseline. When using the algorithm only on OOV-items it slightly improves an MT system that does not use lexical choice in the generation phase without the use of domain specific parallel data. Our model for lexical choice makes use of a combination of context and frequency information and therefore contributes to the generation of fluent natural sounding sentences.

PART IV

Conclusions

CHAPTER 9

Conclusions

In this thesis, we investigated the use of transfer-based MT systems with interlingual representations of words. We first described work that was done in the QTLeap project that aimed to improve MT by way of including linguistic information, while employing the strengths of statistical approaches. The use of these linguistic features was explored by experimenting with methods that were based on syntactic and semantic representations of natural language utterances.

The main parts of this thesis were devoted to the problem of lexical ambiguity in machine translation, which was split into two separate tasks: word sense disambiguation and lexical choice. First, the words in the source language were disambiguated according to their sense, resulting in interlingual representations of words. To map the input representations to target sentences, a lexical choice module was then used that selected the most appropriate word in the target language.

The first part of this thesis focused on MT and the description of the work that was done in the QTLeap project for the Dutch–English language pair. In Chapter 2, we described the task of MT, and gave an overview of previous work. Then in Chapter 3, A description of the MT systems for Dutch and English developed in the QTLeap project was provided. These systems consist of a combination of the TectoMT system and the Alpino parser and Generator and additional modules that assist their combination.

In Chapter 4, then, an extensive, both automatic and manual, evaluation of the MT systems was described. Manual evaluation, performed by volunteer subjects, focused on the impact of the translation on a helpdesk application. In general, the aim was to assess the added value of the translations in terms of their impact on the performance of the QA system of the helpdesk. This evaluation was divided into two parts. The first part aimed to test the impact of the translation system on the retrieval step of the QA system. We measured the percentage of questions the QA algorithm was able to find a candidate answer to within a certain confidence score interval. Better results were obtained with TectoMT translations than with the SMT baseline system.

The second part of the evaluation, in a real usage scenario, targeted the publication of the answer obtained by the QA system translated into the users language (Dutch) and was carried out by human evaluators. In this step, the English to Dutch translation direction was taken into account and the probability of the necessity of calling an operator was assessed. If we considered only the level where the probability of calling an operator was low, better results were obtained when TectoMT or Chimera were used.

In the majority of the evaluation setups, the baseline SMT system has been significantly, and in some cases substantially, outperformed by both TectoMT and Chimera. This demonstrated that improvements in terms of quality can be gained by combining more linguistically informed systems with statistical information. We explored this idea further in the remainder of the thesis.

In the second part of this thesis, we considered the analysis component of our interlingual setup. In Chapter 5, we described previous work and presented our WSD method that was based on a combination of WordNet and sense extended word embeddings. Our knowledge-based WSD approach selected the best sense for a word according to the distributional similarity with respect to the context of the word and its gloss. Similar to the classic Lesk algorithm, it exploited the idea that overlap between the context of a word and the definition of its senses provides information on its meaning. However, instead of counting the number of words that overlap, we used embeddings to compute the similarity between the gloss of a sense and the context of the word.

Evaluation on both Dutch and English on an all-words task demonstrated that our method outperforms Lesk and improves upon state of the art knowledge-based systems. Additional experiments confirmed the effect of the use of glosses and indicated that our method works well in different domains.

In the same chapter, we explored how to further improve our method by adding two different methods; one that further extends the information in both the context and the glosses by utilizing distributional thesauri (Miller et al., 2012) and one that pre-selects context words using the WordNet hierarchy (Vasilescu et al., 2004). Although using a distributional thesaurus improved the simplified extended Lesk (Ponzetto and Navigli, 2010), it did not improve our method. However, when using a selection scheme before creating context vectors, its performance improved and was better than our knowledge-based baselines.

Although our approach was a straightforward extension to the Lesk algorithm, it achieved better performance compared to Lesk and a random baseline. Our method only required unlabeled text and the glosses of senses and is, therefore, readily applicable for other languages if a knowledge base is available.

To further test the functionality of our system, we participated in the SemEval-2017 Task 7, the subtasks of homographic pun location and homographic pun interpretation. In Chapter 6, we described experiments on different methods of splitting the context that surrounds the target puns and compared our method to several baselines. Instead of solely using the best two output senses of our system we further hypothesized that the idea that pun-based jokes can be divided into two parts, each containing information about the two distinct senses of the pun and can be exploited for pun interpretation.

Experiments were undertaken splitting the context that was input to a WSD system into two parts, running WSD for each context and returning the best sense for pun interpretation. Results of our experiments demonstrated that on the pun interpretation task, systems that use such a module outperformed a WSD system that returned the two best senses. Also, our system performed better compared to both the random and the most frequent baseline.

As we expected the two meanings of a pun to be very dissimilar, we used the output of pun interpretation for pun location. Computing cosine distances between each sense-pair and selecting the one that has the highest distance gained higher scores as compared to a system that randomly selected a content word to be the pun.

The third, and final, part of this thesis addressed the problem of the selection of lexical items from interlingual representations of words in the generation framework of MT. In the generation component of an MT system, words can be selected from a target language WordNet synset. Such synsets, or senses, are the output of a WSD system as described in part II. Unfortunately, not every lemma in a synset is a full synonym of its original word. Hence, the synonyms in a synset are not necessarily interchangeable depending on the context.

Chapter 7 was devoted to the description of the task of lexical choice. It furthermore contains preliminary experiments that were carried out to explore what is necessary for good performance of such a lexical choice module. From these experiments, we concluded that both information of the frequency of the lemma and synset combination and the grammatical context is necessary. We therefore used hidden Markov tree models and a modified Viterbi algorithm, that take this information into account, to select lemmas.

We described our model in Chapter 8 where we also evaluated it extensively by implementing it in a machine translation system. In this setup, the senses of the words were determined in English analysis, and then our model was used to select the best Dutch lemma for the given word sense.

Our model did not improve upon a transfer component trained on a parallel corpus. The results of our experiments indicated that the algorithm for lexical choice on the basis of an HMTM works very well. In terms of BLEU score, our model outperformed a most frequent baseline, in which the most frequent lemma of a given word sense was always chosen. A manual evaluation confirmed that our model was able to select the correct lemma when it was given a correct input synset. The majority of errors were caused by incorrect assignment of the word sense in the English analysis phase.

The original transfer component made few mistakes, with the exception of Out of Vocabulary items (OOVs). In a further experiment we, therefore, only used our model for OOVs and obtained a small improvement in BLEU score. When using the algorithm only on such OOV items, in terms of BLEU score, the system yielded similar scores as compared to the one that does not use lexical choice in the generation phase without the use of domain specific parallel data.

Throughout this thesis we discussed the problem of lexical ambiguity in transfer-based machine translation systems with interlingual representations of words. We described the components of this setup that contribute to solving the problem in three different parts. The first component, the transfer-based MT system was able to significantly outperform an SMT baseline. Also, our WSD method that makes use of word and sense embeddings scores competitively compared to state of the art knowledge-based systems. Our lexical choice model, then, successfully chose lemmas from the interlingual representations of words as compared to a most frequent baseline.

Bibliography

Agirre, E., Baldwin, T., and Martinez, D. (2008). Improving parsing and pp attachment performance with sense information. In *Proceedings of ACL-08: HLT*, pages 317–325. Association for Computational Linguistics.

Agirre, E., De Lacalle, O. L., and Soroa, A. (2009a). Knowledge-based WSD on specific domains: Performing better than generic supervised WSD. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence*, pages 1501–1506.

Agirre, E., Di Nunzio, G. M., Mandl, T., and Otegi, A. (2009b). CLEF 2009 Ad Hoc Track Overview: Robust-WSD task. In *Multilingual information access evaluation I. Text retrieval experiments*, pages 36–49.

Agirre, E. and Edmonds, P., editors (2006). *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*. Springer, 1 edition.

Agirre, E., Lacalle, d. O. L., and Soroa, A. (2014). Random Walks for Knowledge-Based Word Sense Disambiguation. *Computational Linguistics*, 40(1):57–84.

Agirre, E. and Soroa, A. (2009). Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41.

Ananthakrishnan, R., Bhattacharyya, P., Sasikumar, M., and Shah, R. M. (2007). Some Issues in Automatic Evaluation of English-Hindi MT: more Blues for BLEU. *ICON*.

Appelt, D. E. (1987). Bidirectional Grammars and the Design of Natural Language Generation Systems. In *Proceedings of the 1987 Workshop on Theoretical Issues in Natural Language Processing*, TINLAP '87, pages 206–212.

Banerjee, S. and Pedersen, T. (2002). An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In *Computational linguistics and intelligent text processing*, pages 136–145.

Basile, P., Caputo, A., and Semeraro, G. (2014). An Enhanced Lesk Word Sense Disambiguation Algorithm through a Distributional Semantic Model. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 1591–1600.

Bengio, Y., Schwenk, H., Senécal, J.-S., Morin, F., and Gauvain, J.-L. (2006). Neural Probabilistic Language Models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A., Koehn, P., Logacheva, V., Monz, C., Negri, M., Névéol, A., Neves, M., Popel, M., Post, M., Rubino, R., Scarton, C., Specia, L., Turchi, M., Verspoor, K., and Zampieri, M. (2016a). Findings of the 2016 conference on machine translation (WMT16). In Bojar, O. and et al ., editors, *Proceedings of the First Conference on Machine Translation (WMT). Volume 2: Shared Task Papers*, volume 2, pages 131–198.

Bojar, O., Dušek, O., Kocmi, T., Libovický, J., Novák, M., Popel, M., Sudarikov, R., and Variš, D. (2016b). Czeng 1.6: Enlarged czech-english parallel corpus with processing tools dockered. *Lecture Notes in Computer Science*, (9924):231–238.

Bojar, O., Rosa, R., and Tamchyna, A. (2013). *Proceedings of the Eighth Workshop on Statistical Machine Translation*, chapter Chimera – Three Heads for English-to-Czech Translation, pages 92–98.

Bojar, O. and Tamchyna, A. (2015). CUNI in WMT15: Chimera strikes again. In *Proceedings of the 10th WMT*, pages 79–83.

Brody, S. and Lapata, M. (2008). Good Neighbors Make Good Senses: Exploiting Distributional Similarity for Unsupervised WSD. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 65–72, Manchester, UK.

Brown, P., Cocke, J., Della Pietra, S., Della Pietra, V., Jelinek, F., Mercer, R., and Roossin, P. (1988a). A statistical approach to french/english translation. In *In Second International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*, TMI 1988.

Brown, P., Cocke, J., Della Pietra, S., Jelinek, F., Mercer, R., and Roossin, P. (1988b). A statistical approach to language translation. In *Proceedings of the 12th Conference on Computational Linguistics - Volume 1*, COLING '88, pages 71–76.

Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computtational Linguistics*, 19(2):263–311.

Callison-Burch, C. and Osborne, M. (2006). Re-evaluating the role of BLEU in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256.

Chen, X., Liu, Z., and Sun, M. (2014). A Unified Model for Word Sense Representation and Disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1025–1035.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014a). *On the Properties of Neural Machine Translation: Encoder–Decoder Approaches*, pages 103–111.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Crouse, M. S., Baraniuk, R. G., and Nowak, R. D. (1996). Hidden markov models for wavelet-based signal processing. In *Conference Record of the Thirtieth Asilomar Conference on Signals, Systems and Computers*, pages 1029–1035. IEEE.

De Kok, D. (2010). Feature selection for fluency ranking. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 155–163.

De Kok, D. (2013). *Reversible Stochastic Attribute-value Grammars*. PhD thesis.

De Kok, D., Plank, B., and van Noord, G. (2011). Reversible stochastic attribute-value grammars. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 194–199.

Diligenti, M., Frasconi, P., and Gori, M. (2003). Hidden tree markov models for document image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4):519–523.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145.

Durand, J.-B., Goncalves, P., and Guédon, Y. (2004). Computational methods for hidden Markov tree models-an application to wavelet trees. *IEEE Transactions on Signal Processing*, 52(9):2551–2560.

Dušek, O., Žabokrtský, Z., Popel, M., Majliš, M., Novák, M., and Mareček, D. (2012). Formemes in english-czech deep syntactic mt. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 267–274.

Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*.

Escudero, G., Màrquez, L., and Rigau, G. (2000). An empirical study of the domain dependence of supervised word sense disambiguation systems. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 172–180.

Federico, M. and Cettolo, M. (2007). Efficient handling of n-gram language models for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 88–95. Association for Computational Linguistics.

Fellbaum, C., editor (1998). *WordNet An Electronic Lexical Database*. The MIT Press, Cambridge, MA ; London.

Forcada, M. L., Ginestí-Rosell, M., Nordfalk, J., O'Regan, J., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Sánchez-Martínez, F., Ramírez-Sánchez, G., and Tyers, F. M. (2011). Apertium: a free/open-source platform for rule-based machine translation. *Machine translation*, 25(2):127–144.

Francis, W. N. and Kucera, H. (1979). The Brown Corpus: A Standard Corpus of Present-Day Edited American English. Brown University Liguistics Department.

Gaudio, R., Burchardt, A., and Branco, A. (2016). Evaluating machine translation in a usage scenario. In Chair), N. C. C., Choukri, K., Declerck, T., Grobelnik, M., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pages 1–8. European Language Resources Association (ELRA).

H. Nyberg III, E. and Mitamura, T. (1992). The kant system: Fast, accurate, high-quality translation in practical domains. In *COLING 1992 Volume 3: The 15th International Conference on Computational Linguistics*.

Habash, N., Dorr, B., and Monz, C. (2009). Symbolic-to-statistical hybridization: Extending generation-heavy machine translation. *Machine Translation*, 23(1):23–63.

Hajič, J., Čmejrek, M., Dorr, B., Ding, Y., Eisner, J., Gildea, D., Koo, T., Parton, K., Penn, G., Radev, D., and Rambow, O. (2004). Natural language generation in the context of machine translation. Technical report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore.

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

Hildebrand, A. S. and Vogel, S. (2008). Combination of machine translation systems via hypothesis selection from combined n-best lists. In *AMTA 2008 - 8th Conference of the Association for Machine Translation in the Americas*.

Hill, F., Cho, K., Jean, S., Devin, C., and Bengio, Y. (2014). Embedding word similarity with neural machine translation. *CoRR*, abs/1412.6448.

Hirst, G. and St-Onge, D. (1998). Lexical chains as representations of context for the detection and correction of malapropisms. In Fellbaum, C., editor, *WordNet: An Electronic Lexical Database*, pages 305–332. MIT Press.

Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 873–882, Stroudsburg, PA, USA. Association for Computational Linguistics.

Huang, L., Knight, K., and Joshi, A. (2006). *Proceedings of the Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, chapter A Syntax-Directed Translator with Extended Domain of Locality, pages 1–8. Association for Computational Linguistics.

Inkpen, D. Z. and Hirst, G. (2003). Automatic sense disambiguation of the near-synonyms in a dictionary entry. In *Computational Linguistics and Intelligent Text Processing, 4th International Conference*, pages 258–267.

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579.

Jauhar, S. K., Dyer, C., and Hovy, E. H. (2015). Ontologically Grounded Multi-sense Representation Learning for Semantic Vector Space Models. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 683–693.

Jean, S., Firat, O., Cho, K., Memisevic, R., and Bengio, Y. (2015). Montreal neural machine translation systems for wmt'15. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 134–140.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709.

Kay, M. (1975). Syntactic processing and functional sentence perspective. In *Proceedings of the 1975 Workshop on Theoretical Issues in Natural Language Processing*, TINLAP '75, pages 12–15, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kay, M. (1996). Chart generation. In *34th Annual Meeting of the Association for Computational Linguistics*.

Kilgarriff, A. and Rosenzweig, J. (2000). English Senseval: Report and Results. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC-2000)*. European Language Resources Association (ELRA).

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume I, pages 181–184, Detroit, Michigan.

Knight, K. and Hatzivassiloglou, V. (1995). Two-level, many-path generation. In *33rd Annual Meeting of the Association for Computational Linguistics, 26-30 June 1995, MIT, Cambridge, Massachusetts, USA, Proceedings.*, pages 252–260.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, pages 388–395.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86. AAMT, AAMT.

Koehn, P. and Hoang, H. (2007). Factored translation models. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180.

Koehn, P., Och, F. J., and Marcu, D. (2003). 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.

Kondo, S., Duh, K., and Matsumoto, Y. (2013). Hidden markov tree model for word alignment. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, Sofia, Bulgaria.

Lesk, M. (1986). Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, SIGDOC '86, pages 24–26, New York, NY, USA. ACM.

Lin, D. (2004). A path-based transfer model for machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics*.

Luong, T., Pham, H., and Manning, D. C. (2015a). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Luong, T., Sutskever, I., Le, Q., Vinyals, O., and Zaremba, W. (2015b). Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 11–19.

Macken, L., Trushkina, J., and Rura, L. (2007). Dutch parallel corpus: MT corpus and translator's aid. In *In Proceedings of the Machine Translation Summit XI*, pages 313–320.

Mareček, D., Popel, M., and Žabokrtský, Z. (2010). Maximum entropy translation model in dependency-based MT framework. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 201–206, Stroudsburg, PA, USA. Association for Computational Linguistics.

Martin-Löf, P. (1982). Constructive mathematics and computer programming. In Cohen, L. J., o, J., Pfeiffer, H., and Podewski, K.-P., editors, *Logic, Methodology and Philosophy of Science VI, Proceedings of the Sixth International Congress of Logic, Methodology and Philosophy of Science, Hannover 1979*, volume 104 of *Studies in Logic and the Foundations of Mathematics*, pages 153–175. North-Holland.

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of the 43rd annual meeting of the Association for Computational Linguistics*, pages 91–98.

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Melamed, I. D., Green, R., and Turian, J. P. (2003). Precision and recall of machine translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 61–63.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *CoRR*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013b). Distributed Representations of Words and Phrases and their Compositionality. *CoRR*, abs/1310.4546.

Mikolov, T., Yih, W., and Zweig, G. (2013c). Linguistic regularities in continuous space word representations. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 746–751.

Miller, T., Biemann, C., Zesch, T., and Gurevych, I. (2012). Using Distributional Similarity for Lexical Expansion in Knowledge-based Word Sense Disambiguation. In *In Proc. of COLING*.

Miller, T. and Gurevych, I. (2015). Automatic disambiguation of English puns. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 719–729, Beijing, China. Association for Computational Linguistics.

Miller, T., Hempelmann, C. F., and Gurevych, I. (2017a). SemEval-2017 task 7: Detection and interpretation of English puns. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Miller, T., Hempelmann, C. F., and Gurevych, I. (2017b). SemEval-2017 Task 7: Detection and interpretation of English puns. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 59–69.

Miller, T. and Turkovi, M. (2016). Towards the automatic detection and identification of English puns. *European Journal of Humour Research*, 4.

Mitamura, T. and Nyberg, E. (1995). Controlled english for knowledge-based mt: Experience with the kant system. volume 145, pages 146–147.

Morato, J., Marzal, M. N., Llorns, J., and Moreiro, J. (2004). Wordnet applications. In *Proceeding of the Second Global Wordnet Conference*.

Nagao, M. (1984). A framework of a mechanical translation between japanese and english by analogy principle. In *Proceedings of the International NATO Symposium on Artificial and Human Intelligence*, pages 173–180.

Navigli, R. (2009). Word Sense Disambiguation: A Survey. *ACM Comput. Surv.*, 41(2):10:1–10:69.

Navigli, R. and Lapata, M. (2007). Graph Connectivity Measures for Unsupervised Word Sense Disambiguation. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 1683–1688.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

Och, F. J. and Ney, H. (2000). Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 440–447. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.

Oele, D. and Evang, K. (2017). BuzzSaw at SemEval-2017 Task 7: Global vs. Local Context for Interpreting and Locating Homographic English Puns with Sense Embeddings. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 444–448, Vancouver, Canada. Association for Computational Linguistics.

Oele, D. and van Noord, G. (2015). Lexical choice in Abstract Dependency Trees. In *First Deep Machine Translation Workshop*, page 73, Prague, Czech Republic.

Oele, D. and van Noord, G. (2016). Choosing lemmas from Wordnet synsets in Abstract Dependency Trees. In *Workshop on Deep Language Processing for Quality Machine Translation (DeepLP4QMT)*, Varna, Bulgaria.

Oele, D. and van Noord, G. (2017). Distributional Lesk: Effective Knowledge-Based Word Sense Disambiguation. In *International Conference on Computational Semantics (IWCS)*, Montpellier, France.

Oele, D. and van Noord, G. (2018). Simple Embedding-Based Word Sense Disambiguation. In *Global Wordnet Conference 2018*, Singapore, Singapore.

Oostdijk, N., Reynaert, M., Hoste, V., and Schuurman, I. (2013). *Essential Speech and Language Technology for Dutch: Results by the STEVIN-programme*, chapter The Construction of a 500-Million-Word Reference Corpus of Contemporary Written Dutch, pages 219–247. Springer Berlin Heidelberg, Berlin, Heidelberg.

Osenova, P., Gaudio, R. D., Silva, J., Burchardt, A., Popel, M., van Noord, G., Oele, D., and Labaka, G. (2015). Interim report on the curation of language resources and tools for deep mt. Technical Report Deliverable D2.5, Version 2.0, QTLeap Project.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318.

Pelevina, M., Arefiev, N., Biemann, C., and Panchenko, A. (2016). Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183.

Pollard, C. and Sag, I. A. (1994). *Head-driven phrase structure grammar*. University of Chicago Press.

Ponzetto, S. P. and Navigli, R. (2010). Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1522–1531. Association for Computational Linguistics.

Popel, M. and Žabokrtský, Z. (2010). TectoMT: Modular NLP framework. In *Proceedings of the 7th International Conference on Advances in Natural Language Processing*, IceTAL'10, pages 293–304.

Popović, M. (2012). rgbf: An open source tool for n-gram based automatic evaluation of machine translation output. *The Prague Bulletin of Mathematical Linguistics*, 98:99–108.

Pradhan, S. S., Loper, E., Dligach, D., and Palmer, M. (2007). SemEval-2007 Task 17: English Lexical Sample, SRL and All Words. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 87–92, Stroudsburg, PA, USA. Association for Computational Linguistics.

Raganato, A., Camacho-Collados, J., and Navigli, R. (2017). Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110. Association for Computational Linguistics.

Ranta, A. (2011). *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford. ISBN-10: 1-57586-626-9 (Paper), 1-57586-627-7 (Cloth).

Reisinger, J. and Mooney, R. J. (2010). Multi-prototype vector-space models of word meaning. In *Proceedings of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-2010)*, pages 109–117.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press, New York, NY, USA.

Resnik, P. (2006). Wsd in nlp applications. *Word sense disambiguation: algorithms and applications*, pages 299–337.

Rosa, R., Duek, O., Novák, M., and Popel, M. (2015). Translation Model Interpolation for Domain Adaptation in TectoMT. In *Proceedings of the 1st Deep Machine Translation Workshop*, pages 89–96.

Rosa, R., Popel, M., Bojar, O., Mareček, D., and Dušek, O. (2016). Proceedings of the 2nd deep machine translation workshop. pages 1–10. ÚFAL MFF UK.

Rosti, A. V., Ayan, N. F., Xiang, B., Matsoukas, S., Schwartz, R., and Dorr, B. (2007). Combining Outputs from Multiple Machine Translation Systems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 228–235, Rochester, New York.

Rothe, S. and Schütze, H. (2015). AutoExtend: Extending Word Embeddings to Embeddings for Synsets and Lexemes. In *Proceedings of the ACL*.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725.

Sgall, P., Hajiová, E., and Panevová, J. (1986). *The meaning of the sentence in its semantic and pragmatic aspects*. D. Reidel, Dordrecht.

Shieber, S. M. (1988). A uniform architecture for parsing and generation. In *Proceedings of the 12th Conference on Computational Linguistics - Volume 2*, COLING '88, pages 614–619.

Shieber, S. M., van Noord, G., Moore, R. C., and Pereira, F. C. N. (1989). A semantic-head-driven generation algorithm for unification-based formalisms. In *Proceedings of the 27th Annual Meeting on Association for Computational Linguistics*, ACL '89, pages 7–17.

Snyder, B. and Palmer, M. (2004). The English All-words Task. In Mihalcea, R. and Edmonds, P., editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43, Barcelona, Spain. Association for Computational Linguistics.

Socher, R., Lin, C. C., Manning, C., and Ng, A. Y. (2011). Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136.

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

Spoustová, D., Hajič, J., Votrubec, J., Krbec, P., and Květoň, P. (2007). The best of two worlds: Cooperation of statistical and rule-based taggers for czech. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing 2007*, pages 67–74.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence-to-sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NIPS'14, pages 3104–3112.

Taghipour, K. and Ng, H. T. (2015). Semi-Supervised Word Sense Disambiguation Using Word Embeddings in General and Specific Domains. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 314–323, Denver, Colorado. Association for Computational Linguistics.

Thurmair, G. (2009). Comparing different architectures of hybrid machine translation systems. In *MT Summit XII: proceedings of the twelfth Machine Translation Summit*, volume 39, pages 340–347.

Tiedemann, J. (2012). Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*.

Toma, P. (1977). Systran as a multilingual machine translation system. In *Proceedings of the Third European Congress on Information Systems and Networks, Overcoming the language barrier*, pages 569–581.

Toutanova, K., Klein, D., D. Manning, C., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics.

Turian, J. P., Shea, L., and Melamed, I. D. (2006). Evaluation of machine translation and its evaluation. Technical report, DTIC Document.

Turney, P. D. (2006). Similarity of Semantic Relations. *Computational Linguistics*, 32(3):379–416.

Van Eerten, L. (2007). Over het corpus gesproken nederlands. *Nederlandse Taalkunde*, 12(3):194–215.

van Noord, G. (1990). An overview of head-driven bottom-up generation. In *Current research in natural language generation*, pages 141–165. Academic Press Professional, Inc.

van Noord, G. (2006). At last parsing is now operational. In *TALN 2006 Verbum Ex Machina, Actes De La 13e Conference sur Le Traitement Automatique des Langues naturelles*, pages 20–42, Leuven.

van Noord, G., Bouma, G., Van Eynde, F., de Kok, D., van der Linde, J., Schuurman, I., Sang, E. T. K., and Vandeghinste, V. (2013). *Large Scale Syntactic Annotation of Written Dutch: Lassy*, pages 147–164. Springer Berlin Heidelberg, Berlin, Heidelberg.

van Noord, G., Schuurman, I., and Bouma, G. (2011). Lassy syntactische annotatie revision: 19455.

Vasilescu, F., Langlais, P., and Lapalme, G. (2004). Evaluating variants of the Lesk approach for disambiguating words. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*.

Vauquois, B. (1968). A survey of formal grammars and algorithms for recognition and transformation in machine translation. In *IFIP Congress-68*, pages 254–260, Edinburgh.

Vogel, S., Ney, H., and Tillmann, C. (1996). HMM-based word alignment in statistical translation. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*, COLING '96, pages 836–841.

Vossen, P., editor (1998). *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers, Norwell, MA, USA.

Vossen, P., Görög, A., Izquierdo, R., and Van den Bosch, A. (2012a). Dutchsemcor: Targeting the ideal sense-tagged corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*. European Language Resources Association (ELRA).

Vossen, P., Görög, A., Izquierdo, R., and den Bosch Antal, V. (2012b). DutchSemCor: Targeting the Ideal Sense-tagged Corpus. In Chair), N. C. C., Choukri, K., Declerck, T., Doan, M. U., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Vossen, P., Izquierdo, R., and Görög, A. (2013a). Dutchsemcor: in quest of the ideal sense-tagged corpus. In Angelova, G., Bontcheva, K., and Mitkov, R., editors, *RANLP*, pages 710–718. RANLP 2013 Organising Committee / ACL.

Vossen, P., Rigau, G., Alegria, I., Agirre, E., Farwell, D., Fuentes, M., et al. (2006). Meaningful Results for Information Retrieval in the MEANING Project. In *Proc. of the 3rd Global Wordnet Conference*, pages 22–26.

Vossen, P., van der Vliet, H., Maks, I., Segers, R., Moens, M.-F., Hofmann, K., Tjong Kim Sang, E., and de Rijke, M., editors (2013b). *Cornetto: A Combinatorial Lexical Semantic Database for Dutch*. Springer.

Žabokrtský, Z. (2010). *From Treebanking to Machine Translation*. Habilitation thesis, Charles University in Prague.

Žabokrtský, Z. and Popel, M. (2009). Hidden Markov tree model in dependency-based machine translation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 145–148.

Žabokrtský, Z., Ptáček, J., and Pajas, P. (2008). Tectomt: Highly modular MT system with tectogrammatics used as transfer layer. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 167–170.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Zeman, D. (2008). Reusable Tagset Conversion Using Tagset Drivers. In *Proceedings of LREC*, pages 213–218.

Zhong, Z. and Ng, H. T. (2010). It Makes Sense: A Wide-coverage Word Sense Disambiguation System for Free Text. In *Proceedings of the ACL 2010 System Demonstrations*, ACLDemos '10, pages 78–83, Stroudsburg, PA, USA. Association for Computational Linguistics.

Samenvatting in het Nederlands

Een van de grootste uitdagingen in het automatisch vertalen van tekst is dat woorden verschillende betekenissen kunnen hebben in verschillende contexten. Dit probleem staat ook wel bekend als lexicale ambiguiteit. Een voorbeeld van zulke ambiguiteit is te vinden in voorbeeld 21 voor de Engelse woorden "drive" and "port".

(21) a. British tourists could soon be allowed to **drive** on the left in the **port** of Calais.
Britse toeristen zouden binnenkort wellicht links mogen rijden in de haven van Calais

b. An external **drive** typically includes a bridging device between its interface to a USB interface **port**.
Een externe schijf bevat meestal een overbruggingsapparaat tussen de interface en de USB-interface poort

Een van de meest eenvoudige manieren om automatisch (digitaal) te vertalen is om elk woord in een zin in de brontaal direct te vertalen naar een corresponderend woord in de doeltaal. Dit werkt helaas niet zo goed, vooral wanneer men het vertaalsysteem zou willen gebruiken in verschillende domeinen, zoals in voorbeeld 21. Als men zin 21a zou willen vertalen naar het Nederlands, zouden de Engelse woorden "drive" en "port" vertaald moeten worden naar "rijden" en "haven". In de tweede zin zou het echter toepasselijker zijn om dezelfde woorden te vertalen met "schijf" en "poort". Om een automatisch vertaalsysteem goed te laten functioneren is het daarom belangrijk dat het systeem zowel begrip heeft van de betekenis van de afzonderlijke woorden als die van de gehele zin.

Als een alternatief voor vertaalsystemen die simpelweg elk woord in een zin vertalen naar een gelijkwaardig woord in de doeltaal, maken interlinguale en transfersystemen gebruik van een taalkundige analyse van de zin in de brontaal. In interlinguale systemen is deze analyse zo grondig dat er geen taalafhankelijke kenmerken meer over blijven. Het resultaat van de analyse bestaat dan ook uit een zogenaamde interlinguale (of betekenis) representatie van de zin. Vanuit deze representaties kunnen vervolgens weer nieuwe zinnen gegenereerd worden in de doeltaal. Ook transfersystemen maken gebruik van een grondige taalkundige analyse die resulteert in een structuur die de grammaticale en semantische betekenis van een zin weergeeft. In tegenstelling tot interlinguale systemen maken deze echter gebruik van een taalspecifieke transfer van een brontaalstructuur naar een doeltaalstructuur.

In deze dissertatie maken wij gebruik van een transfersysteem met interlinguale representaties van woorden. In de analysefase van een dergelijk vertaalsysteem wordt eerst voor elk woord bepaald wat zijn betekenis is in de context. In zin 21b, zou een ideaal systeem bijvoorbeeld eerst bepalen dat het woord "drive" een computerapparaat is en "poort" een computercircuit weergeeft. Deze informatie over de betekenis van het woord wordt vervolgens opgeslagen als een interlinguale weergave zodat deze tijdens de transfer fase ongewijzigd blijft.

Aan de andere kant van het systeem, in de generatiefase, moeten vervolgens lemma's worden geselecteerd uit deze interlinguale representaties. Omdat deze representaties abstract zijn, kunnen verschillende woorden worden gebruikt om de betekenis ervan uit te drukken. Ze passen echter niet altijd even goed in de doelcontext. Voorbeeld 9 bevat zo'n Nederlandse vertaling waarin de representaties uit de doeltaal behouden zijn. In zin 21b, voor het woord "drive" met de betekenis van computerhardware, zijn de optionele doellemma's {'disk', 'magneetschijf', 'schijf'} terwijl, voor " port" gekozen kan worden {"interface", "poort"}.

(22) Een externe {"disk", "magneetschijf", "schijf"} bevat meestal een overbruggingsapparaat tussen de interface en een USB {"interface", "poort"}.
(An external {disk, magnetic disc, drive} typically includes a bridging device between its interface to a USB {"interface", "port"}.)

In tegenstelling tot de lemma's "schijf" en "poort" die goed bij deze context zouden passen, zouden de andere, hoewel ze een vergelijkbare betekenis hebben, in deze context tot een minder vloeiende zin leiden. Daarom wordt in de generatiefase een module gebruikt die het lemma selecteert dat het beste past in de doelcontext.

In dit proefschrift onderzoeken we het gebruik van transersystemen met interlinguale representaties van woorden. Op deze manier benaderen we het probleem van de lexicale ambiguiteit in de automatische vertaalsystemen als twee afzonderlijke taken: het bepalen van woordbetekenis en lexicale selectie. Ten eerste worden de woorden in de brontaal op basis van hun betekenis gedesambigueerd, wat resulteert in interlinguale representaties van woorden. Vervolgens wordt een lexicale selectiemodule gebruikt die het meest geschikte woord in de doeltaal selecteert.

Dit proefschrift bestaat uit drie delen. Het eerste deel concentreert zich op automatische vertaalsystemen en beschrijft onderzoek dat werd gedaan binnen het QTLeap-project (een Europees samenwerkings project voor automatische vertaalsystemen) voor de Nederlands-Engelse vertaalsystemen. We geven een gedetailleerde beschrijving van de ontwikkeling en evaluatie van vertaalsystemen voor Nederlands-Engels in het QTLeap-project. Dit biedt een achtergrond voor de experimenten in het tweede en derde deel van dit proefschrift.

In het tweede deel beschrijven we een methode die de betekenis van woorden bepaalt. Deze is vergelijkbaar is met het klassieke Lesk-algoritme, omdat het gebruik maakt van het idee dat gedeelde woorden tussen de context van een woord en zijn definitie informatie over de betekenis ervan verschaffen. Wij gebruiken echter, in plaats daarvan, woord- en betekenisvectoren om de overeenkomst te berekenen tussen de definitie van een betekenis en de context van een woord. We gebruiken onze methode bovendien voor het lokaliseren- en interpreteren van woordgrapjes.

In het derde deel, ten slotte, presenteren we een model voor lexicale keuze dat lemma's selecteert, gegeven de abstracte representaties van woorden. Dit model houdt rekening met de waarschijnlijkheid van een lemma gezien zijn context, en met de waarschijnlijkheid van een betekenis van een lemma gegeven zijn betekenis.

Groningen Dissertations in Linguistics (GRODIL)

1. Henriëtte de Swart (1991). *Adverbs of Quantification: A Generalized Quantifier Approach.*
2. Eric Hoekstra (1991). *Licensing Conditions on Phrase Structure.*
3. Dicky Gilbers (1992). *Phonological Networks. A Theory of Segment Representation.*
4. Helen de Hoop (1992). *Case Configuration and Noun Phrase Interpretation.*
5. Gosse Bouma (1993). *Nonmonotonicity and Categorial Unification Grammar.*
6. Peter Blok (1993). *The Interpretation of Focus: an epistemic approach to pragmatics.*
7. Roelien Bastiaanse (1993). *Studies in Aphasia.*
8. Bert Bos (1993). *Rapid User Interface Development with the Script Language Gist.*
9. Wim Kosmeijer (1993). *Barriers and Licensing.*
10. Jan-Wouter Zwart (1993). *Dutch Syntax: A Minimalist Approach.*
11. Mark Kas (1993). *Essays on Boolean Functions and Negative Polarity.*
12. Ton van der Wouden (1994). *Negative Contexts.*
13. Joop Houtman (1994). *Coordination and Constituency: A Study in Categorial Grammar.*
14. Petra Hendriks (1995). *Comparatives and Categorial Grammar.*
15. Maarten de Wind (1995). *Inversion in French.*
16. Jelly Julia de Jong (1996). *The Case of Bound Pronouns in Peripheral Romance.*

17. Sjoukje van der Wal (1996). *Negative Polarity Items and Negation: Tandem Acquisition.*
18. Anastasia Giannakidou (1997). *The Landscape of Polarity Items.*
19. Karen Lattewitz (1997). *Adjacency in Dutch and German.*
20. Edith Kaan (1997). *Processing Subject-Object Ambiguities in Dutch.*
21. Henny Klein (1997). *Adverbs of Degree in Dutch.*
22. Leonie Bosveld-de Smet (1998). *On Mass and Plural Quantification: The Case of French 'des'/'du'-NPs.*
23. Rita Landeweerd (1998). *Discourse Semantics of Perspective and Temporal Structure.*
24. Mettina Veenstra (1998). *Formalizing the Minimalist Program.*
25. Roel Jonkers (1998). *Comprehension and Production of Verbs in Aphasic Speakers.*
26. Erik F. Tjong Kim Sang (1998). *Machine Learning of Phonotactics.*
27. Paulien Rijkhoek (1998). *On Degree Phrases and Result Clauses.*
28. Jan de Jong (1999). *Specific Language Impairment in Dutch: Inflectional Morphology and Argument Structure.*
29. Hae-Kyung Wee (1999). *Definite Focus.*
30. Eun-Hee Lee (2000). *Dynamic and Stative Information in Temporal Reasoning: Korean Tense and Aspect in Discourse.*
31. Ivilin Stoianov (2001). *Connectionist Lexical Processing.*
32. Klarien van der Linde (2001). *Sonority Substitutions.*
33. Monique Lamers (2001). *Sentence Processing: Using Syntactic, Semantic, and Thematic Information.*
34. Shalom Zuckerman (2001). *The Acquisition of "Optional" Movement.*
35. Rob Koeling (2001). *Dialogue-Based Disambiguation: Using Dialogue Status to Improve Speech Understanding.*
36. Esther Ruigendijk (2002). *Case Assignment in Agrammatism: a Cross-linguistic Study.*
37. Tony Mullen (2002). *An Investigation into Compositional Features and Feature Merging for Maximum Entropy-Based Parse Selection.*
38. Nanette Bienfait (2002). *Grammatica-onderwijs aan allochtone jongeren.*
39. Dirk-Bart den Ouden (2002). *Phonology in Aphasia: Syllables and Segments in Level-specific Deficits.*

40. Rienk Withaar (2002). *The Role of the Phonological Loop in Sentence Comprehension.*
41. Kim Sauter (2002). *Transfer and Access to Universal Grammar in Adult Second Language Acquisition.*
42. Laura Sabourin (2003). *Grammatical Gender and Second Language Processing: An ERP Study.*
43. Hein van Schie (2003). *Visual Semantics.*
44. Lilia Schürcks-Grozeva (2003). *Binding and Bulgarian.*
45. Stasinos Konstantopoulos (2003). *Using ILP to Learn Local Linguistic Structures.*
46. Wilbert Heeringa (2004). *Measuring Dialect Pronunciation Differences using Levenshtein Distance.*
47. Wouter Jansen (2004). *Laryngeal Contrast and Phonetic Voicing: A Laboratory Phonology Approach to English, Hungarian and Dutch.*
48. Judith Rispens (2004). *Syntactic and Phonological Processing in Developmental Dyslexia.*
49. Danielle Bougaïré (2004). *L'approche communicative des campagnes de sensibilisation en santé publique au Burkina Faso: les cas de la planification familiale, du sida et de l'excision.*
50. Tanja Gaustad (2004). *Linguistic Knowledge and Word Sense Disambiguation.*
51. Susanne Schoof (2004). *An HPSG Account of Nonfinite Verbal Complements in Latin.*
52. M. Begoña Villada Moirón (2005). *Data-driven identification of fixed expressions and their modifiability.*
53. Robbert Prins (2005). *Finite-State Pre-Processing for Natural Language Analysis.*
54. Leonoor van der Beek (2005). *Topics in Corpus-Based Dutch Syntax.*
55. Keiko Yoshioka (2005). *Linguistic and gestural introduction and tracking of referents in L1 and L2 discourse.*
56. Sible Andringa (2005). *Form-focused instruction and the development of second language proficiency.*
57. Joanneke Prenger (2005). *Taal telt! Een onderzoek naar de rol van taalvaardigheid en tekstbegrip in het realistisch wiskundeonderwijs.*
58. Neslihan Kansu-Yetkiner (2006). *Blood, Shame and Fear: Self-Presentation Strategies of Turkish Women's Talk about their Health and Sexuality.*

59. Mónika Z. Zempléni (2006). *Functional imaging of the hemispheric contribution to language processing.*
60. Maartje Schreuder (2006). *Prosodic Processes in Language and Music.*
61. Hidetoshi Shiraishi (2006). *Topics in Nivkh Phonology.*
62. Tamás Biró (2006). *Finding the Right Words: Implementing Optimality Theory with Simulated Annealing.*
63. Dieuwke de Goede (2006). *Verbs in Spoken Sentence Processing: Unraveling the Activation Pattern of the Matrix Verb.*
64. Eleonora Rossi (2007). *Clitic production in Italian agrammatism.*
65. Holger Hopp (2007). *Ultimate Attainment at the Interfaces in Second Language Acquisition: Grammar and Processing.*
66. Gerlof Bouma (2008). *Starting a Sentence in Dutch: A corpus study of subject- and object-fronting.*
67. Julia Klitsch (2008). *Open your eyes and listen carefully. Auditory and audiovisual speech perception and the McGurk effect in Dutch speakers with and without aphasia.*
68. Janneke ter Beek (2008). *Restructuring and Infinitival Complements in Dutch.*
69. Jori Mur (2008). *Off-line Answer Extraction for Question Answering.*
70. Lonneke van der Plas (2008). *Automatic Lexico-Semantic Acquisition for Question Answering.*
71. Arjen Versloot (2008). *Mechanisms of Language Change: Vowel reduction in 15th century West Frisian.*
72. Ismail Fahmi (2009). *Automatic term and Relation Extraction for Medical Question Answering System.*
73. Tuba Yarbay Duman (2009). *Turkish Agrammatic Aphasia: Word Order, Time Reference and Case.*
74. Maria Trofimova (2009). *Case Assignment by Prepositions in Russian Aphasia.*
75. Rasmus Steinkrauss (2009). *Frequency and Function in WH Question Acquisition. A Usage-Based Case Study of German L1 Acquisition.*
76. Marjolein Deunk (2009). *Discourse Practices in Preschool. Young Children's Participation in Everyday Classroom Activities.*
77. Sake Jager (2009). *Towards ICT-Integrated Language Learning: Developing an Implementation Framework in terms of Pedagogy, Technology and Environment.*

78. Francisco Dellatorre Borges (2010). *Parse Selection with Support Vector Machines.*
79. Geoffrey Andogah (2010). *Geographically Constrained Information Retrieval.*
80. Jacqueline van Kruiningen (2010). *Onderwijsontwerp als conversatie. Probleemoplossing in interprofessioneel overleg.*
81. Robert G. Shackleton (2010). *Quantitative Assessment of English-American Speech Relationships.*
82. Tim Van de Cruys (2010). *Mining for Meaning: The Extraction of Lexico-semantic Knowledge from Text.*
83. Therese Leinonen (2010). *An Acoustic Analysis of Vowel Pronunciation in Swedish Dialects.*
84. Erik-Jan Smits (2010). *Acquiring Quantification. How Children Use Semantics and Pragmatics to Constrain Meaning.*
85. Tal Caspi (2010). *A Dynamic Perspective on Second Language Development.*
86. Teodora Mehotcheva (2010). *After the fiesta is over. Foreign language attrition of Spanish in Dutch and German Erasmus Students.*
87. Xiaoyan Xu (2010). *English language attrition and retention in Chinese and Dutch university students.*
88. Jelena Prokić (2010). *Families and Resemblances.*
89. Radek Šimík (2011). *Modal existential wh-constructions.*
90. Katrien Colman (2011). *Behavioral and neuroimaging studies on language processing in Dutch speakers with Parkinson's disease.*
91. Siti Mina Tamah (2011). *A Study on Student Interaction in the Implementation of the Jigsaw Technique in Language Teaching.*
92. Aletta Kwant (2011). *Geraakt door prentenboeken. Effecten van het gebruik van prentenboeken op de sociaal-emotionele ontwikkeling van kleuters.*
93. Marlies Kluck (2011). *Sentence amalgamation.*
94. Anja Schüppert (2011). *Origin of asymmetry: Mutual intelligibility of spoken Danish and Swedish.*
95. Peter Nabende (2011). *Applying Dynamic Bayesian Networks in Transliteration Detection and Generation.*
96. Barbara Plank (2011). *Domain Adaptation for Parsing.*
97. Çağrı Çöltekin (2011). *Catching Words in a Stream of Speech: Computational simulations of segmenting transcribed child-directed speech.*

98. Dörte Hessler (2011). *Audiovisual Processing in Aphasic and Non-Brain-Damaged Listeners: The Whole is More than the Sum of its Parts.*
99. Herman Heringa (2012). *Appositional constructions.*
100. Diana Dimitrova (2012). *Neural Correlates of Prosody and Information Structure.*
101. Harwintha Anjarningsih (2012). *Time Reference in Standard Indonesian Agrammatic Aphasia.*
102. Myrte Gosen (2012). *Tracing learning in interaction. An analysis of shared reading of picture books at kindergarten.*
103. Martijn Wieling (2012). *A Quantitative Approach to Social and Geographical Dialect Variation.*
104. Gisi Cannizzaro (2012). *Early word order and animacy.*
105. Kostadin Cholakov (2012). *Lexical Acquisition for Computational Grammars. A Unified Model.*
106. Karin Beijering (2012). *Expressions of epistemic modality in Mainland Scandinavian. A study into the lexicalization-grammaticalization-pragmaticalization interface.*
107. Veerle Baaijen (2012). *The development of understanding through writing.*
108. Jacolien van Rij (2012). *Pronoun processing: Computational, behavioral, and psychophysiological studies in children and adults.*
109. Ankelien Schippers (2012). *Variation and change in Germanic long-distance dependencies.*
110. Hanneke Loerts (2012). *Uncommon gender: Eyes and brains, native and second language learners, & grammatical gender.*
111. Marjoleine Sloos (2013). *Frequency and phonological grammar: An integrated approach. Evidence from German, Indonesian, and Japanese.*
112. Aysa Arylova (2013). *Possession in the Russian clause. Towards dynamicity in syntax.*
113. Daniël de Kok (2013). *Reversible Stochastic Attribute-Value Grammars.*
114. Gideon Kotzé (2013). *Complementary approaches to tree alignment: Combining statistical and rule-based methods.*
115. Fridah Katushemererwe (2013). *Computational Morphology and Bantu Language Learning: an Implementation for Runyakitara.*
116. Ryan C. Taylor (2013). *Tracking Referents: Markedness, World Knowledge and Pronoun Resolution.*

117. Hana Smiskova-Gustafsson (2013). *Chunks in L2 Development: A Usage-Based Perspective.*
118. Milada Walková (2013). *The aspectual function of particles in phrasal verbs.*
119. Tom O. Abuom (2013). *Verb and Word Order Deficits in Swahili-English bilingual agrammatic speakers.*
120. Gülsen Yılmaz (2013). *Bilingual Language Development among the First Generation Turkish Immigrants in the Netherlands.*
121. Trevor Benjamin (2013). *Signaling Trouble: On the linguistic design of other-initiation of repair in English Conversation.*
122. Nguyen Hong Thi Phuong (2013). *A Dynamic Usage-based Approach to Second Language Teaching.*
123. Harm Brouwer (2014). *The Electrophysiology of Language Comprehension: A Neurocomputational Model.*
124. Kendall Decker (2014). *Orthography Development for Creole Languages.*
125. Laura S. Bos (2015). *The Brain, Verbs, and the Past: Neurolinguistic Studies on Time Reference.*
126. Rimke Groenewold (2015). *Direct and indirect speech in aphasia: Studies of spoken discourse production and comprehension.*
127. Huiping Chan (2015). *A Dynamic Approach to the Development of Lexicon and Syntax in a Second Language.*
128. James Griffiths (2015). *On appositives.*
129. Pavel Rudnev (2015). *Dependency and discourse-configurationality: A study of Avar.*
130. Kirsten Kolstrup (2015). *Opportunities to speak. A qualitative study of a second language in use.*
131. Güliz Güne (2015). *Deriving Prosodic structures.*
132. Cornelia Lahmann (2015). *Beyond barriers. Complexity, accuracy, and fluency in long-term L2 speakers speech.*
133. Sri Wachyunni (2015). *Scaffolding and Cooperative Learning: Effects on Reading Comprehension and Vocabulary Knowledge in English as a Foreign Language.*
134. Albert Walsweer (2015). *Ruimte voor leren. Een etnogafisch onderzoek naar het verloop van een interventie gericht op versterking van het taalgebruik in een knowledge building environment op kleine Friese basisscholen.*
135. Aleyda Lizeth Linares Calix (2015). *Raising Metacognitive Genre Awareness in L2 Academic Readers and Writers.*

136. Fathima Mufeeda Irshad (2015). *Second Language Development through the Lens of a Dynamic Usage-Based Approach.*
137. Oscar Strik (2015). *Modelling analogical change. A history of Swedish and Frisian verb inflection.*
138. He Sun (2015). *Predictors and stages of very young child EFL learners English development in China.*
139. Marieke Haan (2015). *Mode Matters. Effects of survey modes on participation and answering behavior.*
140. Nienke Houtzager (2015). *Bilingual advantages in middle-aged and elderly populations.*
141. Noortje Joost Venhuizen (2015). *Projection in Discourse: A data-driven formal semantic analysis.*
142. Valerio Basile (2015). *From Logic to Language: Natural Language Generation from Logical Forms.*
143. Jinxing Yue (2016). *Tone-word Recognition in Mandarin Chinese: Influences of lexical-level representations.*
144. Seçkin Arslan (2016). *Neurolinguistic and Psycholinguistic Investigations on Evidentiality in Turkish.*
145. Rui Qin (2016). *Neurophysiological Studies of Reading Fluency. Towards Visual and Auditory Markers of Developmental Dyslexia.*
146. Kashmiri Stec (2016). *Visible Quotation: The Multimodal Expression of Viewpoint.*
147. Yinxing Jin (2016). *Foreign language classroom anxiety: A study of Chinese university students of Japanese and English over time.*
148. Joost Hurkmans (2016). *The Treatment of Apraxia of Speech. Speech and Music Therapy, an Innovative Joint Effort.*
149. Franziska Köder (2016). *Between direct and indirect speech: The acquisition of pronouns in reported speech.*
150. Femke Swarte (2016). *Predicting the mutual intelligibility of Germanic languages from linguistic and extra-linguistic factors.*
151. Sanne Kuijper (2016). *Communication abilities of children with ASD and ADHD. Production, comprehension, and cognitive mechanisms.*
152. Jelena Golubovi (2016). *Mutual intelligibility in the Slavic language area.*
153. Nynke van der Schaaf (2016). *Kijk eens wat ik kan! Sociale praktijken in de interactie tussen kinderen van 4 tot 8 jaar in de buitenschoolse opvang.*
154. Simon uster (2016). *Empirical studies on word representations.*

GRODIL
Secretary of the Department of General Linguistics
Postbus 716
9700 AS Groningen
The Netherlands