

University of Groningen

Complementary approaches to tree alignment

Kotzé, Gideon Jozua

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version

Publisher's PDF, also known as Version of record

Publication date:

2013

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Kotzé, G. J. (2013). *Complementary approaches to tree alignment: combining statistical and rule-based methods*. s.n.

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Complementary Approaches to Tree Alignment:
Combining Statistical and Rule-Based Methods

Gideon Kotzé



**university of
 groningen**

faculty of arts

CLCG

ñ | STEVIN

The work presented here was carried out under the auspices of the Dutch-Flemish NTU/STEVIN project PaCo-MT and the Center for Language and Cognition Groningen of the Faculty of Arts of the University of Groningen.



Groningen Dissertations in Linguistics 114

ISBN: 978-90-367-6177-2

ISBN (digital): 978-90-367-6178-9

© 2013, Gideon Kotzé

Cover image: *L'Allée des Alyscamps*, Vincent van Gogh, 1888

Printed by Off Page, Amsterdam, The Netherlands

RIJKSUNIVERSITEIT GRONINGEN

**Complementary Approaches
to Tree Alignment:**
Combining Statistical and Rule-Based Methods

Proefschrift

ter verkrijging van het doctoraat in de
Letteren
aan de Rijksuniversiteit Groningen
op gezag van de
Rector Magnificus, dr. E. Sterken,
in het openbaar te verdedigen op
maandag 24 juni 2013
om 11:00 uur

door

Gideon Jozua Kotzé
geboren op 9 juni 1981
te Empangeni, Zuid-Afrika

Promotores: Prof. dr. G.J.M. van Noord
Prof. dr. ir. J. Nerbonne

Beoordelingscommissie: Prof. dr. E.J. Krahmer
Prof. dr. F. Van Eynde
Prof. dr. J. Van Genabith

ISBN (electronic version): 978-90-367-6178-9
ISBN (print version): 978-90-367-6177-2

Contents

Acknowledgments	ix
List of abbreviations	xiii
1 Introduction	1
2 Background and related work	11
2.1 Introduction	11
2.2 Parallel corpora and treebanks	11
2.3 Tree alignment	25
2.4 Conclusion	34
3 The PaCo-MT project	35
3.1 Introduction	35
3.2 Background	35
3.3 Parallel treebank creation	37
3.4 Syntactic analysis	46
3.5 The transduction grammar	47
3.6 The transduction process	50
3.7 Generation	53
3.8 Conclusion	54
4 Statistical tree-to-tree alignment with Lingua-Align	55

4.1	Introduction	55
4.2	Background	56
4.3	Discriminative tree alignment	56
4.4	Alignment features	60
4.5	Parameters	67
4.6	First experiments	69
4.7	Experiments for PaCo-MT	72
4.8	Error analysis	75
4.9	Conclusion	80
5	A statistical study on the impact of different features on tree alignment performance in Lingua-Align	81
5.1	Introduction	81
5.2	Approach	82
5.3	Presentation and discussion of statistical data and analysis . . .	83
5.4	Conclusion and future work	90
6	Experiments in rule-based alignment error correction using manually constructed rules	93
6.1	Introduction	93
6.2	Simple rules	94
6.3	Bottom-up tree alignment error correction	106
6.4	Conclusion	112
7	Transformation-based tree alignment and alignment error correction	113
7.1	Motivation	113
7.2	Transformation-based learning	114
7.3	Transformation-based learning for tree alignment	116
7.4	Features	117
7.5	Implementation	124
7.6	Feature selection	132
7.7	Experiments	135
7.8	Results	139
7.9	Error analysis	149
7.10	Future work and conclusions	154
8	Translation quality evaluation based on alignment output	157
8.1	Introduction	157
8.2	Translation quality evaluation	158

8.3	Qualitative analysis	170
8.4	Conclusion	173
9	Discussion and conclusion	177
9.1	Summary of the results	177
9.2	Research questions	178
9.3	Contribution	182
9.4	Scientific significance	185
9.5	Future work and conclusion	185
	Bibliography	187
	Samenvatting	205

Acknowledgments

Just before I started my PhD, all I knew that was I was pretty interested in the idea of comparing languages, machine translation and creating language resources. When I saw this position being advertised, I knew I had to give it my best shot. Although I was very nervous during the online interview, it somehow resulted in an invitation for a personal interview - the rest is history. For that I have to thank those responsible in the Dutch Language Union for sponsoring the research, as well as John Nerbonne, Gertjan van Noord and Jörg Tiedemann, who all had a say in my eventual appointment.

For most of these last four and a bit years, I felt that I was very fortunate to have been at the Rijksuniversiteit Groningen. At the beginning, I felt very overwhelmed with all the work to be done, the literature to be read, the theory to be learned. In this regard, I must thank Jörg Tiedemann, alignment expert extraordinaire, from the bottom of my heart. He was my supervisor for the first year only, but he planted the seed of curiosity for the topic that I was about to explore, and did an absolutely stellar job in keeping me on track, even though I was severely lacking in skills and knowledge.

From my second year onwards, Gertjan van Noord took over. Albeit in a different way, his influence on what I managed to achieve was also profound. I was often stressed and anxious, he was always at least as cool as a cucumber, with a great sense of humour to boot. In the face of problems (such as deadlines) his solutions were always practical and workable. With him around, I always felt that things were in control and that there is no need to worry. I must

especially thank him for coming up with the idea of using the transformation-based learning algorithm, which turned out to work amazingly well.

John Nerbonne, co-promotor with Gertjan, played a crucial role throughout, but especially towards the end. Coming from a different perspective, his feedback was invaluable. His criticism was straightforward and to the point, and his expert contributions to the statistical sections of my thesis were also extremely valuable.

Another person who proved to be of much help in these sections is Çağrı Çöltekin, who received his PhD in 2011 and who I (and probably others) look up to as the local *R* expert. (During our two week attendance of the ESSLLI summer school in Copenhagen in 2011, he was probably the best room mate I could have asked for.) I should also thank Danie Venter from the Nelson Mandela Metropolitan University in Port Elizabeth, where I have received my Bachelor's and Honours degrees in Linguistics and Translation Theory, for providing me with sound expert advice in these matters.

This thesis was written in LaTeX, which is a kind of code which is automatically rendered to very nice, readable text. For that, you often need to do a lot of preparatory work such as specifying fonts, importing packages and specifying the structure. Most of this hard work was done for me by Tim van de Cruys, who kindly agreed that I use the code from his thesis (written in 2010) as a template.

During the PaCo-MT project, which lasted for no less than three years, my correspondence and occasional personal acquaintance with Dr Vincent Vandeghinste was of tremendous value to my professional growth. He was the key person behind the design of the system, without whom there would never have been a project. Scott Martens, who received his PhD in 2011, was another really hard-working and amicable member of the team. I had the pleasure of attending the annual Machine Translation Marathon in Dublin with Joachim van den Bogaert, who also worked on the project for a while.

I must also thank Prof Dr Frank van Eynde, who led the project. He was also on my reading committee, together with Prof Dr J. Van Genabith and Prof Dr E.J. Krahmer. I would like to thank all of them for devoting so much of their time to read and evaluate my work.

Wyke van der Meer, co-ordinator of the CLCG and one of the friendliest people I met, was enormously helpful throughout my stay in Groningen.

Even though the work itself had its fair share of stress, working at the CLCG was always a pleasure. This was thanks to my colleagues with whom I had the pleasure of not only sharing a workspace, but also got to know outside of work. If I had to name all the ways they enriched my lives, I would need several more

pages; therefore, I will keep it short. All of these CLCG people had a positive influence on my experience as a PhD student at the RuG:

Barbara Plank, brilliant and always smiling, with whom I taught a first-year course; Çağrı, whom I mentioned above - also an impressive soccer player, together with Jörg; Daniël de Kok, who also recently obtained his PhD and has helped me on countless occasions; Dörte de Kok (special thanks for helping out with planning the defence), Erik Tjong Kim Sang, Esteve Valls i Alecha (whom I visited in Barcelona for what turned out to be one incredible holiday), Fridah Katushemererwe, George Welling, Gisi Cannizzaro, Gosse Bouma, Harm Brouwer, Henny Klein, Jelena Prokić, Johan Bos, who helped me move my belongings; Karin Beijering, Kilian Evang (paranymph), Kostadin Cholakov (the most entertaining office mate ever, especially with Harm around), Leonie Bosveld, Ma Jianqiang, Marjoleine Sloos, Martijn Wieling, who helped me especially with printing-related matters; Noortje Venhuizen, Peter Meindertsma, Peter Nabende (office mate for about three years and regular online chess opponent), Proscovia Olango, Ruggero Montalto, Ryan Taylor, Simon Šuster, Tim van de Cruys, Tim Kallenborn, Valerio Basile (paranymph) and Veerle Baaijen.

I have thoroughly enjoyed not only conversing during working hours, but also playing soccer, climbing, biking, drinking, eating, partying and traveling with you. Also, I have almost certainly left out a few names - I hope you accept my apologies!

Of course, my paranymphs deserve special mention. Valerio: entertaining quirks and humour, extremely approachable, and a wide variety of interests; Kilian: friendly, hard-working and mysterious. Both have been extremely helpful and supportive. Best paranymphs ever!

Outside of the RuG, of course, there were also a few people whose presence and support I appreciate very much:

Santanu Roy: tennis, gym, pool, drinking buddy and physicist; Sergey Artyukhin (more or less the same), Tristan Kohl, fellow South African and family man, also a devoted gym-goer; Froukje Sijtsma, passionate about all things Frisian and curious about all things Afrikaans; Andreas and Jessica Lundberg: the perfect example of living a beautifully balanced life, with responsibility, intelligence, humour, very good food and passion for the outdoors and good health; Mariusz Szymanski: cardiologist, photographer, proudly Polish and always a pleasure to have around; Mahir Faik Karaaba: the Turk with whom I spoke more Dutch and German than English; and Liesbeth van Beusekom, who was my landlady for the last 1,5 years in Groningen, who took me to doctors, called taxis, helped me move, and just generally being the kindest grandmotherliest lady on the planet.

I have had the privilege of sharing much of my last few years with a very special person, Mariya Koleva, whose love and support enriched my life immensely and made it just that much easier for me to cross the finishing line. When I suffered an epileptic attack on a plane that caused three fractures in my spine, she went to the utmost trouble to ensure my physical comfort and care. Her mother Leda, who took me several times to a hospital for tests, care and consultation, and welcomed me with glowing hospitality, also fully deserves my thanks.

Back in South Africa, I have friends, family and teachers, too numerous to mention, who touched my life in one way or another. You know who you are - thank you. Throughout my life, I have also met people from various countries over all the world, some of whom became friends as well. I will not forget you.

Finally, my family in South Africa deserves the most praise. My brother Louis, for his encouragement in everything I do and his support, and with whom I can enjoy life in a multitude of ways; and the love of my father Ernst and my mother Ansie, who are my greatest supporters by far.

List of abbreviations

- AER:** Alignment Error Rate
- BLEU:** Bilingual Evaluation Understudy
- BNC:** British National Corpus
- BTK:** Bilingual Tree Kernel
- CFG:** Context-Free Grammar
- DGT:** Directorate-General for Translation
- DOP:** Data-Oriented Parsing
- DOT:** Data-Oriented Translation
- EBMT:** Example-Based Machine Translation
- EMA:** European Medicines Agency
- Europarl:** Proceedings of the European Parliament
- HMM:** Hidden Markov Models
- JRC:** Joint Research Centre
- LF:** Logical Form (Structure)

LFG: Lexical-Functional Grammar
Megam: MEGA Model Optimization Package
MT: Machine Translation
NIST: National Institute of Standards and Technology
NLP: Natural Language Processing
OPUS: Open Parallel corpUS
PaCo-MT: Parse and Corpus-Based Machine Translation
PBSMT: Phrase-Based Statistical Machine Translation
PFA: Prime Factorization and Alignments (algorithm)
POS: Part-Of-Speech
RBMT: Rule-Based Machine Translation
SCFG: Synchronous Context-Free Grammar
SEARN: Search and Learn
SMT: Statistical Machine Translation
STID: Stochastic Inversion Transduction Grammar
STSG: Synchronous Tree Substitution Grammar
SVM: Support Vector Machines
TBL: Transformation-Based Learning
TER: Translation Edit/Error Rate
TM: Translation Memory
XML: Extensible Markup Language

CHAPTER 1

Introduction

Listen to the trees as they sway in
the wind. Their leaves are telling
secrets...And their roots give
names to all things.

The Perpetual Calendar of Inspiration
VERA NAZARIAN

One of the most efficient ways to learn a new language is to know how the words in this language are translated into your mother tongue. The traditional approach to this is to look up an unknown word in a dictionary. With some luck, the dictionary might provide several contextual examples of the foreign word in use, extracted from real texts, i.e. not made up. With even more luck, the dictionary might provide examples of typical collocations (i.e. words with which they often co-occur), and warn the reader against common mistakes.

Although the expert knowledge underlying the construction of a dictionary is undeniably useful, the fact remains that, as the English linguist John Rupert Firth famously stated, “You shall know a word by the company it keeps”. The context-dependent nature of linguistic items (be it morphological units, words, phrases or sentences) has led to the development of a great variety of lexicographic resources, each with a different structure. Most of these resources are now available in some electronic format, something which provides many advantages over the paper-based versions, such as virtually unlimited space. It provided to be an enormous advantage to the field to the field of *corpus*

linguistics, where large collections of texts called *corpora* are studied in order to understand the behaviour of words and text in context.

Both the dedicated language learner and the professional linguist seek to better understand the structure of human language and how it relates to its context of use. Learning about the similarities and differences between two different languages on various levels is a fascinating experience. A corpus goes a long way in providing contextual information of linguistic items. However, a *parallel corpus* - consisting of a selection of *bitexts* - provides not only the same information for two or more languages, but allows the translations of the texts to be compared to each other.

The increasing availability and growth of parallel corpora on the Internet have led to a significant and very interesting application: statistical machine translation (MT) systems. MT has a long and fascinating history, but it is only relatively recently that we have acquired the means to model languages in parallel for use in the practical application of MT systems.

Depending on the type of domain, it may or may not be possible for an MT system to translate a given text sufficiently well on its own. However, MT is seldom used as a tool for high-quality unsupervised generation of the content of a language in another, but much more often as a way to facilitate the transfer if the end result merely needs to be sufficient for mutual understanding.

One major economic application is for the professional translator, who would benefit from any possible means which could help him or her produce output of a sufficient quality in as little time as possible. In this case, MT is not merely a facilitator, but a crucial means to increase the productivity of experts. Translation is big business today, and there is no shortage of work, so that MT can be regarded as of crucial importance for information technology. Of course, this economic benefit may also extend to people who write for a living, such as journalists.

MT also has many other practical applications, such as assisting the traveler in a foreign country or enabling one to read foreign news websites. Probably the best known example today is the Google Translate system, which can almost instantly produce translations for a variety of language pairs. Although the output quality may vary greatly, there is no denying the usefulness of this service for millions of Internet users all around the world.

Besides its practical use, MT and the theory surrounding it are also of great interest to the academic world. There is much research into how a language should be transferred to and represented in another. How should the languages be represented? Should we use syntax? How are they to be aligned? Which machine learning techniques can we apply? What are the most effective ways

to learn and represent knowledge? What can we learn about the structure and use of the languages involved, and the interplay between the language pairs involved? What are the typical errors? The list goes on.

For both the traditional linguist attempting to understand the structure and behaviour of language as well as the computational linguist who wishes to model her languages of choice as precisely as possible for practical use, a *treebank* is generally considered a rich source of information for various studies and applications. Here, the syntactic structure of sentences is described in detail according to any of a wide variety of different *grammars*. Not surprisingly, a *parallel treebank* is to a treebank as a parallel corpus is to a corpus. This introduces an extra dimension to the study and modeling of bitexts for use in natural language processing (NLP).

It is generally agreed that parallel treebanks are a very useful resource for all kinds of tasks. However, producing them manually is very expensive and time-consuming. For use in NLP applications, the required scale is typically in the range of hundreds of thousands or even millions of sentence pairs, which is practically infeasible, unless they are automatically produced.

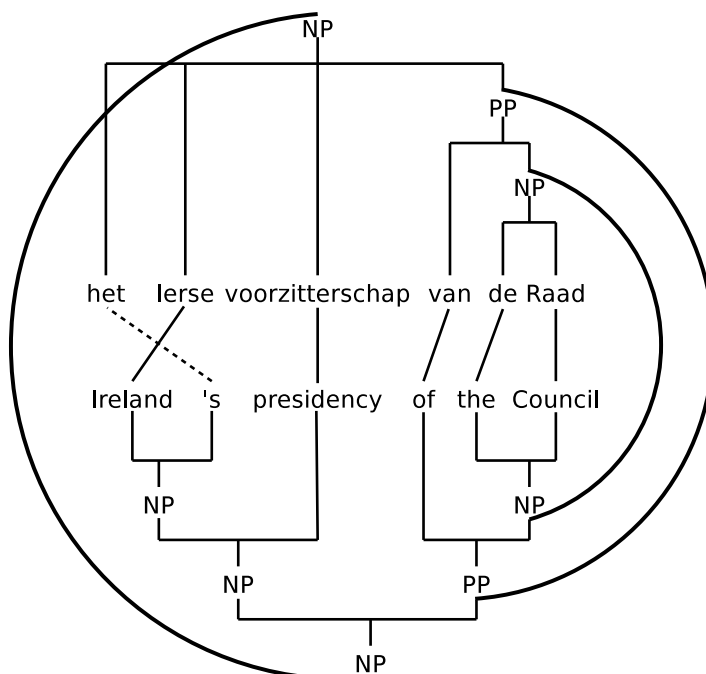
For this, we have high-quality *parsers* that can produce accurate syntactic structures for specific languages. Automatic syntactic analysis remains an ongoing field of study. A parallel treebank also needs to be *aligned* automatically, in order to actually extract the translational units. Typically, the words are aligned using specialized software that itself is trained on a parallel corpus of millions of words.

A relatively recent development is the high-quality automatic alignment of *constituents*, i.e. linguistically motivated phrases of a language. Phrase structures are represented as hierarchical structures in the form of trees, where the non-terminals refer to the constituents, dominating the phrases that are to be aligned. Figure 1.1 is an example. Recent interest in incorporating syntactic structure to MT models has led to more research devoted to this area.

The context of this work is a novel syntax-based MT system called *Parse and Corpus-Based Machine Translation (PaCo-MT)*. Its models are trained on large parallel treebanks aligned on the word and constituent levels. Our role in the project was the development and successful application of high-quality tree alignment software producing such parallel treebanks.

Historically, tree-to-tree alignment suffers from a lack of recall. This means that too few constituents are aligned. The reason for this is that they are restricted to the linguistically informed constituents that appear in the trees, and then only those that happen to be equivalent on both sides. In practice, it often occurs that monolingual parsers produce trees where many nodes do not

Figure 1.1: An example of the alignment of the words and constituents of a phrase pair. Automatic alignment software, discussed in later chapters, may assign varying amounts of confidence to links, based on previously calculated probabilities. Here, solid lines suggest a high confidence, while the dashed line is seen as less confident.



dominate the same strings as in the trees of the opposing side. For example, one parser may have a preference to embed phrases, creating deep trees, whereas another may choose to link more constituents directly to the root node, creating more shallow trees. This makes it impossible for many higher-up nodes to be linked, contributing to the recall problem. Another key problem is the existence of loose translations, as well as idiomatic or idiosyncratic expressions, which leads to structural deviations between the trees as well.

Statistical MT systems operating on just the raw text do not have these restrictions, therefore they usually manage to find more instances of equivalent phrases in comparison with the tree-to-tree approach. On the other hand, these

systems do not have access to the same pre-existing structural and labeled information as their tree-based counterparts.

One argument for the use of parallel treebanks in MT is increased transparency. Modeling for pure statistics-based MT is hard to improve, apart from using more data, more tuning and experimenting with more refined formulae. Many errors in alignment for syntax-based MT can be traced back to specific syntactic problems such as structural divergences which can, for example, be addressed by measures to improve tree alignment.

Finally, the production of parallel treebanks has intrinsic value. Whatever the practical effect on NLP applications, it also adds to the pool of scientific resources available for a given language pair.

In this work, our focus is on improving constituent (tree-to-tree) alignment in the process of parallel treebank creation. We leave the more detailed discussions for and against this approach to chapter 2. This brings us to our first research question:

Can we improve constituent alignment?

Given the context of our research, answering this question also entails attempting to improve the proportion of aligned nodes. We show in this work how we implement various measures in our attempt to answer this question. To increase accuracy, we implement different machine learning approaches, build our own training data and experiment using various alignment strategies, features and parameters. To increase coverage, we experiment with relaxing certain constraints, applying the union of directional alignment approaches, and combining different alignment strategies.

One issue that often comes to the fore with the automatic construction of parallel treebanks is the so-called pipeline problem. Using automatic tools invariably produces some errors. Using a combination of tools where each one processes the output of another may lead to a cascade of errors, widening the gap and increasing the negative effect on performance of the final product.

This and other issues which may affect the quality of constituent alignment lead us to the second research question:

Can we determine which factors influence constituent alignment? If so, can we use this information to improve our alignment approach?

To answer this question, we conduct various qualitative and quantitative analyses to study the effect of various features on alignment quality. We show how alignment improves as we apply what we learn from these studies.

Finally, we are also interested in the impact of our research on the quality of machine translation. Specifically, we measure the effect of various different alignment approaches to the performance of the syntax-based system PaCo-MT (chapter 3). This brings us to the final research question:

How does the quality of tree alignment influence the performance of syntax-based machine translation?

We show that PaCo-MT models trained on parallel treebanks produced by high-recall alignment models significantly outperform a model trained on parallel treebanks produced by a high-precision model.

Our research questions demand a flexible framework where constituents can be aligned according to a wide variety of needs and constraints. This includes:

- the ability to choose a high-precision or a high-recall alignment approach, or something in between
- the ability to choose different alignment strategies and apply various constraints
- the ability to choose from a wide variety and combination of features and parameters
- the ability to experiment with different learning approaches as well as heuristics
- the ability to combine different alignment approaches
- a common platform facilitating the interaction of all alignment software

In this work, we take the opportunity to develop the tools necessary in our attempt to answer these research questions. For this, we investigate two general approaches to tree alignment: stochastic and rule-based.

We show that stochastic tree alignment can be very flexible in terms of the options provided for feature and parameter selection, as well as alignment strategies. The system is able to produce accurate alignments, using a relatively small training data set that can be manually constructed in a day or two.

In our experiments, rule-based tree alignment also produces very accurate alignments. Manual rules are shown to increase both recall and F-score significantly, while a transformation-based learning system leads to both high recall and precision. Moreover, the error analysis procedure for the system is very transparent, since it can be shown which rules led to which good or bad alignments.

Both approaches have advantages and disadvantages. We will discuss this, as well as the extent to which our questions can be answered, in the conclusion.

Our language pair of choice is Dutch to English, even though three other language pair directions were available from the PaCo-MT project: English to Dutch, Dutch to French, and French to Dutch. The main reason for our emphasis on Dutch to English is that our French data produced relatively inferior results, perhaps in part because of lesser quality French trees produced by the parser. We felt that it was important to show that we can produce high-quality results as a proof of concept for our methods and algorithms. This is important as a baseline for future work on other language pairs.

In addition, there is a practical reason for selecting Dutch to English above English to Dutch, namely that our first training data set for PaCo-MT was applied to this direction. It is assumed that very little work would be necessary to adapt our methods to English to Dutch, hence we have not researched this language pair to the same degree. We base this assumption on our findings for the alignment experiments already made for this language pair.

It may be said that the restriction to one language pair does not show the ability to generalize. However, we believe that the flexible framework for alignment presented in this work allows for just that, and that with some effort, we would also be able to obtain a high degree of alignment accuracy with Dutch to French and vice versa.

During the course of the PaCo-MT project and in the writing of this thesis, we have manually developed parallel treebanks consisting of over a thousand sentence pairs. Word alignments and syntax trees were produced automatically, but all constituent alignments were created or checked by hand. However, we have also produced versions of the Dutch-to-English and English-to-Dutch alignment sets used in PaCo-MT where the word alignments were corrected. Our manually produced parallel treebanks include the following:

- Dutch to English: 940 sentence pairs
- English to Dutch: 150 sentence pairs
- Dutch to French: 158 sentence pairs

- French to Dutch: 158 sentence pairs¹

Most of the text is taken from the proceedings of the European Parliament,² but the Dutch-to-English parallel corpus also contains the following text:

- 115 sentence pairs from the DGT (Directorate-General for Translation) translation memories of the JRC (Joint Research Centre) Acquis³
- 335 sentence pairs from the OPUS (Open Parallel corpUS) free online parallel corpus,⁴ of which 220 sentence pairs are from OpenSubtitles (movie subtitles), and 115 comprise documentation from EMEA (the European Medicines Agency)

Reference to chapters with relevant papers

- In chapter 2, we present the background including some relevant concepts and academic work done in the field, focusing on parallel treebank construction and tree alignment.
- In chapter 3, we present the PaCo-MT project. We describe the context within which the system was conceived and designed and its general architecture. Next, we describe the procedures we followed in the creation of the parallel treebanks for the PaCo-MT system. This is followed by a description of the transduction and target language generation processes. Much of this work is based on the contents of Vandeghinste et al. (2013) which provides a detailed description of the workings of the system, as well as Kotzé et al. (2012) which describes the creation and application of our parallel treebanks to the system.
- Chapter 4 presents Lingua-Align, a statistical tree-to-tree aligner implementing the maximum entropy machine learning method with the purpose of constructing large parallel treebanks, especially for syntax-based MT systems such as PaCo-MT. We discuss its general framework of discriminative modeling and present the range of features, parameters and alignment strategies that can be applied. We run experiments using a

¹The actual textual content is the same as for Dutch to French, but alignments differ in some respects due to the direction chosen.

²<http://www.statmt.org/europarl>

³<http://langtech.jrc.it/DGT-TM.html>

⁴<http://opus.lingfil.uu.se>

small trilingual parallel treebank, as well as using data from the PaCo-MT project. Finally, we conduct a qualitative error analysis. This work is based on the contents of Tiedemann and Kotzé (2009a), which describes the discriminative approach, and Tiedemann and Kotzé (2009b), which extends the work in the previous paper to the creation of parallel treebanks for use in the PaCo-MT system. Some examples in the chapter are based on similar examples in Tiedemann (2010), in which the Lingua-Align toolbox is described in more detail.

- In chapter 5, we present a statistical study which shows the correlation and statistical significance of various features for the performance of Lingua-Align. The work presented here is based on Kotzé (2011a).
- In chapter 6, we present our experiments in using manual rule-based heuristics in order to increase the alignment recall of a high-precision Lingua-Align model. The contents are based on the work presented in Kotzé (2011b) and Kotzé (2011c). The bottom-up rule-based heuristic presented in the chapter is reported in Kotzé et al. (2012).
- We present our rule-based tree aligner in chapter 7. It implements the transformation-based learning method. We discuss how we adapted the method for use in tree alignment, the features and parameters that can be applied, experiments, results and a qualitative error analysis. This chapter was adapted for Kotzé (2012).
- In chapter 8, we discuss the results of an extrinsic evaluation of the application of our parallel treebanks to the PaCo-MT system. We compare the results to a state-of-the-art phrase-based statistical MT system, Moses. We perform a statistical significance test on the results, after which we discuss a selection of output sentences from the test set. Some of the extrinsic evaluation results are reported in Vandeghinste et al. (2013) and Kotzé et al. (2012).
- In chapter 9, we discuss the results with reference to our research questions. We also discuss the significance of the research, propose future work and arrive at a number of conclusions.

Background and related work

2.1 Introduction

In the introductory chapter, we briefly touched on the concept of a parallel treebank as an important resource for linguistic study and various natural language processing tasks. Tree alignment is an important process in the creation of such a resource. Briefly speaking, it refers to the automatic alignment of the nodes of the trees that describe the syntactic structure of translationally equivalent sentences. Such alignments implicate equivalence of phrases which is a useful resource for tasks such as cross-linguistic studies and, of special importance for this work, machine translation (MT).

But where does all this come from? How do we get these parallel sentences? How do we make the trees? How do we align? How do we use the output? And why is this important? To answer this, we need to provide some context in which we also need to define some crucial concepts that will be used throughout this work.

2.2 Parallel corpora and treebanks

As mentioned in the previous chapter, language practitioners frequently have need of large collections of empirical data. The assumption is often made that more is better, since more data tends to be more representative than a smaller sample. Of course, a large sample can also be quite skewed in terms of its

representativeness. Therefore, care is often taken to construct such collections so that they include many different kinds of data, such as different genres.

The corpus

The data type on which we will focus for the entirety of this thesis is written text in the form of a *corpus*. Ideally, a corpus is a balanced selection of texts which can be seen as somewhat representative of a language or of the domain one wishes to study. This is especially important if the goal is to apply statistical tests for later generalizations and conclusions. For any kind of processing it is of course required for a corpus to be machine-readable.

However, from the point of view of the computational linguist, this goal may be somewhat different. Often, one simply wishes to train one's software to be as accurate as possible when applied to new data. This is often domain dependent, but not always, or at least not to the same degree than for other tasks. On the one hand, there are shallow tasks such as part-of-speech tagging which, although it may be affected by a change of domains, are relatively easy to implement accurately. On the other hand, there are much harder tasks such as machine translation, the accuracy of which is very dependent on the domain in which it operates.

Between linguistic disciplines, there is sometimes debate over the exact meaning of "corpus". In this work, we will, for practical reasons, simply call any collection of texts which are potentially useful to be applied for natural language processing (NLP) tasks a corpus.

Well-known examples for English are the Brown corpus family (Kučera and Francis, 1967), the British National Corpus (Burnard, 2000), the Corpus of Contemporary American English (Marck, 2008-), The American National Corpus (Reppen et al., 2005) and the International Corpus of English (Greenbaum and Nelson, 1996).

The parallel corpus

Simply put, parallel corpora are corpora for two or more languages where each one is a translation of the other. Just like a corpus is a collection of texts, a parallel corpus is a collection of *bitexts* (Tiedemann, 2011). Tiedemann (2011, p. 1) cites Harris (1988) in defining a bitext as a document with one or more translations in other languages. Conventionally, one side of a bitext or parallel corpus is often called the *source side* and the other side is called the *target side*, even though directionality is not necessarily implied.

Of course, such data can be very useful for any task having to do with translation: studying the agreements and differences between two languages, extracting various statistics, constructing translation memories and dictionaries via alignment processes for aid in manual translation or lexicographic work, word-sense disambiguation tasks, and so on. Parallel corpora are also successfully used as training data for the construction of statistical MT (SMT) systems such as Moses (Koehn et al., 2007), Hiero (Chiang, 2007) and Joshua (Li et al., 2009; Weese et al., 2011).

Examples of well-known large parallel corpora are the JRC-Acquis¹ Multilingual Parallel Corpus (22 languages) containing legislative texts on the European Union (Steinberger et al., 2006), the European Parliament Proceedings (Koehn, 2005) and the OPUS (Open Parallel corpUS) project where documents from various different domains are collected, aligned and annotated (Tiedemann, 2003).

The texts in a parallel corpus can be aligned on a sentential level or they may be more loosely related. It is not unusual for the same sentence to occur in a different part of a translated paragraph. For that reason, one may wish to automatically sentence align a parallel corpus to extract equivalent sentences.

Alignment of parallel corpora

The alignment of parallel corpora often proceeds in a kind of divide-and-conquer approach. Tiedemann (2011, p. 14) describes what is termed a *hierarchical iterative refinement strategy* where big chunks of texts, for example paragraphs or even sections, are first aligned, then smaller chunks, such as sentences, and finally, if required, the words or lexical segments. Proper segmentation as a pre-processing step is required. For example, sentence alignment can only be successful if the units to be aligned are actual sentences, instead of groups or parts of sentences. For each step, one often makes assumptions and induce some constraints in order to reduce the hypothesis space. This also reduces the required computing power and may also increase the alignment accuracy. Wu (2010, p. 373-377), also cited by Tiedemann (2010, p. 12-14), lists some common constraints that can be used in various alignment approaches:

- One may choose to align textual elements in a one-to-one fashion only. This is called the *bijection constraint*.

¹JRC stands for *Joint Research Centre*.

- In directional alignment approaches, the *functional constraint* dictates that every source element be aligned to exactly one target element. This is often applied in word alignment.
- According to the *monotonicity constraint*, all elements on both sides are in the same order. This means that links between elements never cross. This is actually an assumption commonly made by sentence alignment software (Tiedemann, 2010, p. 37).
- *Segmentation constraints* are enforced by the type of segmentation done before the alignment process. For example, if the segments are sentences, alignment is restricted to the sentential unit.
- *Segment size constraints* limit the number of segments to which a specific segment can align.
- *Anchors* are pairs of segments assumed to be coupled and taken as *hard constraints*, where “*anchor constraints* are bisegment constraints that can be thought of as confirmed positions within the matrix that represents the alignment candidate space” (Wu, 2010, p. 373).
- *Syntactic constraints* may also be imposed. For example, word alignments can be constrained by the underlying grammar formalism.
- One may also set *distortion limits* for aligned elements by restricting the maximum distance of these elements from each other between the positions in their respective bitext halves.

Sentence alignment

A sentence aligner might return two sentences on one side for a single sentence on the other side, or even three. This is a regular occurrence for loosely translated texts. For NLP applications, it is nonetheless often useful and pragmatic to restrict the hypothesis space to single sentences - in other words, applying the bijectivity constraint - as is often done for MT systems, although this makes the disambiguation of certain discourse elements more difficult. This is because these elements may refer to entities mentioned in other sentences. However, some research has already been done on the incorporation of so-called discourse units in MT (Ullah et al., 2009) or the disambiguation of discourse elements such as connectives which connect different parts of a discourse (words such

as *however*, *although* and *indeed*) or pronouns - for more accurate MT (Meyer et al., 2012; Meyer, 2011; Popescu-Belis et al., 2012).

Restricting sentence alignment to one-to-one occurrences is still very useful for accurate machine translation, and it dramatically reduces the hypothesis space for required procedures such as word alignment. For this reason, many MT systems make this sacrifice, with the apparent assumption that the disambiguation of discourse elements may be resolved in a separate step if necessary. All the data that we have used in our experiments rely, at some point, on one-to-one sentence alignments. We will therefore only focus on such sentence pairs for the rest of this work.

Word alignment

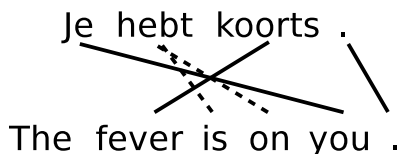
Word alignment is a crucial process in most forms of machine translation. It is also a quite difficult task because of the role that semantics is playing in the alignment process, something that is not directly visible from the data and can only be deduced by looking at various cues and statistics. Furthermore, specific languages may differ very much from each other in terms of their orthography, word order, the way they express certain concepts, nesting of phrases, use of devices such as ellipsis, discourse elements, and so on, all throwing their respective spanners in the works which contribute to making word alignment a very challenging task.

Figure 2.1 is an example of a short word-aligned sentence pair between Dutch and English. The Dutch sentence literally means *You have fever*. In practice, very few translations are exact replicas of each other. This is reflected in the fact that some alignments link words that are only partially or vaguely translatable, such as the ones between *hebt* (*have*) and *is on*, as well as the existence of the unaligned word *The*. It is in fact debatable whether *hebt* should be aligned to *is on* at all, although *is on* can be said to share a similar relationship to *The fever* and *you* as *hebt* has to *Je* (*You*) and *koorts* (*fever*).

This brings us to our next point, which is that different word alignment strategies exist depending on what the output will be used for. If the task is to construct a high quality lexical translation table, it makes sense to opt for a high-precision approach where only confident links are applied. For some stochastic processes such as those used in SMT, however, it may be more useful to adopt a high-recall approach where many alignments are made even though some of them are likely to be wrong.

Before word alignment, proper *tokenization* of lexical units is required. Phenomena such as the existence of abbreviations often makes deciding whether or

Figure 2.1: An example of word alignment between Dutch and English, where *Je hebt koorts* literally means *You have fever*. The solid lines denote confident links (fully translatable) and the fuzzy lines denote less confident links (partly translatable).



not a period is the end of a sentence a tricky proposition. In the context of word alignment, it is useful to choose a tokenization scheme for both languages that maximises the convergence of alignments between lexical items. For example, the English string *I have* can be contracted to *I've*. In tokenized form, it can be written as *I've*. The French equivalent is *J'ai* which can be tokenized as *J'ai*. If English text is to be aligned to French text, it would be useful to be able to align *I* to *J'* and *'ve* to *ai*, as variants of *I/Je* and *have/ai*. If, however, *J'ai* is tokenized as *J' ai*, this may affect the quality of MT, since now, if we assume that *have* is aligned to both *'* and *ai*, we have two competing translations for *have* of which only one is actually correct. This is also the case if *I* is aligned to both *J* and *'*, but here, even aligning *I* to *J* can be perceived as incorrect if the apostrophe is not present, as *J* may not be perceived as an acceptable translation.

It is often assumed that words - in the sense of tokenized strings separated by whitespace - represent distinct units of meaning, but the existence of multi-word units, especially of the idiomatic variation, shows that it is not always the case. Function words can also denote relations between concepts where the same can be represented by morphological units in another language.

For these reasons, conventional tokenization is not always a sufficient method to segment lexical items. They may also be grouped together as single meaningful units, as is sometimes done with Chinese, or perhaps one may choose to write morphemes as separate units.

Segmentation can also be applied on other levels. For example, one may choose to split bitexts into sentence pairs as the lowest meaningful units, as is the case in sentence alignment.

The best word alignment software today employs statistical models. There are two common approaches, generative and discriminative modeling. The generative approach attempts to, given some hidden parameters, generate

the observable data, while the discriminative approach constructs a model of making translation decisions based on the observable data. More formally, generative models learn the joint probability distribution $p(x, y)$, where x and y are the observation and label sequences. Discriminative models learn the conditional probability distribution $p(y|x)$, where y is the unobserved variable whose dependence on the observed variable x we are modeling.

While somewhat outdated, Och and Ney (2003) provide an informative and detailed overview of the word alignment techniques used in classical SMT. They propose a new method (the so-called Model 6) which performs better than any other discussed in the paper.

Generally, there is a distinction between heuristic and statistical models. The authors find that statistical models perform better. One of the inspected methods implements a machine learning method called Hidden Markov Models (abbreviated to HMM, see, for example, Jurafsky and Martin (2009), chapter 6) which is a generative modeling approach. Och and Ney combine HMM and Model 4, which is a model where the assumption is made that “every word is dependent on the previous[ly] aligned word and on the word classes of the surrounding words” (Och and Ney, 2003, p. 27), in a directional alignment approach. Additionally, they implement heuristic refinement methods, applying the intersection of the source-to-target and target-to-source alignments, then iteratively adding neighbouring alignments based on certain previously defined conditions.

The directional approach is implemented in a system called GIZA++ (Och and Ney, 2003) and the directional heuristics can be applied using the Moses SMT system (Koehn et al., 2007). We have used these tools for word alignment in the PaCO-MT project described in the next chapter, as well as in our subsequent research. Although there are some newer systems that report both better accuracy and resulting MT evaluation scores (Hermjakob, 2009; Liu et al., 2010; Lambert and Banchs, 2012), GIZA++ still compares favourably and is still widely used for experiments reported in the field.

The treebank

A *treebank* is a corpus where each sentence has been annotated with some sort of syntactic structure. Since annotating it by hand is not only a time-consuming process but also a difficult task best left to the expert linguist, the general preference is to develop software - called *parsers* - for the accurate large-scale analysis of specific languages. Nevertheless, since manually constructed treebanks are such a precious resource for the training of this software, there

have been a number of efforts to create large manually crafted treebanks for various different languages.

The formal description of language is an ongoing process and has resulted in a range of different fields of study, each attempting in its own way to better understand the structure of the language it is trying to describe. There are many different grammars and grammar formalisms as a result, the scope of which is decidedly too large to discuss in this work. However, one can generally distinguish two main types of grammars: phrase structure and dependency grammars.

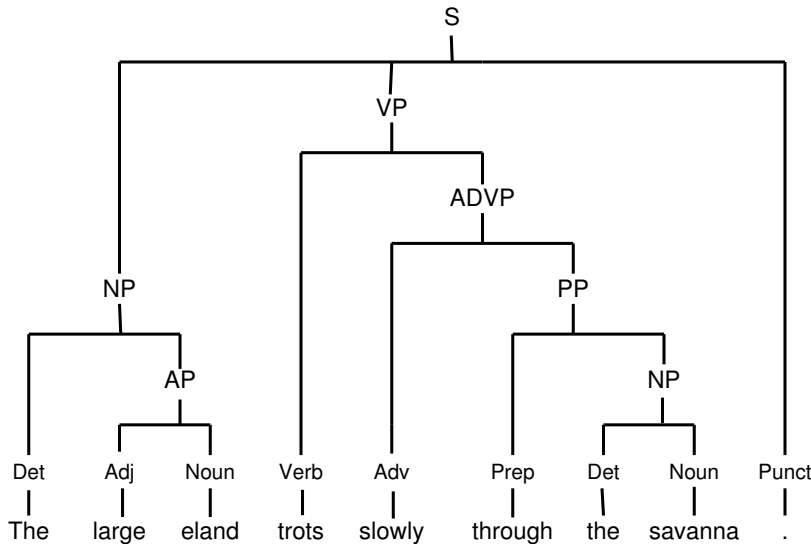
Phrase-structure grammars, chunking and predicate-argument structure

Phrase-structure grammars (synonym: *context-free grammars* (CFGs)) are based on the assumption that words in natural language tend to cluster together in groups called *constituents*. Examples are noun phrases (*the large eland*), verb phrases (*trots slowly through the savanna*) and prepositional phrases (*through the savanna*).

Such a grammar consists of a set of *rules* or *productions*, where on the left side there is the name of the constituent and on the right side one or more constituents or words that the constituent on the left side produces; as well as a *lexicon*, which is the set of all words or symbols that the grammar can produce. It often assumes a hierarchical structure, where constituents can produce other constituents which can themselves produce more constituents or symbols. In the example above, assuming that the words in the sentence comprise the whole lexicon, we could write down the following grammar:

```
S -> NP VP Punct
NP -> Det AP
NP -> Det Noun
VP -> Verb ADVP
ADVP -> Adv PP
AP -> Adj Noun
PP -> Prep NP
Det -> the
Adj -> large
Noun -> eland
Noun -> savanna
Verb -> trots
```

Figure 2.2: An example of a phrase-structure tree.



Adv -> slowly

Prep -> through

Punct -> .

This small grammar would of course also allow for nonsensical constructions such as *The large savanna trots through the eland*. Apart from semantics, a so-called *probabilistic context-free grammar* could, based on frequencies extracted from a treebank, show which productions and sentences occur more typically than others.

Using the above grammar, we can now produce a syntactic tree for our sentence (figure 2.2).

Note that each word has a label. These are the *part-of-speech tags*, such as *Adj* for adjective, *Adv* for adverb, *Det* for determiner, *Prep* for preposition and *Punct* for punctuation. These nodes are the terminal nodes or terminals, and those on higher levels dominating the terminals (the set of *category labels* {*S*, *VP*, *NP*, *AP*, *PP*}) are the non-terminal nodes or non-terminals.

There are various ways to represent a phrase-structure parse. Perhaps the most well-known is the simple bracketing representation which is used in the

Penn treebank. Brackets are naturally suited to displaying hierarchies of elements in a left-to-right order. The example sentence above would be displayed as follows:

(S (NP (Det The) (AP (Adj large) (Noun eland))) (VP (Verb trots) (ADVP (Adv slowly) (PP (Prep through) (NP (Det the) (Noun savanna))))) (Punct .))

The Stanford parser (Klein and Manning, 2003; Klein and Manning, 2003b) extends this approach by adding more metadata in the form of co-indexing and dependency information.

Another representation format that has grown in popularity is TIGER-XML (Mengel and Lezius, 2000; Lezius et al., 2002). This format is currently assumed by some of the software packages we are using in our research, such as the Stockholm TreeAligner.

Chunking, also known as *shallow parsing*, simplifies the problem by ignoring the internal structure but simply identifying non-overlapping phrasal segments. Since this is a much simpler task which can generally be achieved with more accuracy than full parsing, chunking is useful if that is all that is required for further processing.

This is a chunked version of the aforementioned example sentence:

[NP The large eland] [VP trots] [ADVP slowly] [PP through] [NP the savanna]

One may also describe a text with so-called *predicate-argument structure*. A *predicate* is the part of the sentence which modifies the subject. The *arguments* are those parts of a predicate which complete its meaning. One may also enrich an existing phrase or dependency structure (see below for a definition) with predicate-argument structure.

In an article on annotating the Penn Treebank with predicate-argument structure, authors Marcus et al. (1994) provide a few examples in bracketing style, such as:

I consider Kris a fool.

becomes

consider(I,fool(Kris))

and

What is Tim eating?

becomes

eat(Tim,what).

Dependency grammars

Syntactic dependency grammars rely on the assumption that there exist natural dependency relations between specific words in a language. In a sentence, some words can be seen as more important or more crucial to its understanding than others. A proper sentence usually requires the existence of a main verb which is called the root. The other words in the sentence are regarded as directly or indirectly dependent on the root. Between the words themselves, dependencies exist as well. Except for the root, each word has a single *head*, while a head may have one or more *dependents*.

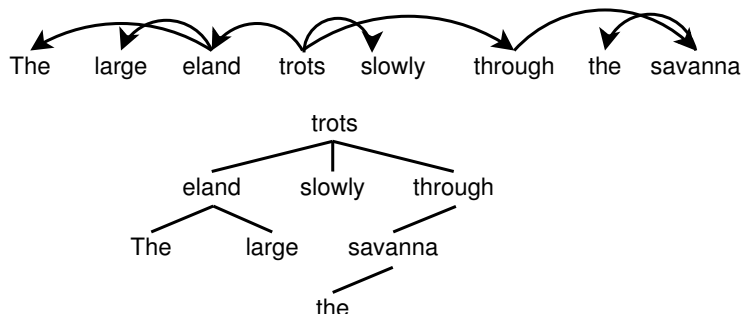
Figure 2.3 presents an example of two different representations of dependency structure, using the example sentence introduced before. In the first example, arrows are used which always point away from the heads towards their dependents. In the second example, heads are always directly above their dependents. Note that this resembles a phrase structure but with words instead of non-terminal constituents.

We will not proceed further into any formal theory, explanation of linguistic concepts or introduction to existing parsers. However, in chapter 3 we will introduce the parsers that we have used to create our parallel treebanks.

Examples of treebanks

A significant number of treebanks already exist, for both phrase-structure and dependency grammars. Some are completely manually produced, while others have been parsed and then corrected and checked for consistency by experts. Many of them cover a few different domains or textual styles, but some are more specialized or limited to a single text. Examples of more specialized treebanks include the Quranic Arabic Dependency Treebank (Dukes et al., 2010), the Penn

Figure 2.3: Two different examples of representing dependencies.



Parsed Corpora of Historical English (Kroch and Taylor, 2000; Kroch et al., 2004; Kroch et al., 2010) the Diachronic Corpus of Present-Day Spoken English (Aarts and Wallis, 2006) and the Ancient Greek Dependency Treebank (Bamman et al., 2009). Other well-known treebanks include the Penn Treebank (Marcus et al., 1993), the Prague Dependency Treebank (Hajič, 1998), the German TIGER Treebank (Dipper et al., 2003), the Tübingen Treebank of Written German (Telljohann et al., 2009) and the French Treebank (Abeillé et al., 2003).

All in all, many languages are covered, including at least the following: Arabic, Bulgarian, Catalan, Chinese, Croatian, Czech, Danish, Dutch, English, Estonian, Farsi, Finnish, French, German, Greek, Hebrew, Hindi, Hungarian, Icelandic, Italian, Japanese, Korean, Latin, Norwegian, Persian, Polish, Portuguese, Romanian, Russian, Slovene, Spanish, Swedish, Thai, Turkish, Urdu and Vietnamese.

The parallel treebank

We define a *parallel treebank* as a parallel corpus that is syntactically analysed - or *parsed* - on both sides, usually aligned in some way on a subsentential level.

Just like treebanks, parallel treebanks consist of more linguistic information and are especially useful for applications that are syntax-based. Just like parallel corpora, parallel treebanks can serve as a useful resource for translational studies and the development of MT systems and other NLP applications dependent on parallel data.

However, parallel treebanks also provide a syntactic and hierarchical structure to the parallel data. It now becomes possible to extract *parallel trees*: labeled

hierarchical structures of nested translational phrases. This is the domain of *tree alignment*, discussed in section 2.3.

A good overview of existing parallel treebanks is provided by Zhechev (2009, p. 9–15). We shortly summarize them here, including some more recent developments.

The SMULTRON parallel treebank provides parallel data between the languages of English, Swedish, German (Samuelsson and Volk, 2006) and, in the latest version at the time of writing (version 3), French and Spanish (Volk et al., 2010), comprising around 2500 sentences.

A project aiming to align Estonian to German sentences is described by Uiho et al. (2005). The phrase-structure trees are manually aligned using English as an intermediary language to obtain translations. However, only NP phrases are aligned.

The Prague Czech-English Dependency Treebank (PCEDT), containing a part of the Penn Treebank and its translation into Czech, is described by Cuřín et al. (2004) and Čmejrek et al. (2004). It is annotated with a type of deep syntax called *tectogrammatical annotation* which is not compatible with phrase-structure alignment as we describe it in this work. Since then, a major update has been released (Hajič et al., 2012) which also includes alignment on the word level.

Uchimoto et al. (2004) present a Japanese-English-Chinese parallel treebank consisting of news paper articles translated from the Penn Treebank and the Kyoto University text corpus. It is morphologically and syntactically annotated and also aligned on a phrasal level. Word alignment has been eschewed in favour of Japanese minimalistic linguistic units called *bunsetsus* which are aligned to their equivalents in Chinese and English.

The Linköping English-Swedish Parallel Treebank (LinES) is a parallel dependency treebank constructed in order to study translational variations of syntactic constructions from English to Swedish (Ahrenberg, 2007). Word alignments are automatically produced but manually reviewed.

The CroCo corpus is an English-German translation corpus in eight different registers for both languages and consisting of one million words (Hansen-Schirra et al., 2006). The texts are chunked instead of properly parsed. The main goal was the investigation of the phenomenon of *explicitation* which is a tendency that translated texts are made more explicit and clear than the originals.

The phenomenon of *translation shifting* occurs when the translation deviates from a formal correspondence with the source text. Studying this is the main purpose of the FuSe parallel corpus (Cyrus, 2006). Again, sentences are not

fully parsed, but annotated with predicate-argument structures which are then aligned.

An English-Swedish-Turkish parallel treebank of fiction and technical documents is described by Megyesi et al. (2010). It was created using the Uplug toolkit (Tiedemann, 2003b), contains about 300,000 tokens in Swedish, 160,000 in Turkish and 150,000 in English and is used for linguistic research and teaching.

The English-French HomeCentre parallel treebank (Hearne and Way, 2006) is a hand-crafted resource comprising 810 sentence pairs from a Xerox printer manual. Hearne and Way utilize it for Data-Oriented Translation (Poutsma, 2000) experiments. Zhechev (2009) also uses it in his thesis for training and testing his unsupervised tree aligner (described in the next section).

Along with HomeCentre, Zhechev (2009) also produces a parallel treebank from the English/German part of Europarl, the European parliament proceedings (Koehn, 2005), containing 10,000 sentence pairs, for his thesis experiments.

Rios et al. (2009) present a small Spanish/Quechua parallel treebank, consisting of the Universal Declaration of Human Rights (about 100 sentences) as well as “some information texts, and the FAQ from the website of the Peruvian Defensoría del Pueblo, which all together contain about 100 sentences” (Rios et al., 2009, p. 54). The Quechua language is strongly *agglutinative*, meaning that affixes tend to correspond to syntactic categories. Therefore, segmentation was done on the level of the morpheme. A morphological analyzer was developed to achieve this. The phrase structure nodes were manually aligned using the Stockholm TreeAligner (Lundborg et al., 2007), which we will describe in the next chapter.

The Turin University parallel dependency treebank is a small resource consisting of the Universal Declaration of Human Rights and some text from the JRC Acquis corpus (Steinberger et al., 2006) for the languages of Italian, English and French (Sanguinetti and Bosco, 2011). More recently, it “has been enlarged with an additional corpus extracted from the open licence ‘Creative Commons’ ” (Bosco et al., 2012, p. 1934) amounting to around 100 sentences. They implement two annotation formats, a native dependency format based on the *Word Grammar* theoretical framework (Hudson, 1984), as well as an enriched (in terms of the annotation of morphology and functional relations) constituent-based Penn treebank style format.

Göhring and Volk (2011) reports a project in which the parallel versions of the French and German yearbooks of the Swiss Alpine Club have been scanned and converted to an electronic parallel corpus, called Text+Berg, consisting of more than 4 million tokens. A selection of 1000 sentences were made from

mountaineering reports to build a parallel treebank. This was included in the latest version of the aforementioned SMULTRON parallel treebank.

Li et al. (2012) present two parallel treebanks, abbreviated to PAT (Parallel Aligned Treebanks) as part of an investigation on an optimal methodology for the large-scale creation of such resources under the direction of the Linguistic Data Consortium (LDC). One PAT is Arabic/English and the other Chinese/English. As with the Spanish/Quechua parallel treebank described above, segmentation is an issue because of the large differences between the languages involved. Additional measures are undertaken in this regard such as the additional splitting of some Arabic tokens and a three-level alignment approach between English and Chinese. The PATs consist of four different genres: newswire, broadcast news, broadcast conversation and weblogs. The data for each language pair consists of just under a million words or characters (for Chinese).

Colhon (2012) reports the construction an English-Romanian parallel treebank for eventual use in an MT system. Romanian trees are generated by using information obtained from English parse trees, word alignments and Romanian POS tags. Through this process, the trees are aligned. Because the Romanian trees are dependent on the English ones, they may be less representative of the language and certain constructions may not be well aligned. To deal with this, the authors propose additional pre- and post-processing of the data. The final product comprises two data sets, one consisting of 200 sentences and the other consisting of 1420 sentences.

Finally, Kotzé et al. (2012) present four large sets of automatically produced parallel treebanks for the language pairs of Dutch to English, English to Dutch, Dutch to French and French to Dutch. This was a major goal of the Parse and Corpus-Based Machine Translation project (PaCo-MT), where the parallel treebanks served as a source of training data for a syntax-based MT system for all four language pairs. It is in this environment that most of our research was done. The project will be presented in more detail in the next chapter.

2.3 Tree alignment

Definition

Tree alignment is a type of *structure alignment* between constituents of a tree pair in a bitext, such that the resulting output is a strictly compositional, hierarchical alignment (Wu, 2010, p. 391). This implies what Wu calls the *crossing constraint*, where no crossing links are allowed. Hearne et al. (2007) also call it the *well-*

formedness constraint, which is the term we will use for the rest of this work. They define it as follows:

- Descendants of a source-linked node may only be linked to descendants of its linked node on the target side and vice versa (target to source).
- Similarly, ancestors of a source-linked node may only be linked to ancestors of its linked node on the target side and vice versa (target to source).²

The authors also add another constraint, which is that nodes may only be linked once. We will discuss this in the next chapter.

To understand the logic of the well-formedness constraint, consider that non-terminal nodes dominate specific phrases. If the non-terminal nodes are aligned, it implies that the phrases are aligned. Let's call these non-terminals A and B. It would follow that any children of A and B dominate part of the phrases that A and B dominate respectively. Let's say C is a non-terminal child of A and D a non-terminal child of B. C now dominates a part of the phrase that A dominates and similar with D with respect to B. Since A and B are aligned, the phrases are considered equal. But unless C and D are aligned, the phrases that they dominate are not considered equal.

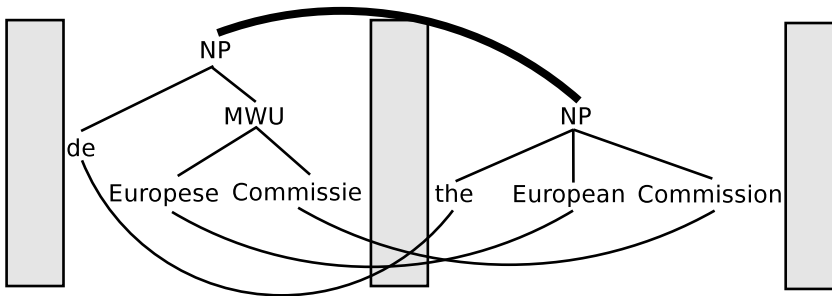
Now imagine that the child C is aligned to a node, G, that does not have B as an ancestor (direct or indirect parent), but F. F dominates a different phrase than B does. G, as a child of F, dominates a part of the phrase that F dominates. Since C is linked to G, it must logically follow that a part of the phrase that A, the parent of C, dominates, is the same as a part of the phrase that F dominates. However, this is not the case. As a result, either the alignment between A and B is wrong or the alignment between C and G is wrong.

The same argument can be applied to parents or ancestors that link to nodes outside the trees instead of children or descendants. To clarify this a bit more, figure 2.4 presents an example of an aligned Dutch/English subtree pair which is well-formed with respect to its alignments. The NP non-terminals are aligned, implying that *de Europese Commissie* is translationally equivalent to *the European Commission*, since those are the phrases that the NPs dominate.

A useful way to understand the concept of well-formedness is to draw an imaginary boundary around the trees, where all other subtrees are outside the lines. If any of the alignment lines ends up outside the boundary, the trees are not well-formed and the crossing constraint has been violated. The outside of

²One side is often called the source side and the other the target side, even if eventual translation from the former to the latter is not the end application one has in mind.

Figure 2.4: An example of a well-formed tree pair, aligning corresponding phrases in Dutch and English. An alignment ending up in a grey block would violate the well-formedness constraint.



these boundaries are represented by the grey blocks in the figure. Note that the lines may travel through the grey areas as long as they do not end up in them. In short, any alignments between two trees must stay between these two trees.

It may sometimes be useful to violate the well-formedness constraint, at least on the word level, such as with high-recall alignment approaches where a certain amount of error is tolerated. We describe our experiments with this in chapters 6 and 7.

Related work in tree alignment

Some of the earliest work in the field has been in the context of example-based machine translation (EBMT). In Nagao's approach (Nagao, 1984), syntactic patterns are found via analogy from which corresponding word and phrase lists can be constructed.

Kaji et al. (1992) present a language-independent system that extracts translation templates for EBMT systems, tested on English-Japanese, using a greedy algorithm. A bilingual dictionary is used for word alignment and solutions are offered to resolve certain kinds of ambiguities. However, no actual trees are produced since the purpose of the system is the extraction of the aforementioned templates.

Matsumoto et al. (1993) also work on English-Japanese. The trees are produced by Lexical-Functional Grammars (LFG) and then converted to dependency structures. Clusters of connected tokens - subgraphs - in the dependency structure called *decompositions* are aligned to their equivalents in the other language.

Meyers et al. (1998) present a system, based on work by Grishman (1994) and Meyers et al. (1996), aligning nodes in so-called *regularized parses*, a type of predicate-argument structure. No actual output in the form of parallel treebanks is produced, but instead, the alignments are directly used as transfer rules for a rule-based MT (RBMT) system.

Instead of utilizing monolingual parsers, Wu (2000) implements *stochastic inversion transduction grammars*, or STIDs (Wu, 1997) where a common structure for both sides is obtained. Phrase-structure like trees are then constructed, where the non-terminals are category labels and the leaves are pairs of corresponding words or phrases for both sides. One problem with this approach is that it assumes the same syntactic structure for corresponding sentences in both languages. In reality, sentences may be more loosely translated and segments may not always correspond precisely. This is also suggested by Wellington et al. (2006) who report experiments showing that bitexts exhibit translational patterns that are more complex than reported in the literature. Based on this, they suggest that STIDs do not have the ability to generate some of these equivalence relations.

Although their criticism extends to tree-to-tree alignment, we argue in this work for the potential merit in using specialized monolingual parsers if those are available, after which trees could be post-processed to improve convergence (see, for example, Ambati and Lavie (2008) and Burkett and Klein (2012)).

Lü et al (2001) use so-called *bracketing inversion transduction grammars* for the extraction of translation templates, similar to the approach of Wu but more simplified. To counter the extraction of ungrammatical phrase pairs, they experiment with parsing one or both sides using monolingual parsers for Chinese and English and using the resulting structures as constraints. For their tree-to-tree alignment experiments, an accuracy of 88.25% is reported, although they admit that the corpus is somewhat small and “normative”.

Menezes and Richardson (2001) align nodes in *logical form structures* (LFs) in the context of the extraction of transfer mappings for MT, using rule-based heuristics (18 rules in total). The words in the LFs are aligned using a pre-existing bilingual dictionary. On the word level, coverage is extended by aggressively matching tentative correspondences and with the help of a deriva-

tional morphology component. Constituent alignment is done by following a best-first strategy and using the best alignments as anchor points.

Groves et al. (2004) present a similar work but using phrase-structure trees instead, produced by Data-Oriented Parsing (DOP) grammars. They also make use of a probabilistically derived bilingual dictionary instead of a pre-existing one. Interestingly, they only use five rules whereas Menezes and Richardson (2001) use 18. Some of the rules make use of specific labels, meaning that some work has to be done to adapt it for other language pairs. They obtain a precision of 73.7% and a recall of 67.84% on their data set which they regard as an indication that their algorithm “provides a serious alternative to manual alignment”. However, their work only focuses on MT in the Data-Oriented Translation (DOT) paradigm (Hearne and Way, 2006) which places a number of additional restrictions on the tree structure and the range of possible alignments, effectively preventing any meaningful direct comparisons with our approaches.

Imamura (2001) applies a high-recall word alignment approach after which corresponding subtree candidates are extracted and scored using a structural similarity measure which they call a *phrase score*. Conditions for the acceptance of phrase equivalence include the requirements that the phrases should have similar enough syntactic categories and that words must correspond semantically as well. Zhechev (2009, p. 23) notes that this relies too much on the assumption that syntactic categories between the two languages should be the same, whereas it often happens that verb phrases, for example, are nominalized in another language.

Eisner (2003) proposes the use of synchronous tree substitution grammars (STSGs) to train MT systems on tree pairs. Hypotheses of all possible tree derivations and all the possible alignments between these derivations are generated, after which various algorithms and machine learning methods are used to learn the tree probabilities, select the best alignments and find the optimal translations. Experiments are done on dependency structures, but the system can also be adapted for phrase structures. However, Zhechev (2009) suggests that this would be computationally much more complex.

Another dependency tree alignment approach is described by Ding et al. (2003). In this case, it is presented as a word alignment algorithm which uses the dependency structures as a source of information. The structure is decomposed into subgraphs called *treelets* which are then also aligned. Experiments are done on a Chinese-English corpus. Slight improvements on IBM word alignment models are reported.

Gildea (2003) presents an alignment approach that is integrated into the translation process, with the possibility of using either a tree-to-tree or a tree-

to-string module. Cloning and reordering operations are introduced in order to deal with structural divergence. Instead of precision and recall, results are reported in terms of alignment error rate (AER). An improvement is measured according to the AER metric when clones are introduced (.36). It seems like a promising approach to tackling the problem of non-isomorphism between trees. However, their parallel corpus (Korean-English) is somewhat limited, in the sense that it is only represented by a single domain (military) and that the average Korean sentence length may be a bit short (13 tokens), at least in comparison with our data sets where the average length often exceeds 20 tokens. Although the 5083 sentence pairs can be considered enough for tree alignment experiments, tree annotation has been done by hand as opposed to fully automatically created trees, making it difficult to draw conclusions as to its applicability for the creation of large parallel treebanks, should it be applied in this way.

According to Ambati and Lavie (2008), tree-to-tree alignment suffers from coverage problems because of segmentation issues, whereas tree-to-string approaches have a higher recall but a lower precision. They propose a solution in the form of restructuring the target parse trees while simultaneously preserving the syntactic boundaries of the original trees. On their French to English MT system, this approach improves their BLEU score, although still doing less well than a Moses system trained on the same model (Koehn et al., 2007). According to Zhechev (2009, p. 25-26), their approach is too dependent on correct word alignment.

Lavie et al. (2008) implement a novel arithmetic mapping algorithm for phrase-structure tree alignment called Prime Factorization and Alignments algorithm (PFA). Each aligned leaf node is assigned a unique prime number, where the unaligned nodes have the value 1. The same is true for the nodes to which these leaves link. Their constituent parent nodes are assigned a number that is the product of these numbers. In this way, aligned non-terminals can be retrieved by simply looking at the values of the numbers. It is quite an elegant algorithm but is again very dependent on the quality of the word alignment, while other useful features such as labels are not considered. For Chinese to English alignment, they report a precision of 81.29 and a recall of 73.25.

Tinsley et al. (2007), Zhechev and Way (2008) and Zhechev (2009) describe a fast unsupervised approach for the alignment of phrase-structure trees, although it could be adapted for dependency structures as well (Zhechev, 2009, p. 100-101). Four different modes of operation are presented: tree-to-tree, tree-to-string, string-to-tree and string-to-string. It makes use of lexical probability tables of automatic word alignment output in order to guide non-terminal

alignment and assign further probabilities to the new links. Several different alignment strategies can be employed. It is designed to be as language-independent as possible. Experimenting on the aforementioned manually created English-French HomeCentre treebank (Hearne and Way, 2006), non-terminal alignments achieve precision scores ranging between 78.67% and 83.83% and recall scores ranging between 70.17% and 77.91%, depending on the strategy used (Zhechev, 2009, p. 69). Although there is much merit to this approach and the system has been used with some success, it does not consider some features, such as labeled data, which could improve its accuracy.

Liu et al. (2009) propose tree-to-tree alignment using *packed forests*. As a measure to improve alignment coverage, instead of using 1-best trees produced by parsers, many candidate trees of the same sentence pairs are aligned. Training a Chinese-to-English syntax-based MT system using *probabilistic synchronous tree substitution grammars*, the use of packed forests increased their BLEU scores by a good amount (30.59), even slightly better than Moses trained on the same data (30.43), although not statistically significant (Liu et al., 2009, p. 565). This is a promising approach which we also discuss in the future work section in chapter 9.

Macken (2010) presents a subsentential alignment system for the purpose of bilingual terminology extraction. The alignment units are restricted to words, word groups or chunks. A bootstrapping approach is described where the recall of the intersection of GIZA++ alignments is increased without sacrificing precision. The system is able to align discontinuous chunks, which is an advantage with language pairs such as Dutch and English, where some Dutch verbs are split and occur in different parts of the sentence. We describe in the next chapter how we dealt with this problem in the context of full tree alignment.

In the work of Sun et al. (2010), so-called Bilingual Tree Kernels (BTKs) are proposed for modeling the tree structures. Lexical (from automatic word alignment) and syntactic features are extracted and used to describe subtree correspondences using a polynomial kernel, which is then combined with the BTKs to construct a composite kernel. They then implement a binary classifier using Support Vector Machines (SVM) in a greedy search algorithm to classify each node pair as aligned or unaligned. Next, they apply the system on a Chinese-English parallel treebank, HIT, containing “Chinese-English Machine Translation evaluation campaigns, English books of Chinese Universities and Middle Schools as well as dictionaries”.³ The trees and word alignments in the gold standard are manually checked and the non-terminal alignments

³<http://mitlab.hit.edu.cn/index.php/resources.html>

manually produced. After several experiments, the best F-score produced is 85.32, compared to the baseline (65.53) achieved by the tool described by Tinsley et al. (2007). Running the system on another parallel corpus, FBIS,⁴ produces a slightly lower F-score of 82.7, where the baseline system scores a much higher 74.36. Although the gold standard also contains subtree alignments produced by humans, this resource has been automatically parsed.

Training various MT systems on the FBIS corpus produces decent results (Sun et al., 2010, p. 313-14). Their alignment evaluation results are comparable with those of our transformation-based learning system described in chapter 7, where a ten-fold cross validation experiment produces an F-score of 82.5. Note that we used a set of 350 sentence pairs, which is much less than the 5,000 sentence pairs that the authors used for their experiments on the HIT corpus. We also only worked on automatically produced and uncorrected phrase-structure trees and word alignments.

Unfortunately, unless experiments are done on the same data using the same language pairs, no direct comparison can be made. However, it can be said that our TBL system does have the advantage over pure stochastic systems in that error analysis is made much easier because of its rule-based approach.

Araújo and Caseli (2010) attempt to improve the PFA algorithm presented by Lavie et al. (2008) in the context of aligning Brazilian Portuguese and English trees, by using *part-of-speech filtering*, a high-precision approach. A word alignment gold standard is produced of which all POS tag pairs of aligned words are stored in a database. Then by applying PFA, only those word pairs for which the POS tags are to be found as pairs in the database are taken into account. Experiments are done using both normal PFA and the adapted method, on data that has been both manually and automatically aligned. Using regular PFA on manual word alignments, they obtain an F-score of 83.3. However, when applied on GIZA++ output - whether source-to-target, target-to-source, or union - the recall drops to the 20s. Applying POS filtering improves the recall, but it is still very low (in the 30s). Notably, the GIZA++ output is of bad quality in this case, with a best F-score of 40.3 (Portuguese to English), which was probably the major contribution to the poor results. However, using POS filtering increased the precision of GIZA++.

A follow-up paper (Araújo and Caseli, 2011) reports experiments on a different data set containing 108 sentence pairs from the Pesquisa FAPESP Brazilian magazine.⁵ Much better GIZA++ scores are reported although it is not stated

⁴http://malt.ml.cmu.edu/mw/index.php/FBIS_corpus

⁵<http://revistapesquisa.fapesp.br>

what the source of the training data is. Generally, accurate word alignments can only be obtained by training on a very large parallel corpus. The output of the approaches reported in Tinsley et al. (2007) and the PFA algorithm (Lavie et al., 2008) are combined in various intersection and union heuristics. The resulting scores are very high, with a best F-score of 92.18 using POS filters. Unfortunately, it remains unclear what text type was used, the nature of which may make a great difference in alignment accuracy. This, along with the fact that the test set is quite small, may make any definite conclusion premature. However, the authors intend to apply this method to a larger collection of 16,994 Brazilian Portuguese-English tree pairs (Araújo and Caseli, 2011, p. 172).

Nakazawa and Kurohashi (2011) align dependency subtrees for English-Japanese in a fashion reminiscent of Matsumoto et al. (1993). In this case, they implement a Bayesian alignment model which incorporates a tree-based reordering model. Their experiments achieve a 3.52% better alignment error rate (AER) than using a state-of-the-art word alignment model (bidirectional GIZA++ with neighbouring heuristics), suggesting that word alignment incorporating dependency structures may improve results.

In the previous section, we have discussed the English-Romanian parallel treebank reported by Colhon (2012). Romanian trees are generated by using information obtained from English parse trees, word alignments and Romanian POS tags. Through this process, the trees are aligned. They achieve very accurate results, with a best precision of 86.9% and a best recall of 84.1%. However, one should note that the trees are perfectly convergent, since Romanian trees were built based on the English ones. Moreover, word alignments were manually produced and the corpus was manually POS tagged, lemmatized and tokenized. It is clear that the accuracy of Romanian trees is very dependent on the accuracy of the word alignment as well as the quality of the English trees. The imposed structural isomorphism leads to the conclusion that no alignments are possible for more language-specific constructs. For dealing with this, the authors propose additional pre- and post-processing of the data.

In this work, we present a platform for tree-to-tree alignment for a wide variety of needs. In particular, we experiment with using 1-best parse tree pairs in the context of a syntax-based MT project which uses large parallel treebanks for training data (next chapter). For this project, the assumption is made that 1-best trees from quality parsers are sufficient for the extraction of enough transfer rules from parallel treebanks with a few million sentence pairs. We also believe that tree-to-tree alignment should in theory at least be able to provide a richer source of information for syntax-based MT than the previously described string-to-string, tree-to-string or string-to-tree methods. Our systems

should be able to process forests as well. However, this is an endeavour on which we hope to be able to work in the future.

The main issue in tree-to-tree alignment, non-isomorphism or lack of convergence, which leads to a reduction in coverage for the extraction of transfer rules, is one of the problems which we attempt to tackle in this work. In short, we increase the recall of alignment by relaxing the well-formedness requirement for constituent alignment as well as applying various alignment strategies.

Although we aim to improve MT performance, our main concern is improving alignment quality. With our statistical aligner *Lingua-Align* (Tiedemann and Kotzé, 2009a; Tiedemann and Kotzé, 2009b; Tiedemann, 2010), we address a problem which has been hampering most previously described systems, which is a lack of flexibility using different features, parameters and alignment strategies. Using maximum entropy modeling with a rich feature set, experiments show that training data sets as low as 100 sentence pairs may lead to F-scores close to 80. This system is described in chapter 4.

In chapter 7, we also present a new system (Kotzé, 2012) which uses the transformation-based learning method (Brill, 1995). We describe experiments in which we show that rule-based alignment is an effective complement to statistical alignment and lead to high rates for both precision and recall. Additionally, this allows for a much simpler error analysis procedure than in the case of stochastic systems. It also uses a feature set that draws from existing word and constituent alignments, labels and the tree structure, making this also a flexible tool for alignment as well as alignment error correction. Here, too, we show that a few hundred sentence pairs are sufficient to produce very accurate results.

2.4 Conclusion

We have presented a short introduction to some concepts relevant to this work and summarized most of the work done in this field. In the next chapter, we present Parse and Corpus-Based Machine Translation (PaCo-MT), a syntax-based MT system that relies on tree-to-tree alignments from large parallel corpora.

The PaCo-MT project

3.1 Introduction

In this chapter, we present Parse and Corpus-Based Machine Translation, PaCo-MT in short, which is a project during which several parallel treebanks and a novel syntax-based MT system with the same name were created. The parallel treebanks were used as a knowledge base for the training of this system. This project served as the testing ground for the research presented in this work.

Apart from some minor changes, sections 3.2, 3.4, 3.5, 3.6 and 3.7 were described in Vandeghinste et al. (2013). Section 3.3 is mostly new content, some of which is based on Kotzé et al. (2012).

The PaCo-MT system itself was mainly implemented by Dr Vincent Vandeghinste and Dr Scott Martens. Vandeghinste is also the author of the tree restructuring script as described in section 3.3.

3.2 Background

The current state-of-the-art in machine translation consists of *phrase-based statistical machine translation (PBSMT)* (Koehn, 2010), an approach which has been used since the late 90's, evolving from word-based SMT proposed by IBM (Brown et al., 1990). These *string-based* techniques, which use no linguistic knowledge, seem to have reached their ceiling in terms of translation quality, while there are still a number of limitations to the model. It lacks a mechanism to deal

with long-distance dependencies, it has no means to generalize over non-overt linguistic information (Riezler and Maxwell III, 2006) and it has limited word reordering capabilities. Furthermore, in some cases the output quality may lack appropriate fluency and grammaticality to be acceptable for actual MT users. Sometimes essential words are missing from the translation.

To overcome these limitations, efforts have been made to introduce *syntactic knowledge* into the statistical paradigm, usually in the form of syntax trees, either only for the source (tree-to-string) or the target language (string-to-tree), or for both (tree-to-tree).

Galley et al. (2004) describes an MT engine in which *tree-to-string* rules have been derived from a parallel corpus, driven by the problems of SMT systems raised by Fox (2002). Marcu et al. (2006) and Wang et al. (2010) describe *string-to-tree* systems to allow for better reordering than phrase-based SMT and to improve grammaticality. Hassan et al. (2007) implement another string-to-tree system by means of including supertags (Bangalore and Joshi, 2010) to the target side of the phrase-based SMT baseline.

Most of the *tree-to-tree* approaches use one or another form of *synchronous context-free grammars* (SCFGs) a.k.a. syntax directed translations (Aho and Ullman, 1969) or syntax directed *transduction* grammars (Lewis and Stearns, 1968). This is true for the tree-based models of the Moses toolkit,¹ and the machine translation techniques described in, amongst others, Wu (1997), Yamada and Knight (2001), Probst et al. (2002), Chiang (2005), Zollman and Venugopal (2006) and Lavie (2008). A more complex type of translation grammars is *synchronous tree substitution grammar* (STSG) (Schabes, 1990; Eisner, 2003) which provides a way, as Chiang (2006) points out, to perform certain operations which are not possible with SCFGs without flattening the trees, such as raising and lowering nodes. Examples of STSG approaches are the *Data-Oriented Translation* (DOT) model from Poutsma (2003) and Hearne and Way (2003) which uses data-oriented parsing (Bod, 1992) and the approaches described in Graham (2010), Graham and Van Genabith (2010a), Graham and Van Genabith (2010b) and Riezler and Maxwell (2006), using STSG rules consisting of dependency subtrees, and a top-down transduction model using beam search.

The *Parse and Corpus based Machine Translation* (PaCo-MT) project (2008-2011) was sponsored by the STEVIN programme (STE07007)² of the Dutch Language Union. Project partners included Frank van Eynde, Vincent Vandeghinste and Joachim Van den Bogaert from the Catholic University of Leuven (KU

¹<http://www.statmt.org/monos>

²<http://www.stevintst.org>

Leuven), Belgium, Jörg Tiedemann and the author of this dissertation from the University of Groningen (RuG), The Netherlands, as well as Koen Desmet from the company OneLiner bvba in Sint-Niklaas, Belgium.

KU Leuven was responsible for the main development of the PaCo-MT system. RuG worked on creating a series of large parallel treebanks to be used as training data for the system, while OneLiner provided us with the test sets to be used in evaluation, as well as translation memories which were also processed to comprise part of the parallel treebanks.

The PaCo-MT engine described in this chapter³ is another tree-to-tree system that uses an STSG, differing from related work with STSGs in that this engine combines dependency information with constituency information and that the translation model abstracts over word and phrase order in the synchronous grammar rules: the daughters of any node are in a canonical order representing all permutations. The final word order is generated by the tree-based target language modeling component.

Figure 3.1 presents the architecture of the PaCo-MT system. A source language (SL) sentence gets syntactically analysed by a pre-existing parser which leads to a source language parse tree, making abstraction of the surface order. This is described in section 3.3. The unordered parse tree is translated into a forest of unordered trees (also known as a bag of bags) by applying *tree transduction* with the *transfer grammar* which is an STSG derived from a parallel treebank. Section 3.5 presents how the transduction grammar was built and section 3.6 how this grammar is used in the translation process. The forest is decoded by the *target language generator*, described in section 3.7 which generates an *n*-best list of translation alternatives by using a tree-based target language model. In chapter 8, we present the results of applying various evaluation metrics against the MT output of a test set.

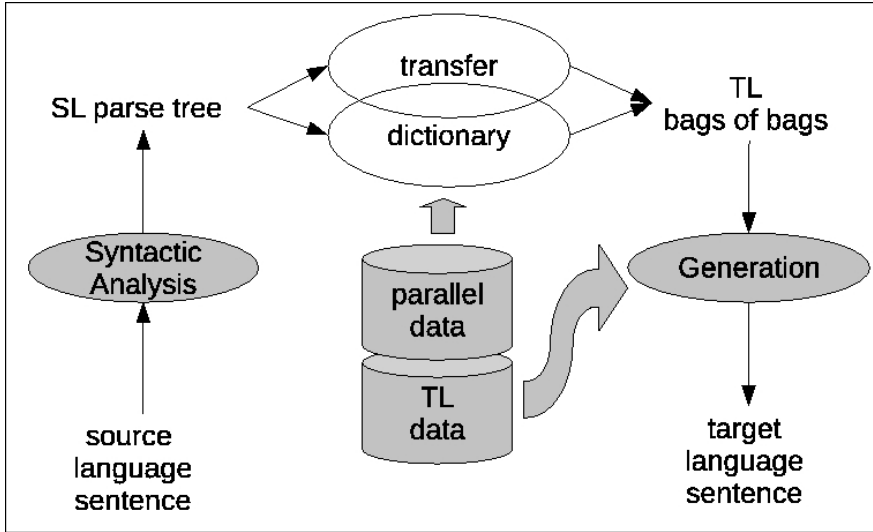
Next, we describe how we created the resource necessary for grammar induction - the parallel treebank.

3.3 Parallel treebank creation

In the previous chapter, we have mentioned the pipeline of activities that are often employed in the process of constructing parallel treebanks, which in our case are used for the purpose of inducing translation grammars for PaCo-MT. These parallel treebanks were produced by obtaining a collection of parallel

³Previous versions were described in Vandeghinste and Martens (2009) and Vandeghinste and Martens (2010).

Figure 3.1: The architecture of the PaCo-MT system



corpora containing only the raw text and processing them automatically by applying a series of tools. In short, these steps were followed in chronological order:

- corpus selection and cleaning
- sentence alignment
- tokenization
- parsing
- word alignment
- constituent alignment

Because of availability for research purposes, we collected Dutch-English and Dutch-French subsets of the following parallel corpora:

- Europarl 3 (Koehn, 2005), a corpus containing the proceedings of the European Parliament in several languages from 1996 to 2006.

Table 3.1: Word and sentence counts on the 1:1 Dutch-to-English and English-to-Dutch parallel treebanks.

Corpus	Dutch-English		English-Dutch	
	Sentence pairs	Words	Sentence pairs	Words
Europarl	1,180,706	57,812,824	1,190,501	57,810,789
DGT-TM	478,972	19,974,703	484,186	20,222,021
OPUS+TM	1,097,834	20,393,721	1,102,958	20,493,131
Total	2,757,512	98,181,248	2,777,645	98,525,941

- the DGT Multilingual Translation Memory of the Acquis Communautaire: DGT-TM.⁴ This is a series of translation memories containing texts from different domains.
- OPUS:⁵ the open parallel corpus (Tiedemann, 2009).⁶
- an additional private translation memory (Transmem). Again, various different domains are represented.

After applying some cleaning procedures, we proceeded to align the sentences. We assumed an input of sentence-segmented bitexts, although tokenization was not necessary at this point. Initially, we used the sentence aligner of Gale and Church (1993), but soon we switched to Hunalign (Varga et al., 2005) after an analysis of some output sentences suggested better performance.

We filtered the output of Hunalign to only obtain the 1:1 sentences for further sub-sentential alignment. However, all sentences were eventually parsed, as we intended to use as much data as possible for target language modeling.

Table 3.1 displays the word and sentence counts of the 1:1 Dutch-to-English and English-to-Dutch parallel treebanks, where Table 3.2 displays the same statistics for the Dutch-to-French and French-to-Dutch parallel treebanks.

Dutch input sentences are parsed using Alpino (Van Noord, 2006), a stochastic rule-based dependency parser, resulting in structures as in Figure 3.2. The Alpino distribution has its own tokenizer which we applied before the parsing step. Part-of-speech tagging is integrated into the parsing procedure.

⁴<http://langtech.jrc.it/DGT-TM.html>

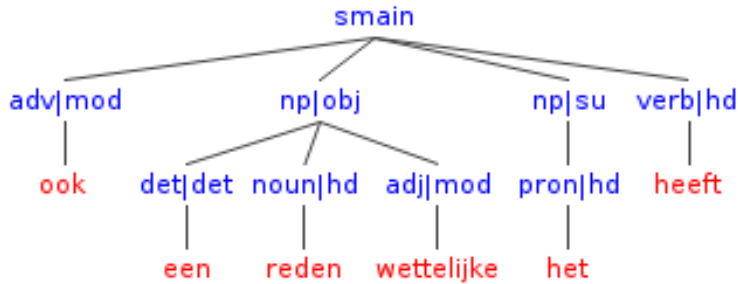
⁵Open Parallel corpUS

⁶<http://opus.lingfil.uu.se>

Table 3.2: Word and sentence counts on the 1:1 Dutch-to-French and French-to-Dutch parallel treebanks.

	Dutch-French		French-Dutch	
Corpus	Sentence pairs	Words	Sentence pairs	Words
Europarl	1,188,022	60,987,015	1,188,757	61,231,725
DGT-TM	385,317	17,750,535	385,644	17,801,831
OPUS+TM	720,487	18,962,966	722,494	19,038,759
Total	2,293,826	97,700,516	2,296,895	98,072,315

Figure 3.2: An unordered parse tree for the Dutch sentence *Het heeft ook een wettelijke reden* “It also has a legal reason”, or according to Europarl “It is also subject to a legal requirement”. Note that edge labels are marked behind the ‘|’.



We applied some limited restructuring to make the resulting parse trees more uniform. For example, lonely nouns, pronouns and names are placed under an NP. For Dutch as a source language, so-called *split verbs* are merged in order to increase convergence with their English counterparts.⁷ A similar restructuring of syntax trees is shown by Wang (2010) to improve translation results.

English sentences are tokenized using TreeTagger (Schmid, 1994), after which they are annotated with the Stanford phrase structure parser (Klein and Manning, 2003) containing dependency information (De Marneffe et al., 2006).

⁷In Dutch, verbs that are combined with prepositions, such as *opstaan* (stand up) are written as one word. However, under some circumstances, they are split in two (*staan* and *op*) and these split parts often occur in different parts of the sentence, creating problems for constituent alignment. These cases were those that we have merged.

The bracketed phrase structure and the typed dependency information are integrated into an XML format consistent with the Alpino-XML format. The tree structure is then enriched with lemmas, also using the TreeTagger tool.

For French, we experimented with various different parsers. Eventually, we decided to use a version of the Berkeley parser (Petrov and Klein, 2007), trained on French data. As the model does not add dependency information, we implemented head-finding rules in a similar style as De Marneffe (2006). The output is a bracketed phrase structure format which we also converted to Alpino-XML for further processing.

After the parsing and the abovementioned tree restructuring process for Dutch have been completed, we proceeded to word align all 1:1 sentence pairs. As mentioned in the previous chapter, we have opted to use GIZA++ (Och and Ney, 2003) which implements an asymmetric approach that can be applied in two directions: source to target and target to source. GIZA++ implements generative models in an unsupervised learning approach.

The output of both directions can be combined to produce an intersection or union, corresponding to a high-precision and high-recall approach, respectively. Additionally, one can employ various heuristics to iteratively add neighbouring alignments in an attempt to balance high-precision with high-recall alignments. This is implemented by the aforementioned SMT system Moses. We have adopted *grow-diag*, which first finds an aligned word pair, then check for their immediate neighbours. If any of them is unaligned, and both of them are in the union, they are also aligned.

Another step is the so-called *final* step, which adds yet more alignments based on similar considerations. However, our opinion was that *grow-diag* was a suitably balanced choice which increased the recall somewhat but still kept the precision reasonably high.

Constituent or non-terminal alignment, which we will simply call tree alignment in most cases,⁸ was the last important step in the creation of our parallel treebanks. At the time, there was a relative scarcity of good alignment software dedicated to this task. This has led to the development of the statistical tree alignment toolbox Lingua-Align, described in the next chapter.

Because it applies supervised learning, Lingua-Align requires training data in the form of quality aligned parallel tree pairs. Experiments with a manually

⁸Technically, tree alignment also includes word alignment, as words are associated with the leaves of the tree. However, in this work we make a clear distinction between word alignment and constituent alignment, as these are two very different tasks. It is the case, though, that in the literature, discussions of phrase-structure tree-to-tree alignment often put the focus on the alignment of non-terminals.

produced parallel treebank, SMULTRON (Gustafson-Čapková et al., 2007), suggested that a relatively small amount of training data is sufficient to produce quite accurate alignments. For the language pairs used in PaCo-MT, we could not find any gold standards. Therefore, we decided to make our own training data.

A very useful tool for this purpose is the Stockholm TreeAligner (Lundborg et al., 2007). It offers an interface showing graphical representations of the trees, where one can manually draw lines between nodes indicating the alignments. By default, a distinction is made between so-called *good* and *fuzzy* alignments, which correspond to confident and less confident alignments. However, one may specify any number of link types when setting up a project.

The Stockholm TreeAligner only accepts trees in TIGER-XML format (Mengel and Lezius, 2000), which is a phrase-structure representation which can also encode dependencies and other metadata. Therefore, we proceeded to develop and apply scripts for the conversion of the outputs of the Stanford, Berkeley and Alpino parsers to TIGER-XML.

The source and target TIGER-XML files have unique identifiers for each word and each constituent. A native XML format, which we have adopted in Lingua-Align, is used to refer to the identifiers in these two files. Pairs of references to both a source and a target node can be found, implying an alignment. Additional information can be added such as the author, whether or not the alignment is good or fuzzy, and so on.

The following is an example:

```
<align type="good" author="Lingua-Align">
  <node treebank_id="1" node_id="s1_0"/>
  <node treebank_id="2" node_id="s1_501"/>
</align>
<align type="good" author="Lingua-Align">
  <node treebank_id="1" node_id="s1_1"/>
  <node treebank_id="2" node_id="s1_502"/>
</align>
<align type="good" last_change="2012-07-17" author="Gideon">
  <node treebank_id="1" node_id="s1_2"/>
  <node treebank_id="2" node_id="s1_503"/>
</align>
```

The above displays three alignments from a gold standard file, each between three different node pairs. Node that the third alignment is manual, made by

the author of this dissertation, and the first two have been automatically created by Lingua-Align. This is because we created some of our gold standards by correcting output from this system.

Construction of gold standards for tree alignment

Unlike some efforts to construct a parallel treebank completely by hand, as discussed in the previous chapter, we select our tree pairs from automatically produced alignment sets. In other words, sentence alignment, parsing and word alignment have been done automatically in all cases. Apart from saving much time, in our opinion, this also enables our aligner to model a typical input tree pair more closely. For example, if we corrected all word alignments in favour of a high-precision approach and trained our models on this data, the tree aligner might make unexpected decisions in the presense of input containing high-recall word alignments. However, during the PaCo-MT project we did construct data sets where the word alignments were corrected in the case of Dutch to English as well as English to Dutch (see section 4.7).

In constructing our gold standards, we thought it wise to follow certain guidelines. Already mentioned in the previous chapter, the *well-formedness constraint*, mentioned by Hearne et al. (2007) makes logical sense and helps to ensure consistency:

- Descendants of a source linked node may only be linked to descendants of its linked node on the target side and vice versa (target to source).
- Similarly, ancestors of a source linked node may only be linked to ancestors of its linked node on the target side and vice versa (target to source).

However, since word alignment is not perfect, strict application of this constraint invariably leads to low coverage, leaving many non-terminals unaligned. For the purpose of MT, which typically functions better the more data is available for training, this is undesirable. Therefore, we have ignored this constraint in cases where we still deemed the phrases worthy of alignment.

Samuelsson and Volk (2007) discuss a few more guidelines which they applied to building the SMULTRON parallel treebank. One of them is to never align an NP containing a pronoun to a full noun phrase. This makes sense for MT, since learning to produce any specific noun phrase as a translation for a pronoun should not depend on probabilities learned from a whole parallel corpus, but should instead be the result of co-reference resolution. However, translating a full noun phrase as a pronoun can be acceptable in some cases, since this is

much less context dependent. For example, one can always translate *the girl* to the equivalent of *she* or *her* (depending on the grammatical and discourse context) in the other language. However, *she* or *her* can only be translated to *the girl* in very specific contexts - it does not suffice to know that translating these words to *the girl* has certain probabilities. Although we have not explicitly implemented this distinction, choosing to keep pronouns unaligned (unless the other NP also contains a pronoun) this may be useful in future versions of our gold standards.

Similarly, one side of a translation can be either more or less specific than the other side. In the case of noun phrases, the more specific phrase can be considered a *hyponym* of the less specific phrase, which is a *hypernym* of the more specific phrase. For example, *the vehicle* can be considered a hypernym of *the car*, which is a hyponym of *the vehicle*. This is a typical is-a relation, found in the literature surrounding lexical databases such as WordNet (Miller, 1995; Fellbaum, 1998). In later experiments in the context of our transformation-based learning alignment system (chapter 7) we have allowed alignments where the target side is less specific, but not vice versa. All cars are vehicles, but not all vehicles are cars - it must be determined from context whether or not a specific vehicle is also a car.

Sometimes, a candidate phrase pair is not exactly equivalent, but the phrases also do not stand in an is-a relationship to each other. This is a very common occurrence and is often the result of a freer translation in order to conform to language-specific stylistic requirements. Although in some cases the decision was not clear-cut, we have used the advice of Samuelsson and Volk (2007, p. 3) as a guideline, which states:

Phrases shall be aligned only if the tokens that they span represent the same meaning, if they could serve as translation units outside the current sentence context. The grammatical forms of the phrases need not fit in other contexts, but the meaning has to fit.

Similar to what the authors did, we aimed to align phrase pairs with lesser translational equivalence using fuzzy links, reserving good links for exact matches.

Since we decided to follow a high-recall alignment approach suitable for MT, we have allowed 1:n and n:1 alignments of words. However, we implemented a strict 1:1 non-terminal alignment approach. The reason for this is that in the case of 1:n or n:1 phrase alignments, it necessarily means that one of them is not exact. The non-exact pair can of course still be aligned using a fuzzy

link, but this information is superfluous, since we have already found an exact alignment.

Words, however, can be aligned using any combination of links, since it is common for more than one word, as a group, to be translationally equivalent to one or more word in the other language.

Samuelsson and Volk (2007) also mention the importance of checking for *completeness* and *consistency*. A tree is complete if every token and constituent is part of it, while a treebank is consistent if “the same token sequence (or part-of-speech sequence or node sequence) is annotated in the same way across the treebank” (Samuelsson and Volk, 2007, p. 2).

As an example of consistency checking, the authors extract all function triples containing a mother category, the function label and a daughter category. Where the mother category is a VP and the function is “accusative-object”, in the vast majority of cases the daughter category is an NP. A handful of daughter categories was different, which may point to erroneous labeling.

In the case of alignment, one may, for example, extract examples of which category label pairs are typically aligned and what functional categories are encoded in their respective edge labels. This may give clues to the degree of consistency to which the nodes were aligned. However, although we did run a little experiment where we extracted the category label pairs for incorrect alignments (see chapter 6), we did not check for full consistency. Although we obviously attempted to construct our gold standards as accurately as practically possible, the reality was that we worked with automatically produced parse trees and that the results of such a consistency check may not necessarily point to real errors.

We believe that experiments using inter-annotator agreement from the output of multiple annotators would be more sufficient in this case. In the future, it may be interesting to measure the impact of such an approach to both tree alignment and MT evaluation more precisely.

Automatic tree alignment

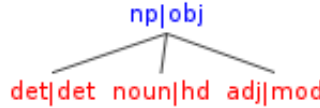
For all the language pairs, we have constructed and applied various models for alignment with Lingua-Align. Our experiments are described in chapter 4.

In the latter part of the project, we have experimented with using rules in order to increase alignment recall. These experiments are discussed in chapter 6. Table 3.3 displays the counting statistics of all alignments made for the final versions of our parallel treebanks.

Language pair	M/G++	Lingua-Align	Rules	Total
Dutch/English	48,011,840	5,049,496	9,818,623	62,879,959
English/Dutch	38,060,911	11,820,036	9,301,941	58,182,888
Dutch/French	46,976,046	9,128,714	6,021,967	62,126,727
French/Dutch	47,017,307	6,237,574	5,259,590	58,514,471
Total	180,066,104	32,235,820	30,402,121	242,704,045

Table 3.3: Counts of alignments applied to all the parallel treebanks, where M/G++ denotes Moses/GIZA++.

Figure 3.3: An example of a horizontally complete subtree which is not a bottom-up subtree.



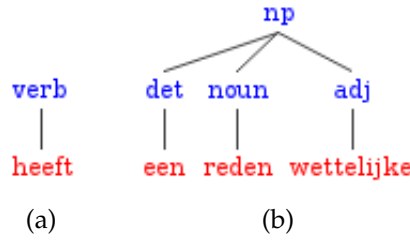
3.4 Syntactic analysis

Abstraction is made of the surface order of the terminals in every parse tree used in the PaCo-MT system. An *unordered tree* is defined⁹ by the tuple $\langle V, V^i, E, L \rangle$ where V is the set of nodes, V^i is the set of internal nodes, and $V^f = V - V^i$ is the set of frontier nodes, i.e. nodes without daughters. $E \subset V^i \times V$ is the set of directed edges and L is the set of labels on nodes or edges. $V^l \subseteq V^f$ is the set of lexical frontier nodes, containing actual words as labels, and $V^n = V^f - V^l$ is the set of non-lexical frontier nodes, which is empty in a full parse tree, but not necessarily in a subtree. There is exactly one root node $r \in V^i$ without incoming edges. Let T be the set of all unordered trees, including subtrees.

A subtree $s_r \in T$ of a tree $t \in T$ has as a root node $r \in V_t^i$ where V_t^i is the set of internal nodes of t . Subtrees are *horizontally complete* (Boitet and Tomokiyo, 1995) if, when a daughter node of a node is included in the subtree, then so are all of its sisters. Figure 3.3 shows an example. Let $H \subset T$ be the set of all horizontally complete subtrees.

⁹This definition is inspired by Eisner (2003).

Figure 3.4: Two examples of bottom-up subtrees



Bottom-up subtrees are a subset of the horizontally complete subtrees: they are lexical subtrees: every terminal node of the subtree is a lexical node. Some examples are shown in figure 3.4. Let $B \subset H$ be the set of all bottom-up subtrees. $\forall b \in B : V_b^n = \emptyset$ and $V_b^l = V_b^f$, where V_b^n is the set of non-lexical frontier nodes of b and V_b^l is the set of lexical frontier nodes of b . V_b^f is the set of all frontier nodes of b .

3.5 The transduction grammar

In order to translate a source sentence, a stochastic synchronous tree substitution grammar G is applied to the source sentence parse tree. Every grammar rule $g \in G$ consists of an elementary tree pair, defined by the tuple $\langle d^g, e^g, A^g \rangle$, where $d^g \in T$ is the source side tree (Dutch), $e^g \in T$ is the target side tree (English), and A^g is the alignment between the non-lexical frontier nodes of d^g and e^g . The alignment A^g is defined by a set of tuples $\langle v_d, v_e \rangle$ where $v_d \in V_d^n$ and $v_e \in V_e^n$. V_d^n is the set of non-lexical frontier nodes of d^g , and V_e^n is the set of non-lexical frontier nodes of e^g . Every non-lexical frontier node of the source side is aligned with a non-lexical frontier node of the target side: $\forall v_d \in V_d^n$ is aligned with a node $v_e \in V_e^n$. An example grammar rule is shown in figure 3.5.

Grammar rule induction

Figure 3.6 is an example¹⁰ of two sentences aligned at both the sentence and subsentential level. For each alignment point, either one or two rules are

¹⁰The edge labels have been omitted from these examples, but were used in the actual rule induction.

Figure 3.5: An example of a grammar rule with horizontally complete subtrees on both the source and target side. Indices mark alignments

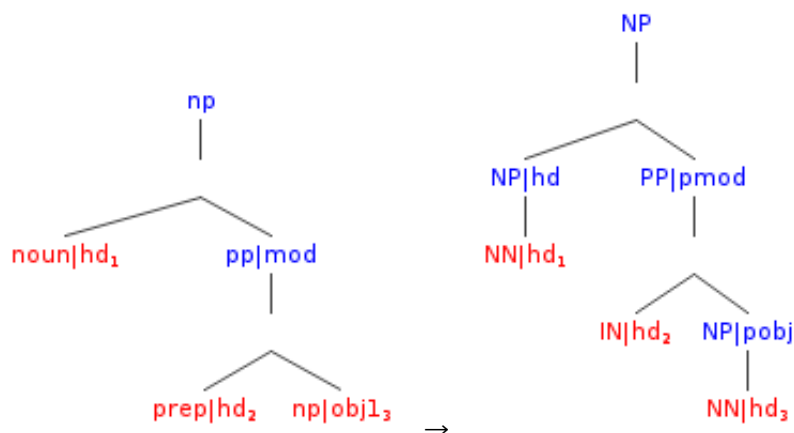
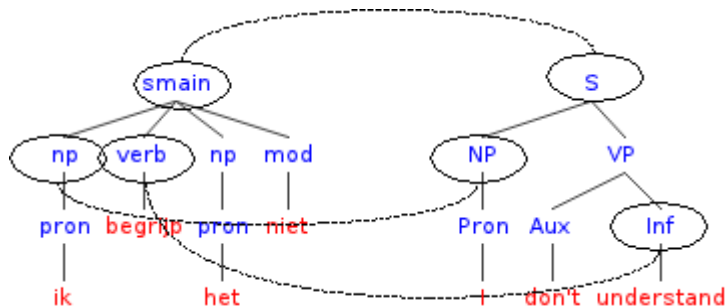


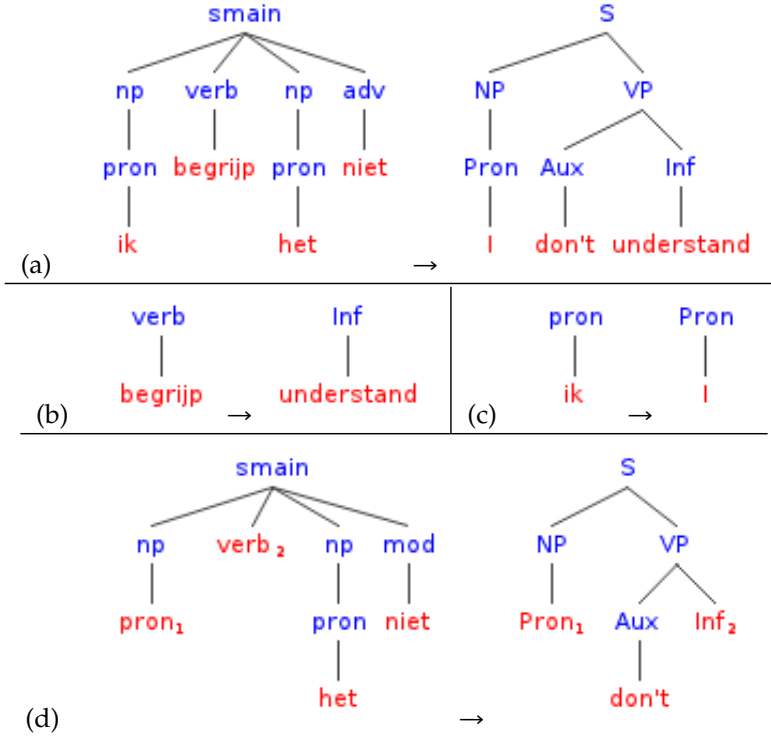
Figure 3.6: Two sentences with subsentential alignment.



extracted. First, each alignment point is a lexical alignment, creating a rule that maps a source language word or phrase to a target language one (Figure 3.7 (a), (b) and (c)).

Secondly, each aligned pair of sentences engenders further rules by partitioning each tree at each alignment point, yielding non-lexical grammar rules. For these rules, the alignment information is retained at the leaves so that these trees can be recombined (Figure 3.7 (d)).

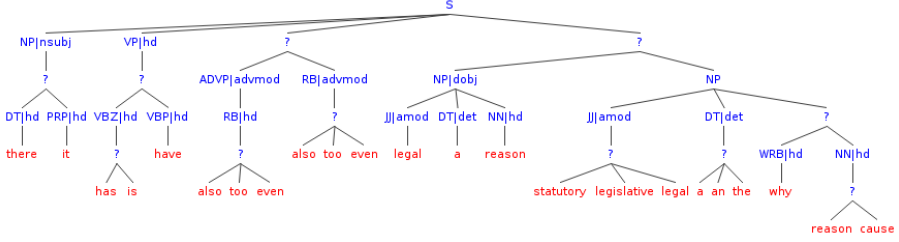
Figure 3.7: Rules extracted from the alignments in Figure 3.6



The rule extraction process was restricted to rules with horizontally complete subtrees at the source and target side. Rule extraction with other types of subtrees was considered out of the scope of the current research.

Figure 3.7 shows the four rules extracted from the alignments in Figure 3.6. Rules are extracted by passing over the entire aligned treebank, identifying each aligned node pair and recursively iterating over its children to generate a substitutable pair of trees whose roots are aligned, and whose leaves are either terminal leaves in the treebank or correspond to aligned vertices. As shown in Figure 3.7, when a leaf node corresponds to an alignment point, we retain the information to identify which target tree leaf aligns with each such source leaf.

Figure 3.8: An example of a packed forest as output of the transducer for the Dutch sentence *Het heeft ook een wettelijke reden*. Note that ? marks an alternation.



Many such tree substitution rules recur many times in the treebank, and a count is kept of the number of times each pair appears, resulting in estimates for a *stochastic* synchronous tree substitution grammar.

3.6 The transduction process

The *transduction process* takes an unordered source language parse tree $p \in T$ as input, applies the transduction grammar G and transduces p into an unordered weighted packed forest, which is a compact representation of a set of target trees $Q \subset T$, which represent the translation alternatives. An example of a packed forest is shown in Figure 3.8.

For every node $v \in V_p^i$, where V_p^i is the set of internal nodes in the input parse tree p , it is checked whether there is a subtree $s_v \in H$ with v as its root node, which matches the source side tree d^g of a grammar rule $g \in G$.

To keep computational complexity limited, the subtrees of p that are considered and the subtrees that occur in the source and target side of the grammar G have been restricted to horizontally complete subtrees (including bottom-up subtrees).

When finding a matching grammar rule for which $s_v = d^g$, the corresponding e^g is inserted into the output forest Q . When not finding a matching grammar rule, a horizontally complete subtree is constructed, as explained in section 3.6.

The weight that the target side e^g of grammar rule $g \in G$ will get is calculated according to a formula that is similar to the approaches of Graham (2010) and Riezler and Maxwell (2006), as it contains largely the same factors. We multiply the weight of the grammar rule with the relative frequency of the grammar

rule over all grammar rules with the same source side. This is divided by an alignment point penalty favouring the solutions with the fewest alignment points.

When no translation of a word is found in the transduction grammar, the label $l \in L$ is mapped onto its target language equivalent. Adding a simple bilingual word form dictionary is optional. When a word translation is not found in the transduction grammar, the word is looked up in this dictionary. If the word has multiple translations in the dictionary, each of these translations receives the same weight and is combined with the translated label (usually part-of-speech tags). When the word is not in the dictionary or no dictionary is present, the source word is transferred as is to Q .

Subtree matching

In a first step, the transducer performs *bottom-up subtree* matching, which is analogous to the use of phrases in phrase-based SMT, but restricted to linguistically meaningful phrases. Bottom-up subtree matching functions like a sub-sentential translation memory: every linguistically meaningful phrase that has been encountered in the data will be considered in the transduction process, obliterating the distinction between a translation memory, a dictionary and a parallel corpus (Vandeghinste, 2007).

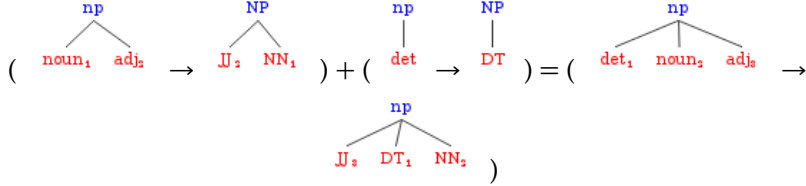
For every node $v \in V_p$ it is checked whether a subtree s_v with root node v is found for which $s_v \in B$ and for which there is a grammar rule $g \in G$ for which $d = s_v$. These matches include single word translations together with their parts-of-speech.

A second step consists of performing *horizontally complete subtree* matching for those nodes in the source parse tree for which the number of grammar rules $g \in G$ that match is smaller than the beam size b .

For every node $v \in V_p^i$ the set $H_v \subset H \setminus B$ is generated, which is the set of all horizontally complete subtrees minus the bottom-up subtrees of p with root node v . It is checked whether a matching subtree $s_v \in H_v$ is found for which there is a grammar rule $g \in G$ for which $d^g = s_v$.

An example of a grammar rule with horizontally complete subtrees on both source and target sides was shown in figure 3.5. This rule has 3 alignment points, as indicated by the indices.

Figure 3.9: An example of a constructed grammar rule



Backing off to constructed horizontally complete subtrees

In cases where no grammar rules are found for which the source side matches the horizontally complete subtrees at a certain node in the input parse tree, grammar rules are combined for which, when combined, the source sides form a horizontally complete subtree. An example of such a constructed grammar rule is shown in figure 3.9. The newly constructed grammar rule is the union of the alignments of the grammar rules that are combined in the process.

As the newly constructed grammar rule has an absolute frequency of 0, we estimate the frequency of occurrence according to formula 3.1.

$$F(f) = w(y_v) \times \frac{F(g)}{F(d^g)} \times \frac{F(h)}{F(d^h)} \quad (3.1)$$

where

- f is the newly constructed grammar rule
- g and h are the grammar rules which are combined to form f
- $w(y_v) = \sqrt[m]{\prod_{i=1}^m w(A_i^g)}$ is the weight of grammar rule g , which is the geometric mean of the weight of each individual occurrence of alignment A , as produced by tree alignment;
- $F(g)$ is the frequency of occurrence of grammar rule g
- $F(d^g)$ is the frequency of occurrence of the source side d^g of grammar rule g
- $F(h)$ is the frequency of occurrence of grammar rule h
- $F(d^h)$ is the frequency of occurrence of the source side d^h of grammar rule g^h

Constructing grammar rules leads to overgeneration. As a filter, the target language probability of such a rule is taken into account. This is explained in more detail in Vandeghinste et al. (2013).

When constructing a horizontally complete subtree fails, a grammar rule is constructed by translating each child separately.

3.7 Generation

The main task of the *target language generator* is to determine word order, as the packed forest contains unordered trees. An additional task of the target language model is to provide additional information concerning lexical selection, similar to the language model in phrase-based SMT (Koehn, 2010).

The target language generator has been described in detail in (Vandeghinste, 2009), but the system has been generalized and improved and was adapted to work with weighted packed forests as input.

For every node in the forest, the surface order of its children needs to be determined. For instance, when translating “een wettelijke reden” into English, the bag $NP\langle JJ(\textit{legal}), DT(a), NN(\textit{reason}) \rangle$ represents the surface order of all permutations of these elements.

A large monolingual treebank is searched for an NP with an occurrence of these three elements, and in what order they occur most, using the relative frequency of each permutation as a weight. If none of the permutations are found, the system backs off to a more abstract level, only looking for the bag $NP\langle JJ, DT, NN \rangle$ without lexical information, for which there is most likely a match in the treebank.

When still not finding a match, all permutations are generated with an equal weight, and a penalty is applied for the distance between the source language word order and the target language word order to avoid generating too many solutions with exactly the same weight. This is related to the notion of distortion in IBM model 3 in Brown et al. (1990).

In the example bag, there are two types of information for each child: the part-of-speech and the word token, but as already pointed out in section 3.3 dependency information and lemmas are also at our disposal.

All different information sources (token, lemma, part-of-speech, and dependency relation) have been investigated with a back-off from most concrete (token + lemma + part-of-speech + dependency relation) to most abstract (part-of-speech).

The functionality of the generator is similar to the one described in Guo et al. (2008), but relative frequency of occurrence is used instead of n -grams of dependencies. As shown in Vandeghinste (2009) this approach outperforms SRILM 3-gram models (Stolcke, 2002) for word ordering. Velidal and Oepen (2006) use *feature templates* for translation candidate reranking, but these can have a higher depth and complexity than the context-free rules used here.

Large monolingual target language treebanks have been built by using the target sides of the parallel corpora and adding the British National Corpus (BNC).¹¹

3.8 Conclusion

We have described the PaCo-MT project, mentioning both the steps involved in creating a parallel treebank, as well as the implementation of a novel syntax-based MT system, using the parallel treebanks as a knowledge base. It is in this context that the research presented in this work took place.

In chapter 8, we present a detailed evaluation of the system. In the next chapter, we discuss our experiments with the statistical aligner Lingua-Align.

¹¹<http://www.natcorp.ox.ac.uk>

Statistical tree-to-tree alignment with Lingua-Align

4.1 Introduction

In chapter 2, we have defined a parallel treebank as a collection of sentence pairs that have been grammatically tagged, syntactically annotated and aligned on a sub-sentential level. Large parallel treebanks are a valuable resource for various NLP applications such as machine translation, contrastive linguistics and corpus-based translation studies. We are interested in the alignment of non-terminal constituents as a process in the creation of such a resource. In this chapter, we present the conception, design, implementation and application of a statistical tree-to-tree aligner which we use to create a series of large parallel treebanks for use in machine translation. We credit Jörg Tiedemann for the main implementation, as well as the conceptual and mathematical development of the system.

The contents of sections 4.2, 4.3 (apart from the list of assumptions), 4.4 and 4.6 have been presented earlier in two papers, Tiedemann and Kotzé (2009a) and Tiedemann and Kotzé (2009b). We add more detailed explanations, as well as further examples, inspired by those in Tiedemann (2010). Section 4.5 contains more detailed information on parameters that is also found in the software documentation. Sections 4.7 and 4.8 are new.

4.2 Background

A number of state-of-the-art MT systems use phrase-based statistical models, some incorporating syntactic information. Other more experimental systems are more heavily syntax-based and depend on parallel treebanks where the non-terminal constituents are also aligned, such as the PaCo-MT system (Vandeghinste and Martens, 2010) and Data-Oriented Translation (DOT) based systems (Poutsma, 2000). In the context of the PaCo-MT project (see chapter 3), we have developed the statistical tree-to-tree aligner *Lingua-Align*. It is essentially a toolbox which allows the user to customize it to their needs in almost every aspect. At the time of its conception, we have realized the need for an easily customizable tool that can be used for the fast creation of parallel treebanks on a large scale.

Most existing systems have only been applied in relatively restricted environments, apart from the Dublin Aligner (Zhechev, 2009). The latter was designed to be unsupervised and as language independent as possible, only using the tree structure and existing word alignments for guidance. Although there is much merit to this approach and the system has been used with some success, it does not consider some features, such as labeled data, which could improve its accuracy. The *Lingua-Align* system, however, offers the possibility of a potentially greater accuracy that may result from implementing a rich feature set extracted from training data, as well as full customizability in terms of parameters and the aforementioned feature set.

4.3 Discriminative tree alignment

For the rest of this chapter, we make the following assumptions:

- Tree alignment is the problem of predicting the optimal set of links between nodes in the source and target phrase structure trees of an aligned sentence pair such that translational equivalence is maximized.
- We only make use of phrase-structure trees, as defined in chapter 2. Note that *Lingua-Align* can also align the dependency trees produced by the Alpino parser for Dutch, but links are still between non-terminal constituents referred to by IDs in the XML.
- For every alignment decision, the maximum hypothesis space is restricted to the sentence pair in question. Depending on the search algorithm, certain additional restrictions may apply.

- Predicting a specific alignment depends on the values of a set of features extracted from the data (see section 4.4). This includes the tree structure, existing alignments, words, labeled data and contextual data.

Lingua-Align is a statistical aligner that utilizes discriminative modeling, based on a manually specified set of features extracted from the training data (see chapter 2). A maximum entropy classifier is used to implement discriminative modeling in the form of a log-linear model. Using maximum entropy has the advantage that one does not need to consider the dependencies between the features in the underlying data structure, even when using a rich feature set. The log-linear model is learned from training data that is manually constructed before training. In short, the task is to predict the likelihood of a link a_{ij} between two nodes s_i and t_j given the features associated with these nodes. Depending on a given threshold and this calculated likelihood, the nodes are either linked or not. We therefore use a binary classification approach. However, phrase-structure trees are complex structures where alignment decisions can be influenced by previous decisions and the characteristics of nodes other than the ones that are currently being considered. It is therefore a structured prediction problem. In section 4.3, we discuss how Lingua-Align is adapted to take this into account.

A log-linear link prediction model

Similar to related work on discriminative word alignment (see, for example, Ker and Chang (1997), Ahrenberg et al. (1998) and Melamed (2000)) we base our model on features extracted for each possible alignment candidate. For tree alignment, each pair of nodes $\langle s_i, t_j \rangle$ from the source and the target language parse tree is considered and a score x_{ij} is computed that represents the degree to which both nodes should be aligned according to their features $f_k(s_i, t_j, a_{ij})$, where a_{ij} denotes the alignment between s_i and t_j , and corresponding weights λ_k derived from training data. For the score we calculate a pre-determined probability threshold, which can be manually set and functions as the cutoff point at which to decide whether or not to align the nodes.

In our approach we use conditional likelihood using a log-linear model for estimating the probability of a link:

$$P(a_{ij}|s_i, t_j) = \frac{1}{Z(s_i, t_j)} \exp \left(\sum_k \lambda_k f_k(s_i, t_j, a_{ij}) \right)$$

where Z is a normalization value such that P constitute a proper probability distribution. Here, the mapping of data points to features is user provided (see section 4.4) and the corresponding weights are learned from aligned training data. As mentioned earlier, we simplify the problem by predicting individual alignment points for each candidate pair instead of aiming at structured approaches. Hence, we can train our conditional model as a standard binary classification problem.

In our experiments we use a standard maximum entropy classifier using the log-linear model as stated above. Moreover, state-of-the art toolboxes are available with efficient learning strategies. In this work, we apply the freely available toolbox Megam (Daumé, 2004), short for “MEGA Model Optimization Package”, which has a very efficient training procedure and for which the author reports good benchmark scores on its website.¹

Structured prediction

Tree alignment is a typical structured prediction problem. This means that this is a problem where we attempt to predict the values of multiple variables where there are constraints or dependencies among them. This is clear from the fact that link decisions influence each other. For example, the well-formedness principle (see below) implies that if two nodes are linked, their children must be somehow equivalent to each other, increasing their chances to be linked as well. Whether or not parents or children - or even grandparents and grandchildren - are linked are often valuable cues in deciding whether or not the currently inspected node pair should be linked as well.

Because of this, treating links in isolation as outlined above causes a lot of errors due to the dependencies between link decisions. Therefore, applying a binary classifier alone and linking according to the individual decisions is not a good idea. However, building a discriminative predictor for the entire structure is not feasible due to the sparsity of the data, since most tree structures are complicated structures with features that are often unique.

In our implementation, we opt for a *recurrent architecture* (Dietterich, 2002) using history-based features and a sequential classification process. Both a top-down and a bottom-up link classification procedure can be used.

The idea behind this strategy is simple: previous decisions of the global classifier are used as input features for current decisions. Training is simple, as link decisions are readily available and the classifier can learn directly using

¹<http://www.cs.utah.edu/~hal/megam/>

those features. In classification, we have to use a sequential setup in order to obtain history features. In the bottom-up strategy we start with predicting links for leaf nodes moving up towards the tree root. Here, we assign history features to be taken from the child nodes. Conversely, using a top-down strategy, history features are taken from the parent nodes.

Lingua-Align offers another way of incorporating structural dependencies in prediction, namely to use a simple greedy alignment strategy. In tree alignment, it is common to restrict the process to one link per node. This allows us to define a greedy best-first alignment strategy to account for competition between link candidates (Melamed, 2001). Further constraints can be applied to guide the alignment even further.

Additionally, certain well-formedness constraints as proposed by Hearne et al. (2007) can be used to guide the alignment:

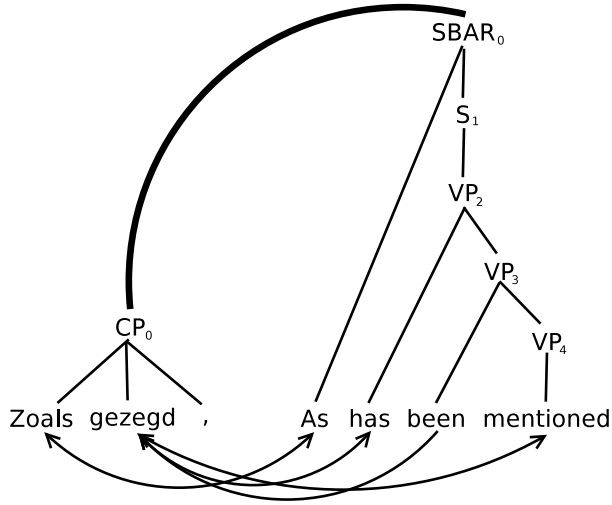
- A node in a tree may only be linked once.
- Descendants of a source-linked node may only be linked to descendants of its linked node on the target side and vice versa (target to source).
- Similarly, ancestors of a source-linked node may only be linked to ancestors of its linked node on the target side and vice versa (target to source).²

Our implementation allows for both the well-formedness constraints and the greedy best-first strategy to be applied. However, we do not enforce the one-to-one restriction among word alignments, since in the context of MT, it has generally been found to be too restrictive (see, for example, Tiedemann (2004)). In addition, we also introduce a constraint to restrict alignments in such a way that non-terminal nodes are aligned to non-terminal nodes and terminal nodes to terminal nodes only. This reduces the hypothesis space significantly and seems to be sufficient to represent translational equivalence.

Tiedemann (2010) describes in more detail how different alignment strategies can be combined. For example, one can apply directional alignment strategies similar to those used in word alignment (source to target or target to source) and use symmetrization heuristics such as calculating their intersection. Alignment can be split into steps such as doing either word or non-terminal alignment first and the other second, and/or removing links not conforming to well-formedness at either step.

²As mentioned in chapter 2, one side is often called the source side and the other the target side, even if eventual translation from the former to the latter is not the end application one has in mind.

Figure 4.1: Example tree alignment from a Dutch to English aligned version of Europarl, including word alignments produced by both the source-to-target and target-to-source phase of the GIZA++ word alignment phases, as indicated by the arrows. Dutch is the source and English the target side. The thick solid line links the roots of the subtrees implying equivalence. Note that no nodes dominated by the two roots are linked to nodes with different ancestors (i.e. in other trees), making this tree pair well-formed with respect to their links.



4.4 Alignment features

Figure 4.1 is a simplified example of the alignment of a well-formed tree pair from the Europarl parallel corpus (Koehn, 2005). We will use this example to illustrate some feature descriptions.

Lexical equivalence features

For use in the Dublin Aligner, Zhechev and Way (2008) introduce lexical probabilities to be used in unsupervised tree alignment, taken from GIZA++ probability tables. They are combined into an alignment score γ which is composed out of so-called *inside* scores ($\alpha(s_l|t_l)$, $\alpha(t_l|s_l)$,) and *outside* scores ($\alpha(\overline{s_l}|\overline{t_l})$, $\alpha(\overline{t_l}|\overline{s_l})$). Inside scores refer to scores derived from word alignment probabilities of the words dominated by the currently inspected non-terminals, and outside scores

refer to those that are derived from probabilities from words that are not dominated by these constituents. In Zhechev (2009), two main formulae are given for the calculation of these scores. In the first, for each source-side word, the target side word alignment probabilities are summed, after which the sums are multiplied with each other:

$$\alpha(y|x) = \prod_i \sum_j P(y_j|x_i)$$

Since it was found by Zhechev (2009) that the formula had some drawbacks, such as a bias towards short strings, in his work a second normalized formula was introduced, where the averages of the sums were multiplied instead:

$$\alpha(y|x) = \prod_j \frac{\sum_i P(y_j|x_i)}{|x|}$$

With Lingua-Align, use of both of these features is possible, as well as versions where null links are not considered, which speeds up feature extraction. We also introduce two new variants of the above formula which are slightly modified. For the first one, we select the maximum lexical score for each token instead of taking the sum or average of all possible word connections:

$$\alpha(x|y) = \prod_{i=1}^{|x|} \max_j P(x_i|y_j)$$

In our experiments this modification gave us a better performance. The main motivation for this is the idea that it is more important for words to be strongly aligned to exactly one other word instead of having many weaker alignments. The latter is often the result of having function words that can be aligned to almost anything, and therefore is not necessarily a strong indication of equivalence. In other words, a few strong links might be more important than a lot of weaker links.

Finally, we also introduce a version where we compute the average of the maximum scores instead of their product:

$$\alpha(x|y) = \frac{\sum_{i=1}^{|x|} \max_j P(x_i|y_j)}{|x|}$$

Word alignment features

Other features can be derived directly from existing word alignments. We define a feature measuring the proportion of *consistent* links among all *relevant* links l_{xy} involving either source s_x or target language words t_y dominated by the current tree nodes (s_i and t_j). Denoting dominance as $s_x \geq s_i$, this feature is defined as follows:

$$\begin{aligned}
 cratio(s_i, t_j) &= \sum_{l_{xy}} consistent(L_{xy}, s_i, t_j) / \sum_{l_{xy}} relevant(l_{xy}, s_i, t_j) \\
 consistent(L_{xy}, s_i, t_j) &= \begin{cases} 1 & \text{if } s_x \geq s_i \wedge t_y \geq t_j \\ 0 & \text{otherwise} \end{cases} \\
 relevant(L_{xy}, s_i, t_j) &= \begin{cases} 1 & \text{if } s_x \geq s_i \vee t_y \geq t_j \\ 0 & \text{otherwise} \end{cases}
 \end{aligned}$$

where *cratio* denote the proportion (ratio) of consistent links - links shared by both subtrees - among all relevant links - links shared by at least one subtree - as described above.

One may apply the Viterbi word alignments produced by GIZA++ (Och and Ney, 2003) using the IBM 4 model in both directions, as well as the symmetrized versions produced by Moses (Koehn et al., 2007), i.e. the union of these links and their intersection. These four scores can also be used as features by Lingua-Align. Consider the following example which displays the GIZA++ alignments for both directions of the phrase pair displayed in figure 4.1:

```

zoals gezegd ,
NULL ({ }) as ({ 1 }) has ({ 2 }) been ({ }) mentioned
({ 2 })

as has been mentioned
NULL ({ }) zoals ({ 1 }) gezegd ({ 2 3 4 }) , ({ })

```

Each word has a number associated with its position. The first word is at position 1, where a reserved *null* element is at position 0. The first two lines display the alignments made in the source-to-target phase, whereas the next two lines display the alignments made in the target-to-source phase. For example, the target word *as* is linked to the first source word *zoals*, and the source word *gezegd* is simultaneously linked to the 2nd and 4th words *has* and *mentioned* on the target side. However, note that these links were made in the

source	target	GIZA _{src2trg}	GIZA _{trg2src}
CP ₀	SBAR ₀	1	1
CP ₀	S ₁	2/3	1
CP ₀	VP ₂	2/3	1
CP ₀	VP ₃	1/3	1
CP ₀	VP ₄	1/3	null

Table 4.1: Example of word alignment consistency scores derived from GIZA++

source-to-target phase - in the target-to-source phase, *been* has also been linked to *gezegd*. The comma remains unlinked in both directions.

If we now apply the formula to the trees, we obtain the results presented in table 4.1. Note that we present the union of the GIZA++ alignments which may or may not reflect the final result depending on post-processing using Moses heuristics (see chapter 2 for more information).

We present the following example for clarification: CP₀ dominates the words *Zoals gezegd*, and VP₂ dominates *has been mentioned*. To calculate the source to target score, we note that there are three links that have been made in the source to target phase. Only two of them are between words that are dominated by both constituents, namely the links pointing to *has* and *mentioned*. Therefore, the source-to-target score is 2/3.

In the target-to-source phase, one link have been assigned, namely the one linking *been* to *gezegd*. This link is dominated by both constituents, therefore the target-to-source score is 1.

We can also combine the above GIZA++ scores and measure the union without taking direction into account. In that case, we simply look whether or not such a link is between terminal nodes dominated by both trees or just one.

Similarly, we can use the output of the Moses heuristics which can involve the union, intersect or other refinements to produce the final word alignments, and calculate the proportions as described above.

Finally, we define another word alignment feature for pairs of terminal nodes. It is simply a binary feature being set to one if and only if both nodes are linked in the underlying word alignment. This is useful if terminal node alignment is included in the tree alignment model as it is in our initial experiments.

Tree features

Next, we introduce a number of features that are independent of external tools and resources. Using the relative position of each node in the parse tree we define two features: tree-level similarity (*tls*) and tree span similarity (*tss*). For the former we use the distance to the root node ($d(s_i, s_{root})$ resp. $d(t_i, t_{root})$) and normalize this distance with respect to the size of the tree (maximum distance between any node and the root node). For the latter we compute the relative “horizontal” position of a node based on the span of the entire subtree which is rooted in that node. This position is then normalized by the length (number of leaf nodes) of the sentence. Furthermore, we define a *leafratio* feature to measure the difference in subtree spans.

We formulate the tree level similarity as follows:

$$tls(s_i, t_j) = 1 - abs\left(\frac{d(s_i, s_{root})}{\max_x d(s_x, s_{root})} - \frac{d(t_i, t_{root})}{\max_x d(t_x, t_{root})}\right)$$

where $d(s_i, s_{root})$ and $d(t_i, t_{root})$ denote the distances of the current source and target nodes to the roots of the respective sentences, and $\max_x d(s_x, s_{root})$ and $\max_x d(t_x, t_{root})$ denote the maximum distance from any terminal to these respective roots. Therefore, we obtain a normalized distance measure for both nodes, and subtract them from each other to obtain a measure of their difference. Finally, we subtract this score from 1 in order to obtain a tree level similarity measure of the two non-terminals in question, where 1 is the most and 0 the least similar.

The tree span similarity is formulated as follows:

$$tss(s_i, t_j) = 1 - abs\left(\frac{s_{start} + s_{end}}{2 * (length(S) - 1)} - \frac{t_{start} + t_{end}}{2 * (length(T) - 1)}\right)$$

where S and T are the entire sentences in which s_i and t_j occur, respectively. s_{start} and s_{end} denote the starting and end position - in other words, the spans - of the terminals that are dominated by s_i , and similarly for t_{start} and t_{end} in terms of t_j . To compare these spans to each other, we can look at their middle points - this is achieved by summing the start and end points and dividing by 2. For each side, this is then normalized by dividing the latter by the total length of the respective sentences. We subtract 1 from the length since we assume that the start of the sentence is at position 0.

Finally, we present the formulation of the leaf ratio score as follows:

$$leafratio(s_i, t_j) = \frac{\min(|leaf\ nodes(s_i)|, |leaf\ nodes(t_j)|)}{\max(|leaf\ nodes(s_i)|, |leaf\ nodes(t_j)|)}$$

source	target	tree-span	tree-level	leaf-ratio
CP_0	$SBAR_0$	1	1	3/4
CP_0	S_1	5/6	4/5	1
CP_0	VP_2	5/6	3/5	1
CP_0	VP_3	2/3	2/5	2/3
CP_0	VP_4	1/2	1/5	1/3

Table 4.2: Example of tree feature scores

This is simply the proportion of the number of leaves (terminal nodes) between those that are dominated by s_i and those that are dominated by t_j . For example, if s_i has 3 leaves and t_j has 4, the leaf ratio score is 3/4.

Similar to Table 4.1, we display the scores obtained when applying the formulas pertaining to the tree features to our example trees in table 4.2.

Finally, we also define features derived from the syntactic annotation. Intuitively, category labels (for non-terminal nodes) and part-of-speech labels should be valuable indicators for a possible link. For example, it is conceivable that an NP is more likely to be linked to another NP, as opposed to a VP. These features are simply binary features which are set to one if the particular label combination is present and zero otherwise. In the case of a linked NP pair, for example, we can write it as a function:

$$f(NP, NP) = 1$$

and similary, for the same pair:

$$f(NP, VP) = 0$$

Edges connecting nodes may also be labeled in order to indicate various dependencies or underlying syntactic structures. We introduce a similar feature as above, where a specific combination of edge labels between the current constituents and their immediate parents is assigned the value 1, and 0 if they do not exist.

Contextual features

So far we only considered features directly attached to the candidate nodes. However, tree nodes are connected with other nodes in the tree structure and

their alignment may as well depend on features of neighboring nodes. Therefore, features from surrounding nodes should be considered as well. Using the tree structure we can extract the same features as described above from other nodes connected to candidate nodes. For this we define the following functions that can be used to move within the tree when extracting features: *parent* - move to the immediate parent and take the feature values from this node; *child* - compute the average feature value for all child nodes; *sister* - compute the average feature value for all nodes with the same parent node. These functions can be applied recursively. For example, applying *parent* twice will force the feature extraction process to move to the grand-parent node (if this node exists). Additionally, one can be more specific with the *neighbor* feature, for which one can specify a specific sister on both the left and right hand side and specifying the horizontal distance from the current node. Note that these functions can be applied to either the source or target language tree or both.

In this way we have many possibilities to explore contextual features. Proper feature engineering is necessary to define useful templates.

Link dependency features

Here we refer to history features mentioned earlier in our discussion on structural prediction. For bottom-up classification, the following two variants are defined in our implementation:

- The *children_links* feature is the number of links between child nodes of the current node pair normalized by the maximum of the number of source language children and the number of target language children. In other words, we introduce the relative number of links between terminals dominated by the source and target side non-terminals as a discriminating feature.
- Similarly, the *subtree_links* feature is the number of links between nodes in each entire subtree dominated by the current nodes. This score is then normalized by the larger number of nodes in either the source subtree or the target subtree.

In classification, only the prediction likelihood is used for estimating these feature values. In other words, we use soft counts instead of counting actual links. Hence, classification can be done in a bottom-up fashion before applying the greedy best-first search in the final step.

Complex features

Some features may be correlated in a non-linear way. To account for those it is possible to create complex features. Any of the features above can be combined in such a way that they form a new feature function with their values combined. Possible combinations are:

- product (*): This multiplies the value of 2 or more feature types. For example, *inside2*outside2* would refer to the product of the inside normalized lexical equivalence scores with the outside ones.
- average (+): As above, but calculating the average (arithmetic mean) instead of the product.
- concatenation (.): Here, we merge 2 or more feature keys and compute the average of their scores. The idea is that for binary feature types such as POS tags (1 or 0), as opposed to real-valued feature types such as *inside* or *outside*, it might make more intuitive sense to call it concatenation. For example, *catpos.edge* concatenates the keys of the first labeled feature involving POS tags or category labels with the keys of the labeled edge feature (which is also binary) and computes the arithmetic mean of both scores. We effectively treat the binary-valued features here not as categorical, but rather as numerical.

Implementing complex features gives us a combinatorial explosion of possible features and careful feature engineering is again necessary to select valuable ones. Furthermore, complex features are even more exposed to sparseness problems. Nevertheless, various combinations lead to significant improvements in our experiments as we will see in our experiments.

4.5 Parameters

Apart from using various combinations of features as described above, some parameters can also be set in both the training and testing phase. Apart from those which specify the actual features, the specific files or directories and how many sentences are to be used in training and testing, the following are also possible:

- type of input alignment file: Currently, the options are *sta* (Stockholm TreeAligner format) and *opus* (CES XML format as it is used in the OPUS

corpus (Tiedemann, 2003)). The Stockholm TreeAligner is a manual viewer and editor of parallel treebanks in TIGER-XML format, which we used to build all our training data.

- type of output alignment file: Currently, the options are *sta* and *dublin* (format used in the Dublin tree aligner).
- type of treebank file: The currently supported types are TIGER-XML, Alpino-XML, Penn Treebank style (brackets with dependencies) and the type used by the Berkeley format (only brackets).
- swap alignment direction when reading the parallel treebank
- also align index nodes as used in Alpino-XML
- type of classifier: Currently, the options are *megam*, as mentioned above, and *clue*, which, as the documentation states, refers to a noisy-or like classifier with independent precision-weighted features. This requires probabilistic values for each feature and supports only positive features.
- switching on the linked children and/or the linked subtree nodes features (see section 4.4)
- using a specified number of iterations for adaptive SEARN (Search and Learn) style learning. This is a technique introduced by Daumé III (2006) in order to deal with the so-called label bias problem, discussed by Van den Bosch (1997). In our experiments, we have not yet experienced significant improvements by using this option. The current implementation is quite slow as well, although this might be improved in the future.
- aligning terminal nodes only, or non-terminal nodes only
- specifying training weights for good, fuzzy or negative examples (non-aligned nodes)
- keeping the extracted feature file which is created during training and usually removed afterwards
- specifying the score threshold above which an alignment is made
- specifying the alignment strategy. Currently, the options are *inference*, which refers to a two-step procedure with local classification in the first step and alignment inference in the second (see below); and *bottom-up*,

where we align in a greedy bottom-up fashion starting at the terminal node level and going up to the root nodes. It is greedy in the sense that nodes are linked immediately when the classification score improves upon the set threshold, after which aligned nodes are removed from the search space.

- specifying *add-links mode*. In this mode, already existing links will be kept in the output file and may not be removed.
- specifying that we now use the scores of existing links which will compete with the new links. This also implies that existing links may disappear to conform to well-formedness.

4.6 First experiments

During the development of Lingua-Align we used the SMULTRON treebank (Gustafson-Čapková et al., 2007) for evaluating the aligner’s performance with various strategies and feature sets. SMULTRON includes two trilingual parallel treebanks in English, Swedish and German. The corpus contains the alignment of English-Swedish and German-Swedish phrase structure trees from the first two chapters of the novel “Sophie’s World” by Jostein Gaarder and from economical texts taken from three different sources. The alignment has been done manually using the Stockholm Tree Aligner (Lundborg et al., 2007).

The alignment includes so-called *good* and *fuzzy* links, which refer to confident and less confident links respectively. We will use both but give them different weights in training: Good alignments get three times the weight of fuzzy and negative examples (good: 3, fuzzy: 1, negative: 1). Negative examples are instances of node pairs which are not linked, the feature values of which are also extracted for learning in the training phase.

We mainly worked with the English-Swedish treebank of Sophie’s World which includes just over 500 sentences per language (528 for English and 536 for Swedish, including 6,671 good links and 1,141 fuzzy links). In our experiments we used the first 100 aligned parse trees for training and the remaining 418 sentence pairs for testing, in order to obtain more reliable results. Later on, we also present the results of a training curve, where more training data was used.

For evaluation we use the standard measures of precision, recall and F-scores as they are used in word alignment evaluation. Due to the distinction between good and fuzzy alignments we compute values similar to word alignment

evaluation scores in which *sure* (good) and *possible* (both good and fuzzy) links are considered:

$$Precision(A, P) = \frac{|P \cap A|}{|A|} \quad Recall(A, S) = \frac{|S \cap A|}{|S|} \quad F = \frac{2 * Precision * Recall}{Precision + Recall}$$

S refers here to the good alignments in the gold standard and P refers to the possible alignments which includes both, good and fuzzy. A are the links proposed by the system. We will only use a balanced F-score with $\alpha = 0.5$. We also omit alignment error rates due to the discussion about this measure in the word alignment literature (Fraser and Marcu, 2007). Note that the proportion of fuzzy links is reasonable and we do not expect severe consequences on our evaluation as discussed in Fraser and Marcu (2007) for word alignment experiments with more unbalanced gold standards.

The upper part of table 4.3 summarizes the results for different feature sets when running the aligner on our development corpus. We also trained the Dublin aligner (Zhechev and Way, 2008) on the same data and include the results as a baseline.

Settings	Precision	Recall	$F_{\alpha=0.5}$
Dublin aligner	57.15	55.66	57.57
lexical features	59.76	41.81	49.20
+ tree features	49.40	57.25	53.03
+ alignment features	57.18	60.58	58.83
+ label features	77.17	76.10	76.63
+ align-context features	78.12	78.42	78.27
train=novel, test=economy	77.39	73.50	75.39
train=economy, test=novel	76.66	74.62	75.62

Table 4.3: Results for different feature sets (top) and textual domains (bottom), using the results of the Dublin aligner as a baseline.

The alignment results are very promising. We can see that adding features consistently helps to improve the performance. The advantage of a discriminative approach with a rich feature set can be seen when comparing our results with the performance of an unsupervised tree aligner. Running the Dublin aligner on the same data set yields a balanced F-score of 57.57%. The low score is

most probably due to the sparse amount of training data for estimating lexical probabilities, on which the Dublin aligner relies much more than Lingua-Align. Therefore, we proceeded to train a new word alignment model based on both the SMULTRON data and a sentence aligned version of the Europarl 3 parallel corpus. However, the performance improves only slightly to about 58.64%.³ This is probably the effect of using data from a very different domain.

We can see that the additional features and the optimization of their contributions through machine learning has a strong positive effect on the performance, to such a degree that using only 100 sentence pairs already results in F-scores of close to 80. We speculate that much of this is thanks to the existence of good word alignments - one of the most important features as shown in later chapters - which serve to disambiguate many candidates, as subtrees sharing word alignments tend to be equivalent. Applying the well-formedness constraint filters out more candidates, increasing precision. Other features such as relative subtree sizes and label pairs also seem to be very telling.

A drawback of supervised techniques is that we have to drop the generality of the unsupervised approach and require aligned training data to build language pair specific models. Furthermore, there is a risk of overfitting. In order to test the flexibility of our approach we ran several cross-domain experiments using the two domains present in the SMULTRON treebank. The results are presented in the lower part of table 4.3.

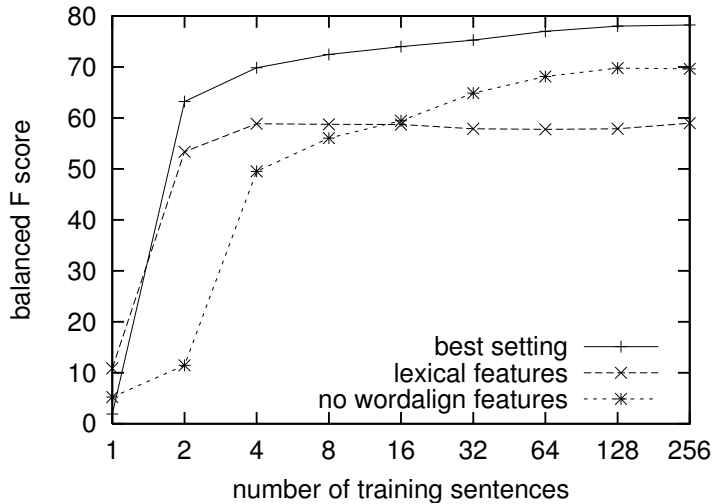
As we can see, there is only a slight drop in performance when training on a different textual domain. However, we still have reasonably high accuracy which is certainly encouraging, especially considering the relatively little effort of human annotation necessary when preparing appropriate training data.

Finally, we also looked at the training curves with varying amounts of training data. For this we used about a third of the corpus (2667 links) for testing and trained on parts of the remaining data. Figure 4.2 shows the impact of training size on F-scores for three feature settings.

We can clearly see that the aligner yields a high performance already with little amounts of training data. The training curve already levels out at training sizes way below 100 sentence pairs. This is especially apparent for the settings that include word alignment features. Their values bear a lot of positive link evidence on their own and corresponding weights do not need to be adjusted very much. Other features such as the binary label-pairs require larger amounts

³Suprisingly, this is significantly better than our discriminative approach when using the same lexical features only (inside/outside scores) which is probably due to the alignment search optimization in the subtree aligner.

Figure 4.2: Training curve for various amounts of training data from the SMULTRON parallel treebank.



of training examples. However, the alignment performance does not seem to improve significantly after about 128 sentence pairs, even when those features are included (see, for example, “no wordalign features” in figure 4.2).

4.7 Experiments for PaCo-MT

The development of Lingua-Align took place in the context of the PaCo-MT project, which is described in detail in chapter 3. Because of its promising results as described in the previous section, we decided to align a selection of parallel treebanks manually in order to train an alignment model. This we used to construct aligned parallel treebanks on a large scale for training the MT system.

Our data source is a large selection of multilingual corpora from various domains. In order to construct this training data, the data have to be cleaned, sentence aligned, tokenized, word aligned and parsed. The word alignments were a combination of the intersection of the source-to-target and target-to-source GIZA++ alignments (confident alignments) as well as the output of the *grow-diag* heuristic (less confident alignments, see chapter 3).

Data set	Word al.	Sent. pairs	Precision	Recall	F-score
Dutch to English	auto	140	78.3	68.3	76.3
Dutch to English	manual	140	78.9	68	75.9
English to Dutch	manual	150	74.4	72.9	75.5
Dutch to French	auto	158	66.1	67.6	67
French to Dutch	auto	158	68	70.5	69.4

Table 4.4: Ten-fold cross validation scores of the training data sets used in the PaCo-MT project. The entries under *Word al.* (word alignment) refers to whether or not the word alignments of that particular data set have been manually corrected. *Sent. pairs* refers to the number of sentence pairs.

The languages involved were Dutch/English, and Dutch/French. A separate translation model is built for each direction. Therefore, we needed to construct four sets of parallel treebanks. For constructing the manual alignments, we utilized the Stockholm TreeAligner (Lundborg et al., 2007). After the abovementioned pre-processing, a selection of sentence pairs was chosen and manually aligned. For the initial experiments in PaCo-MT using Dutch and English, we also corrected word alignments. In the latter part of the project where we also applied rules and constructed new data sets, we did not change the automatic word alignments, as we believed it better to simulate the input conditions as much as possible, while word alignments are quite accurate anyway. In addition, manual word alignment is also very time consuming. The word alignments of the Dutch/French sets were not corrected.

For inclusion, sentences had to be correctly sentence aligned. Some of the parse trees were quite different from each other, restricting the number of alignments that could be made between the constituents. We decided to include these as well, since they provide valuable evidence in terms of negative alignment examples for the training phase. All of our data were extracted from the Europarl corpus.

Eventually, we constructed four different models for each of the language directions. Table 4.4 displays the resulting scores after applying ten-fold cross validation to each data set, training on 90% and testing on 10% of the data. Note that the number of sentence pairs differs because of the fact that some of them were discarded because of the occurrence of sentences that are wrongly aligned, or too short, or duplicated elsewhere, and so on. We did not manage to recover the English-to-Dutch set containing the automatic word alignments.

Data set	Word al.	Linked/unlinked T	Linked/unlinked NT
DUT/ENG	auto	0.73	0.51
DUT/ENG	manual	0.8	0.51
ENG/DUT	manual	0.82	0.48
DUT/FRE	auto	0.82	0.41
FRE/DUT	auto	0.82	0.41

Table 4.5: Ratios of linked terminal vs. total terminal count (T) and similarly for non-terminals (NT) for all training data sets used in PaCo-MT.

It is clear from the table that Dutch/English models fared reasonably well. On the other hand, French models did significantly worse, even after extensive feature engineering. One possible explanation that we have proposed for this is that the French trees do not contain as much labeled data as the English trees, where the latter also contained edge labels denoting grammatical functions such as subject and object. However, in our experiments, the edge feature did not make that much of a difference in evaluation output. Rather, we have noted that there were relatively many unalignable nodes during the construction of the training data because of differences in tree structure. Most probably, we needed more training data to learn more reliable feature values from positive examples.

Table 4.5 illustrates this problem more clearly by displaying the ratios between linked nodes and unlinked nodes for the PaCo-MT training data sets, divided between terminals and non-terminals. The high ratio between linked and unlinked terminals for all sets is a clear indication of the high recall approach. However, the non-terminal ratios of the Dutch/French sets are noticeably lower. Note that we have actually used the same sentence pairs for both directions (Dutch to French and French to Dutch) but they are still word aligned differently due to the asymmetric directional approach of GIZA++.

In the latter stages of the PaCo-MT project, we constructed more training data for various other experiments. They are described in more detail in chapters 6 and 7. There were also a number of stages in the project during which important changes were made to the system and different alignment sets were constructed. This is discussed in detail in chapter 3.

4.8 Error analysis

Here we present a few examples of errors in the output of the Lingua-Align models for PaCo-MT. Although a clear error analysis of the output of statistical systems is often quite difficult, we believe that some insight can still be gleaned in view of the features and parameters that we have specified in the training phase.

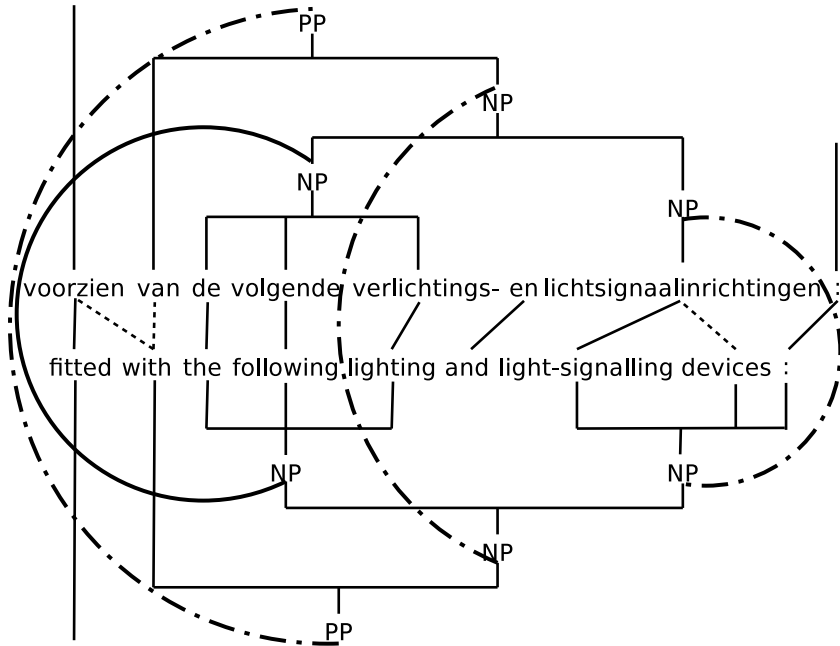
Since the models are trained on Europarl data, we are more interested in the output from other domains, since our experiments in section 4.6 suggest that they are likely to have more errors. We think that our first example in figure 4.3, taken from the Dutch to English DGT translation memory corpus, illustrates the recall problem as well as the problem with using strict well-formedness as a constraint. The full sentence pair is:

- Iedere tweewielige motorfiets mag bovendien zijn voorzien van de volgende verlichtings- en lichtsignaalinrichtingen:
- In addition all two-wheel motorcycles may be fitted with the following lighting and light-signalling devices:

The following problems are apparent:

- The NP containing the word *lichtsignaalinrichtingen* should be aligned with the NP dominating the words *light-signalling devices*:. Technically they are not the same because of the existence of the colon on the English side. This seems quite unusual, but punctuation is in fact systematically attached to lower trees in the version of the Stanford parser that we have used. In a high-recall approach where punctuation is handled in a separate step or ignored such as with PaCo-MT, aligning such trees would be desirable. Currently, Lingua-Align only sees a good word alignment that links a dominated token (English colon) with a Dutch token that does not have the other NP as an ancestor. The trees are therefore not well-formed so they are not linked.
- Similarly, the parents of the two NPs just mentioned should have been linked as well. Almost every descendent on both sides is linked confidently but the existence of the outgoing link from the colon violates the well-formedness constraint.
- Once again, the next parent pair (two PPs) should have been linked. In this case, there is also a fuzzy link aligning *voorzien* with *with*. Since

Figure 4.3: An example of false negatives in Lingua-Align output containing Dutch to English alignments. The Dutch colon, as well as the words *voorzien* and *fitted*, are linked to higher up constituents that are not displayed in this figure.



voorzien is not a descendant of the Dutch PP, the existence of this link also violates the well-formedness constraint.

Eventually, many good alignments are thrown out in this way, driving the recall score down. It is possible to set a parameter so that Lingua-Align ignores the well-formedness constraints after all good well-formed candidates have been linked. However, in our experience this has not led to better evaluation scores. It does seem reasonable to allow certain cases of non-wellformedness but not others, for example in cases like in the figure, or where only fuzzy links are concerned, or only a certain number of violating links. This is certainly possible to implement in Lingua-Align as features but has not yet been done. Our rule-based experiments in chapter 6 suggest that such distinctions carry some merit.

We also apply them as features in our transformation-based learning system described in chapter 7.

Using Lingua-Align, one can also allow the removal of alignments between words that lead to non-wellformedness between candidate subtree pairs in the alignment phase, depending on the linking strategy. However, we believe that in the context of a high-recall alignment approach, although not proven, this may detract from the ultimate goal of better MT quality.

An example from computer manual data in the French to Dutch parallel corpora shows an instance of a false positive (figure 4.4) with two other false negatives. The full sentences are:

- Cliquer sur un item dans cette application affiche alors une fenêtre d'aide (si celle-ci existe pour cet item particulier), expliquant la fonction de l'élément en question.
- Het klikken op items in dit programma zal een helpvenster openen (als er een bestaat voor dit specifieke item) met een omschrijving van de functie van dit item.

A possible English translation would be:

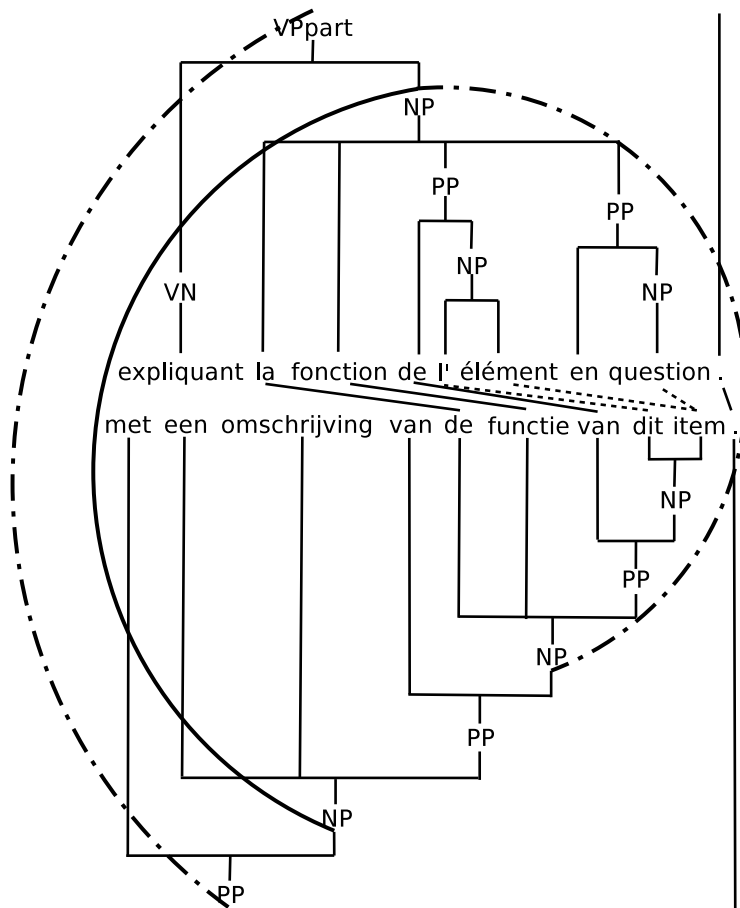
Clicking on items in this application will open a help window (if one exists for the particular item) explaining the function of this item.

In the figure, the NP dominating the phrase *la fonction de l'élément en question* (the function of this item) is erroneously linked to the NP dominating the phrase *een omschrijving van de functie van dit item* (literally: a description of the function of this item). Instead, the French NP should be linked to the NP dominating *de functie van dit item* (the function of this item).

So what went wrong? First of all, there are some false negatives among the word alignments as well. The French verb *expliquant* should be linked to the noun *omschrijving*, which is the closest equivalent, with possible fuzzy links to *met*, *een* and *van*. If this were the case, the NPs in question would almost certainly not have been linked, since *expliquant* does not have the NP as an ancestor and therefore the linked subtrees would not have been well-formed.

Secondly, note that the spans of the incorrectly linked subtrees are very similar, both with regard to each other (eight leaves each) as well as with regard to the whole sentence (both ending one position before the final sentence token). On the other hand, the spans of the correct subtrees are less similar (eight leaves versus five leaves). This span difference is partly due to the French *en question*

Figure 4.4: An examples of both false and positive negatives in Lingua-Align output containing French to Dutch alignments. Both periods are connected to higher up constituents which are not shown in this figure. Solid word alignments are good and dashed ones are fuzzy. The dashed non-terminal alignments indicate false negatives (should have been aligned) and the solid curve connecting the two NPs is a false positive.



(*in question*) which is not explicitly present on the Dutch side. It seems that this feature must have played a more important role than, for example, the leaf ratio, which has a higher score when applied to the leaves dominated by the correct subtree pair.

Having somehow missed the lower Dutch NP (*de functie van dit item*) as the best target side candidate, the classifier seems to have proceeded, with a lack of proper word alignment evidence, to utilize structural features such as the tree spans and labeled data (choosing the French NP over, for example, the VPpart) to guess the most likely choice which in this case, proved to be wrong.

Since *expliquant* is roughly equivalent to *which describes*, which can be roughly translated to *with a description of*, which is the literal translation for the Dutch *met een omschrijving van*, aligning the French VPpart to the Dutch PP seems quite reasonable. However, here these nodes stay unaligned.

It is also apparent how many non-terminal nodes cannot be linked. The phrase *l'élément en question* can be aligned to *dit item*, but there is no subtree root governing this specific French phrase. The case is similar with *de l'élément en question* which is translatable to *van dit item*. The Dutch NP dominating *een omschrijving van de functie van dit item* has also no equivalent. In our experience, the lack of convergence was very apparent while building our training data, which is exemplified by the previously discussed Table 4.5.

The most obvious solution to these problems is to include more training data. In the case of Dutch to English, we have constructed a larger Europarl data set for the experiments described in chapters 6 and 7. Indeed, F-scores generally increase in this case. However, while precision can be very high, recall tends to stay low. We believe that this is mainly because of the well-formedness restriction. As mentioned before, we discuss how we go about this problem while describing our rule-based experiments in chapter 6. In both our examples it was apparent how even slightly wrong word alignments can lead to wrong alignment decisions on the constituent level as well. We believe that relaxing well-formedness requirements would alleviate this problem at least to some degree.

Related to the last point, different word alignment strategies might also prove effective. We have not yet had the opportunity to experiment with other strategies or systems. However, it does remain important to use as much parallel data as possible and also make sure that the data is properly preprocessed.

4.9 Conclusion

We have presented a discriminative tree aligner that can be trained on small amounts of hand-aligned training data using a rich feature set. With this tool we achieve satisfying results for the problem of tree-to-tree alignment that can be used to align parallel treebanks on a larger scale. However, it was apparent that the strength of alignment is sensitive to both the quality of word alignment and to how convergent the source and target trees are. Convergence was especially a problem for the alignment of Dutch/French data.

For a more in-depth error analysis, a quantitative study is needed. In this regard, we present a statistical study in the next chapter on the impact of various different features on alignment performance.

A statistical study on the impact of different features on tree alignment performance in Lingua-Align

5.1 Introduction

In the previous chapter, we have introduced Lingua-Align, a supervised statistical tree-to-tree aligner implementing log-linear models. As part of our error analysis, we seek to better understand the role that different features play in eventual alignment performance. To achieve this, we extract various feature values from an alignment data set and conduct a multiple regression analysis by fitting a linear model showing which features have the strongest and most significant influence on F-scores. Other than with Lingua-Align, where the features are extracted with respect to a specific node pair, we are interested in the effect of global features on alignment performance, such as the ratios of sentence lengths, the number of good or fuzzy links, the ratio of good against fuzzy links for terminals and non-terminals, differences in the depths of trees, et cetera. This enables us to draw more general conclusions about the effects of alignment strategies, tree structures, node counts, and so on.

In the next section, we present the experimental setup. After that, in section 5.3, we present and discuss our statistical data. Finally, in section 5.4, we present our conclusion.

Precision	Recall	F-score (all)	F-score (recall_good)
78.63	68.94	73.43	76.46

Table 5.1: Average precision, recall and both types of balanced F-score of the results of the ten-fold cross validation performed on the Dutch-to-English alignment set.

5.2 Approach

As mentioned in the previous chapter, testing the tree aligner requires a data set consisting of sentence pairs that are translationally equivalent, syntactically parsed and that are also word aligned. In this work, we mostly focus on improving our Dutch to English parallel data. For this study, we use the same selection of 140 Dutch-to-English sentence pairs from the Europarl 3 corpus (Koehn, 2005) as described in the previous chapter, using automatic word alignments. Again, we make a distinction between good and fuzzy links, reflecting the level of confidence of the link.

We pre-processed the manually produced data set by applying ten-fold cross validation and then evaluated the accuracy of the automatically produced non-terminal node links with their respective gold standards. Figure 5.1 displays the average result.

Next, we extract a set of basic statistics. Per sentence pair, they are:

- based on all links with reference to the gold standard, the alignment precision, recall and both types of balanced F-score
- node counts (terminals and non-terminals)
- link counts (good and fuzzy, terminals and non-terminals)
- sentence lengths and ratios
- tree heights and ratios
- averages of tree level and sentence length ratios
- average height of terminal nodes to the root node
- standard deviation of these heights

For each sentence tree, we further assign a score based on its parse quality using manual inspection. The scores are on a scale of 1 to 3, where 1 is a good parse, 2 is not so good but reasonable, and 3 is a bad parse.¹

We define the *maximum height*, or just height, of the tree as the longest path found between a terminal and the root. Assuming that a greater difference between tree heights as well as a greater difference between sentence lengths may lead to lower F-scores, we also take the average of these ratios as a feature. We also look at the average path lengths of terminal nodes to the roots. If the average is close to the maximum height of the whole tree, this suggests that the structure is evenly spread, perhaps making it easier for nodes to be linked, especially if the same is true of the other side. The distribution of terminal node heights can also be described by calculating their standard deviation, which we also used as a feature.

Ratios are calculated by dividing the minimum value by the maximum value. They are further normalized to be in the same scale as the F-scores for further experiments. For example, if a Dutch tree sentence has a length of 10 tokens and the English tree a length of 12, the sentence length ratio would be 0.83. The normalized ratio would then be 83.

Not considering evaluation scores, a total of 73 features are extracted.

After feature extraction, we have for each sentence tree pair, and in some cases for each sentence tree, a set of data values whose significance with alignment evaluation scores we can investigate. With these values, we can produce distributions of the different variables over the whole set of sentence pairs.

In the next section, we present the distribution of the scores and the results of the statistical analysis, with a discussion of our findings.

5.3 Presentation and discussion of statistical data and analysis

Figure 5.1 presents a diagram representing the distribution of F-scores for all sentence pairs as produced by Lingua-Align. We use the F-scores pertaining to precision and recall for all non-terminal node links involved as a measure, and also present those measures in the diagram. It is clear that alignment accuracy can vary quite extensively. Roughly the first half of the F-score line tends towards a logarithmic curve. It is also interesting to note that there is much more variation in precision than in recall, while it is clear that precision regularly outperforms recall.

¹Many thanks to Prof. Gertjan van Noord for checking the quality of the Dutch trees.

Figure 5.1: Distribution of F-score, precision and recall of non-terminal alignments.

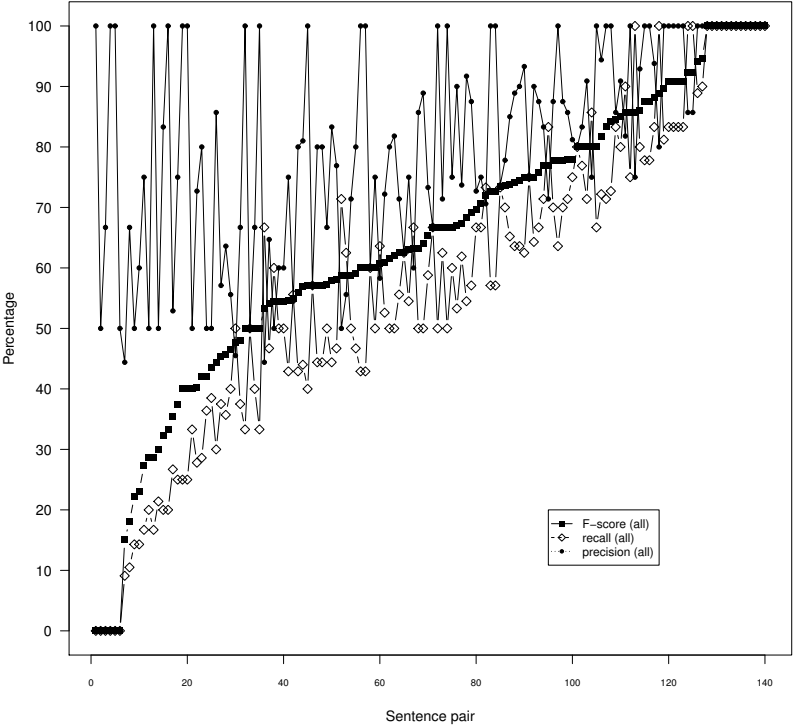
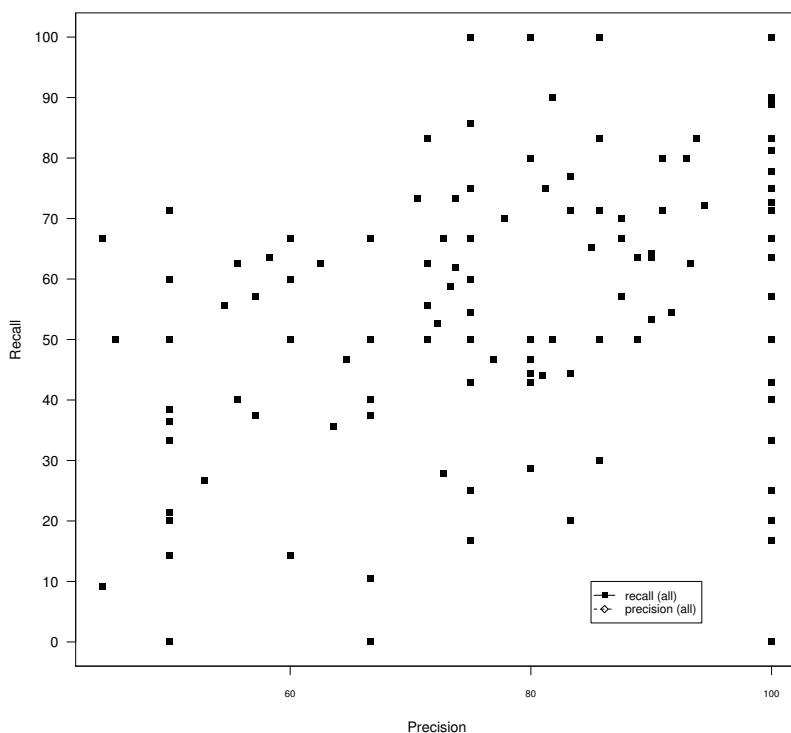


Figure 5.2: Scatterplot of precision against recall per sentence pair.



Plotting precision against recall suggests only a mild correlation, with much variation, as figure 5.2 shows. As is clear from the previous figure, precision tends to be high and recall tends to be low.

The second F-score distribution, where the recall of only the good alignments is taken into account, is very similar and we will not discuss this here.

Using the feature values that we have extracted, we proceed to conduct a multiple regression analysis, intending to find the strongest features while taking the degree of this influence into account.

We use the R statistics software package (R Core Team, 2012)² to fit a linear model using all features, but excluding precision, recall and the second F-score, as we are only interested in the first F-score. During the process, 25 features are dropped because of singularities, leaving us with 48. We achieve a surprisingly good fit, with a reported multiple R-squared value of 0.8315 and a p value of less than 10^{-15} . Furthermore, a handful of features is reported as very significant: Two of them have a p value of less than 0.01, two less than 0.05 and four more are marginally significant with p values of less than 0.1 but more than 0.05.

Table 5.2 displays these values, along with the coefficients indicating the strength of influence and the standard error. We also include the Pearson's correlation coefficients for the features which indicate to which degree they individually correlate with F-score.

We note that the four significant features strongly overlap with each other. For example, the number of terminal nodes is surely a strong contributing factor to the total number of nodes. However, as a whole one can conclude that node counts, and especially terminal node counts, significantly influence F-score given the other features included in the model.

The top feature has the lowest p value by far. It has a negative coefficient, meaning that more terminal nodes lead to worse F-scores given the features in the linear model. It also correlates negatively on an individual basis. From personal experience, longer sentences seem to be more likely to deviate translationally, leading to more fuzzy and/or incorrect word alignments, which may in turn lead to a lower quality of non-terminal alignment. So the values attributed to this feature make intuitive sense.

The next most significant feature is more interesting. It still has a negative correlation, but a positive coefficient. In the context of the model, but not individually, longer source-tree sentences lead to better F-scores. This is parallel to the fact that a larger number of nodes in the source tree tends to lead to worse scores (fourth feature), but conversely, "number of nodes" is positive.

These complications are probably due to the fact that the features compensate for each other to some degree. However, it remains difficult to explain.

The fact that terminal node counts (sentence length) seem to be important, as well as the focus on only one side (source tree), suggest that sentence lengths as well as length differences are significant. As discussed above, sentence length may have important effects on the quality of word alignment, which in turn may influence non-terminal alignment to a significant degree. This is further hinted at by the fact that the ratio of good terminal links is a marginally significant

²<http://www.R-project.org>

Feature	p value	Coefficient	Standard error	Individual corr.
Number of terminal nodes	0.00485	-7.09192	2.45503	-0.12755
Number of terminal nodes in source tree	0.00847	12.48369	4.63761	-0.11231
Number of nodes	0.02002	2.86113	1.20815	-0.12757
Number of nodes in source tree	0.03598	-5.08683	2.38929	-0.10856
Ratio of good terminal links against all terminal links	0.05615	-2.87053	1.48362	0.31482
Standard deviation of distances of terminal nodes in target tree to root	0.05966	35.86178	18.80121	-0.20123
Average of the standard deviations of source and target tree distances of terminal nodes to their roots	0.06609	-69.22080	37.20699	-0.16076
Standard deviation of distances of terminal nodes in source tree to root	0.09867	31.95879	19.15297	-0.06343

Table 5.2: Statistics displaying the most significant features as determined by a multiple regression analysis, taking all extracted features into account. The features with their p values in bold we regard as significant - the rest are marginally significant. *corr.* denotes *correlation*.

feature. The third and fourth highest correlations that are also in the regression model (Table 5.3) also happen to be the ratio of fuzzy links and all links (-0.42) and the ratio of good links and all links (0.42).

The length of sentence pairs negatively influences alignment accuracy so that longer sentences are aligned less accurately. The fact that source sentence length in contrast positively affects F-scores corrects the general effect. This may indicate that the negative influence of the sentence pair length is mitigated in those cases where it is necessary due to a long source language sentence. The latter influence F-scores positively.

The rest of the marginally significant features are all related to the standard deviations of the distances of terminal nodes to the tree roots. First of all, the coefficients are large because the standard deviations are relatively large. It is therefore not clear to what extent they influence performance. Individual correlations are consistently negative. In other words, the more variation there is between the distances, possibly indicating a more complicated tree structure, the lower the F-score. Although this is interesting, one should not read too much from it, firstly because these are just correlations, secondly because the coefficients associated with the individual trees are positive, and finally because these features are not that significant.

For the sake of interest, we also present in Table 5.3 the top individual correlations that are also in the regression model. We also list their p values and coefficients from the multiple regression analysis for comparative purposes.

A few of the top correlated features have been dropped in the model, and are not displayed here. None of the features in the table are statistically significant. Only one is marginally significant, which is the ratio of the number of good terminal links and all terminal links, presented in bold. Even here, the correlation appears to contradict the coefficient, where the former is positive and the latter negative.

The fact that these variables have strong correlations with F-scores but negligible, mostly very insignificant p-values means that their potentially predictive value has been realized by collinear predictors (in the multiple regression model). We include them in order to provide more insight into the question of which variables might be useful candidates for future work. It might turn out that some of these variables would surpass those in the current regression model if they might be improved in accuracy, for example.

In fact, 6 out of the 10 correlations are on different sides of the number 0 than the corresponding coefficients in the regression model. This, as well as the fact that there is very little overlap with the most significant features in the model, make it clear that the features influence each other to a great extent and

Feature	Correlation coeff.	Mult. reg. Coeff.	Reg. p value
ratio: number of linked non-terminal nodes and all non-terminal nodes	0.52201	0.24013	0.88887
ratio: number of linked non-terminals in the source tree and all non-terminal nodes in the source tree	0.457970	-0.16336	0.84247
ratio: number of fuzzy links and all links	-0.42123	1.39882	0.76768
ratio: number of good links and all links	0.42052	4.91613	0.29204
ratio: number of linked non-terminal nodes in the target tree and all non-terminal nodes in the target tree	0.40347	1.09138	0.13472
tree span similarity	0.35287	29.22232	0.12004
average of tree span and tree level similarity	0.32909	-58.08508	0.12263
ratio: number of good non-terminal links and all non-terminal links	0.32170	-0.35884	0.43665
ratio: number of good terminal links and all terminal links	0.31482	-2.87053	0.05615
ratio: number of linked non-terminal nodes in source tree and all linked nodes in source tree	0.23141	-5.37038	0.87868

Table 5.3: Top ten individually correlated features that are also in the multiple regression model, where *coeff.* denotes *coefficient*, *reg.* denotes *regression* and *Mult.* denotes *multiple*. None of the features have p values of less than 0.05, with only one marginally significant feature included, presented in bold.

that individual correlation by no means tells the whole story. Still, it is worth noting that many of the top correlations are ratios of links or linked nodes, as well as link types. The coefficients from the model suggest that they do have an influence, but the p values suggest that this might also be just by chance.

Finally, the scores that were assigned to parse trees did not feature in any of our tables. Perhaps parse tree quality is not as important as much as improved convergence of the trees involved, even if the parses are inaccurate.

5.4 Conclusion and future work

We have presented a statistical study consisting of determining the effect of a selection of features on the performance of an automatic tree aligner, given a reasonably good alignment model and reasonably good automatic word alignments. The most significant features suggest that node counts, but especially terminal node counts, have important influences on F-scores. Among these features, source trees also receive special attention, suggesting that there are important differences between the source and target sides.

From the list of marginally significant features, it would seem that the ratio of good terminal links against all terminal links may be important. It might be that, once we have accepted the general influence of length and tree size, that the more good word alignments there are in general, the more likely it is that the well-formedness constraint will lead to constituents staying unlinked in the case of a wrong word alignment. Flexibility with regard to less-than-perfect word alignments is an issue that we will investigate in the next chapter.

The other marginally significant features indirectly report the effect of tree structure. Interestingly, the more variation there is between the distances of terminal nodes to the root, the better the F-score.

The top correlations suggest that link ratios should have important effects, with good links leading to better scores and fuzzy links to worse scores. However, the results of our multiple regression model suggests that much of this may be due to chance, or simply wrong. Instead, node counts seem to be by far the most important and significant feature group.

For future quantitative evaluation experiments, more features could be extracted. In this study, we have mostly focused on counts and ratios at sentence level, but link-centered features describing the typical contexts of good and bad links may provide more insight.

As always, more data is always better, and using a second data set from a different domain may help strengthen or disprove any findings that resulted

from the first data set. Additionally, using different alignment models and even different tree aligners may provide more robustness to any future conclusions that we may draw.

In the next chapter, we will venture into new territory and explore the use of rule-based heuristics for the improvement of tree alignment, using the knowledge that we have gained.

Experiments in rule-based alignment error correction using manually constructed rules

6.1 Introduction

In this chapter, we introduce rule-based error correction as a viable complement to the statistical alignment process.

First, we present some experiments showing that even a single manually crafted rule can add reasonably accurate non-terminal alignments and also improve the total accuracy of already existing statistically produced alignments. We also experiment with strict and more relaxed versions of the rule and of the well-formedness constraint (see chapter 2), as well as different implementations of directionality. Finally, we implement a single complex rule based on our findings, applying it in a bottom-up fashion which significantly increases alignment recall.

All algorithms described in this chapter, unless otherwise specified, are applied to the output of a high-precision model trained by *Lingua-Align*. We therefore cannot comment on their effectiveness on the output of other models or aligners, although we remain positive that as a measure to increase recall they will prove effective, given the existence of quality word alignments.

6.2 Simple rules

We presented a statistical error analysis of Lingua-Align output in chapter 5 showing which features correlate the strongest with F-score evaluation. The linear model which resulted from our multiple regression analysis showed that differences in sentence length are very significant. As for other feature types, the results were sometimes difficult to interpret because of colinearity. However, some link ratios and link types had strong individual correlations. There was especially a clear distinction between good and fuzzy links. More good links have a relatively strong correlation with higher scores and more fuzzy links with lower scores. This is valid for both non-terminal and terminal alignments.

Although the impact of link ratios and link type ratios was not so clear from the regression model, it makes logical sense that more links should lead to a higher recall. Since this is a problem with the model that we have investigated, increasing it is one of our main goals.

Attempts to increase recall using Lingua-Align on this particular data set have so far led to decreased F-scores. In general, our best results on other data sets also tend to have higher precision and lower recall evaluation scores.

The importance of accurate word alignments in constituent alignment is underlined by our qualitative analysis in chapter 4 of the Lingua-Align system, the sharp drop in alignment performance when Moses and GIZA++ features are removed, as well as the fact that other alignment systems depend quite heavily on it (Groves et al., 2004; Lavie et al., 2008; Zhechev, 2009). In this chapter, we show that measures to decrease the sensitivity to less-than-perfect word alignment not only leads to a high recall but to better F-scores as well.

One of our goals is to measure the extent to which we can use rules to complement the statistical alignment process, not only to increase recall, but general accuracy as well. In that respect we deviate from Groves et al. (2004), who opt for a high-precision approach and do not apply statistical alignment as a first step. In the light of this, we proceed to design and implement simple rule-based experiments. Our additional aim is to determine whether or not high-recall alignments lead to better MT performance. In chapter 8, we show that alignments produced by our bottom-up rule-based algorithm lead to increased BLEU scores.

Therefore, instead of building another set of training data in order to train a discriminative model, we manually construct a set of rules based on our previous findings and manual inspection of the data. For the latter, we produce a list of the most frequently occurring node-pair mismatches sorted on the

Source cat.	Target cat.	False +	False -
PP	NP	1	14
PP	VP	4	8
INF	S	0	5
PP	S	2	3
OTI	PP	2	2
...			

Table 6.1: Examples of mismatches by category label combination

particular category label pair involved. We distinguish between false positives, which lower precision, and false negatives, which lower recall.

Our data set consists of the 140 parallel sentences for Dutch to English alignment that we used for our experiments on Lingua-Align training in the PaCo-MT project (chapter 4). Preprocessing steps remain unchanged: Hunalign is used for alignment on the sentence level and GIZA++, using the combination of the *intersection* and *grow-diag* heuristics, for word alignment. The Alpino and Stanford parsers are used to parse the Dutch and English texts, respectively, where the Dutch trees have been post-processed for improved syntactic convergence.

Table 6.1 displays an abbreviated list of some of the category label mismatches. These examples include sentence IDs and matching word phrases, so that they are easily found when viewing the alignments with the Stockholm TreeAligner (Lundborg et al., 2007).

The top result, PP-NP, has 15 mismatches, of which 14 are false negatives and 1 is a false positive. The number of false negatives underscores the recall problem. Also, no PP-NP combination is aligned correctly. It would appear that subtrees with some unusual category label combinations — PPs are usually aligned with PPs and NPs with NPs — are generally not aligned. To rectify this problem using Lingua-Align, we would need to provide more training data - since category labels count among the features used (see the latter part of section 4.4) - or manually tune some parameters. Here we demonstrate that a rule-based approach may be a simple alternative way to deal with the problem.

We proceed by constructing some basic rules where we ignore the actual category label combinations and other metadata. For the purpose of our experiments, we simply assume, similar to the developers of the Dublin aligner (Zhechev and Way, 2008), that the existence of word alignments and

well-formedness of subtree pairs are the most important features determining whether such a pair should be linked.¹ As explained in chapter 2, a subtree pair is well-formed when all of the links of the nodes dominated by the source and target root nodes are shared between these trees only, as well as when the ancestors of node pairs are only linked to each other and not to non-ancestors. A third condition is that nodes are only linked once. For the purpose of this chapter, we only refer to the restriction where descendents are concerned when mentioning the concept of well-formedness. As for the principle on strict one-to-one linking of nodes, we do not regard this as important, since in the overall picture of a stochastic MT system, where some erroneous alignments are to be expected and where high-recall is often preferred over high-precision alignments, allowing violations of this principle should not detract from the ultimate goal. On the other hand, the first principle — namely that of exclusive linking between subtrees — is a significant indication of equivalence which, as we shall see, plays an important discriminative role.

First-level rule

We define the *height* of a non-terminal node as the maximum distance between this node and any of the terminals that it dominates. We have found viewing the height of a node as a function of the distance between non-terminals and its dominated terminals instead of the distance between the non-terminal and the root of the sentence tree to be a useful discriminating feature for our alignment and alignment error correction experiments.

If the height of a non-terminal is 1, this implies that all of its children are terminal nodes. If we restrict our search space to trees where the root has a maximum height of 1, we can more easily study the effect of existing word alignments on non-terminal alignment, without having to take other possible structural features such as height and number of recursive subtrees into account.

We hypothesize that first-level trees sharing word alignments should generally be linked. No alignments should be shared with other trees, making the trees well-formed with respect to their alignments. As we are attempting to increase our recall, we implement the following conditions:

- We do not have a threshold for the number of shared word alignments. Even a single word alignment shared between the trees is enough, as long as they are well-formed.

¹One big difference, however, is that the Dublin aligner uses word alignment probabilities, whereas we do not, at least not at the time of writing.

- We do not distinguish between good (confident) and fuzzy (less confident) alignments in deciding whether or not trees should be linked.

With the above in mind, we can present our algorithm as follows:

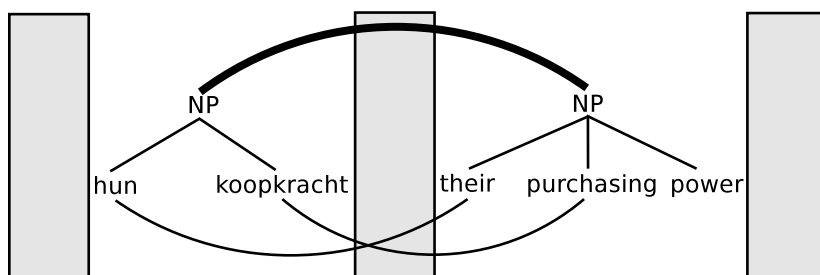
- For every unlinked non-terminal source tree node:
 - If all the children are terminal nodes and one or more of them are linked:
 - Get the target-side nodes to which these terminal nodes link.
 - If these nodes share the same unlinked parent:
 - If this parent's children are all terminal nodes, and
 - If the linked children of this parent all link to children of the above source-side node:
 - Link this parent with the source-side non-terminal node using a confident link.

This simple rule adds a good link between two subtrees if they share one or more word alignment links, as long as they are well-formed and both candidate nodes have only terminals as children. Apart from the relaxed constraints as described above, we also ignore all links involving punctuation, since we noticed that the parsers treat punctuation differently. For example, with some equivalent subtree pairs, a comma would be included on one but not on the other side, and we feel that this should not affect any alignment decisions.

This rule is similar to the first one in the previously discussed work of Menezes and Richardson (2001), with the exception that they only link the trees if all lexical items are already aligned — implying a high-precision approach — and that they apply this on a kind of dependency structure (see chapter 2 for a brief discussion).

Figure 6.1 is an example from the Europarl corpus, showing two trees that have only terminal nodes as children. For simplicity, we have not displayed the part-of-speech tags of the terminal nodes. The word alignments are not perfect: *koopkracht* should be linked to both *purchasing* and *power* and not only to *purchasing*. This does not deter rule application because we have not set any threshold regarding a shared word alignment ratio, leading to correct linking of the two trees in the figure. As we show here, we suggest linking non-terminals even if some of the terminals they dominate are not properly linked to each other. We disallow only that the terminals be linked to nodes that are not children of the candidate non-terminal to be linked.

Figure 6.1: An example of an added link: the two NPs in the picture are linked after application of the rule. If any of the shared word alignments are linked to a node that does not have either of the two NPs as an ancestor, the subtree pair is not well-formed with respect to its links and the rule cannot be applied. The grey rectangles represent such “no go” areas for word alignment links coming from the terminal nodes in question - they may pass through but not end up in them.



Applying the rule in this case goes like this: If we assume that the left side *hun koopkracht* is the source side and we test for the conditions stated by the rule, we start with the left NP. We notice that it is unlinked and we confirm that all children are terminal nodes (*hun* and *koopkracht*) and that at least one of them are linked. Getting the nodes to which these terminals link (*their* and *purchasing*) we can determine that they share the same parent, the right (target) NP. We see that it is unlinked and that its children (*their*, *purchasing* and *power*) are all terminal nodes. We get all the linked terminals (*their* and *purchasing*) and determine that the nodes that they link to (*hun* and *koopkracht*) share the same non-terminal parent, which is the same as the original source-side NP. Now that we have determined that the rule is applicable, we proceed to link the two NPs, implying the translational equivalence of the two substrings that they dominate.

For our experiments, we use 200 Dutch-to-English sentence pairs from a data set which we also used as a test set for our transformation-based learning system (see chapter 7), consisting of a manually aligned gold standard and its

Data set	Precision	Recall	F-score
Lingua-Align	93.4	61.3	74.0
With rule applied	88.8	66.0	75.7

Table 6.2: Evaluation scores after application of first-level rule.

tree-aligned equivalent.² Applying the abovementioned rule to the set reveals 126 added links. 63 of them match with good links in the gold standard,³ and 5 with fuzzy links, where 58 do not match. Ignoring the fuzzy matches, we achieve an accuracy of 52.07%. The result is a clear increase in recall, with a 1.7 increase in F-score (table 6.2). The F-scores reported are those which include both types of links in the recall score, since we would like to take all links into account.

Although the result is not that spectacular, it seems promising enough to warrant further investigation. We hypothesise that it works relatively well because it relies largely on the accuracy of word alignments and that, since it is restricted to the first level above the terminals, divergences between source and target parse trees do not yet play a significant role. Furthermore, the rule presents some flexibility in the case of less than perfectly accurate word alignments, as figure 6.1 demonstrates: The word *koopkracht* should actually be aligned to both *purchasing* and *power*, but still, the current alignments are seen as presenting enough evidence to align these phrases.

On the other side of the coin, we can imagine two very large first-level subtrees sharing only a single word alignment. One might think that they should generally not be aligned, because of the scarcity of word alignment evidence, but application of the rule would still align them. Yet this seems to be an infrequent and negligible occurrence: First-level subtrees have almost always a quite limited amount of leaves, and assuming correctly aligned sentences and relatively accurate word alignment, such a lack of alignments between terminals is scarce.

A more realistic problem is when two subtrees with a substantial difference between the number of leaves are aligned. Manual inspection of erroneous

²We switched to a new data set so that we can experiment on a typical Lingua-Align output instead of on the 140 sentence pairs on which the model was originally trained.

³We have encountered an error in the text itself, equating *Chechnya* with the Dutch for *Czech Republic* (*Tsjechië*), where it should have also been *Chechnya* - in other words, a mistranslation. As a result, we have corrected the word in the data set.

cases suggests that this is a more frequently occurring problem. Lingua-Align partly deals with this problem by using relative leaf counts as a feature. In describing our rule-based bottom-up algorithm in section 6.3, we attempt to deal with this problem as well.

Extensions to the first-level rule

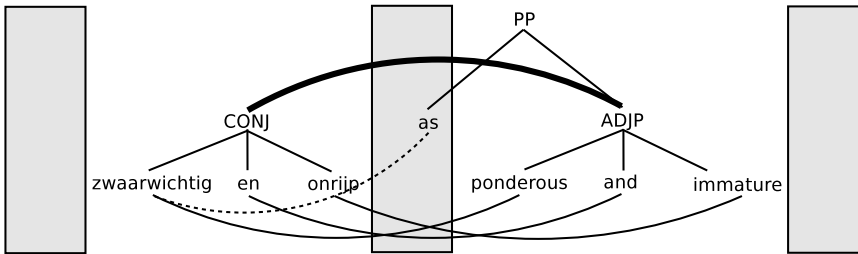
We experiment with three ways to extend the first-level rule. We first apply the same algorithm but from the target side, resulting in a second data set, after which we calculate their union and intersection. Secondly, we attempt to increase the precision by only linking the trees if there is at least one good (confident) link shared between the candidate nodes. Finally, we experiment with a relaxed form of the well-formedness constraint by allowing one fuzzy (less confident) link to go to the “outside” - meaning that a terminal dominated by one of the non-terminals in question links to a node that does not have the other non-terminal as an ancestor (in terms of figure 6.1, ending up in the grey area). We will call this *near-wellformedness*. This is based on our observation in studying training data that a significant number of linked subtrees have such word alignments. Figure 6.2 gives an example of a nearly wellformed subtree pair where the strings *zwaarwichtig en onrijp* and *ponderous and immature* are translationally equivalent even though a fuzzy link goes to the outside.

For the relaxed constraint, we use a slightly different version of our algorithm which is easier to implement, albeit computationally more expensive. Instead of directly checking the linked terminals for the current source-side candidate node, we consider, for every unlinked source-side node, every unlinked first-level target-side node, and determine whether or not they are well-formed in the sense of the algorithm described above.

Implementing target-side rule application, we find that the sets resulting from both the latter and the source-to-target-side application are identical, rendering the same evaluation scores. This can be explained by way of the nature of the constraints: Since only fully well-formed subtrees can be linked, it follows that there is at most one candidate for any given subtree and vice versa, as long as only first-level subtrees are considered. Hence, calculating the union and intersection is a useless endeavour.

In the case of near-wellformedness, it is possible for the order of application (source-to-target versus target-to-source) to matter. With near-wellformedness, it follows that under some circumstances, a second subtree may function as a candidate. A different tree may therefore be linked depending on the order of node traversal. However, in our case, this did not make any difference.

Figure 6.2: An example of a case where nearly wellformed subtrees are linked. The phrase *zwaarwichtig en onrijp* should be translated to *ponderous and immature*, word by word. The confident links are solid and the less confident links have dashes. Here, the word alignment software has also assigned a fuzzy link aligning *zwaarwichtig* with *as*. This link ends up outside in the grey area - *as* does not have the required ancestor *ADJP* - therefore the subtrees are not well-formed. However, since we allow for exactly one fuzzy outgoing link as long as at least one confident link is shared between the subtrees, we proceed to link their roots (*CONJ* with *ADJP*).



The extra constraint that one only aligns the subtrees if there is at least one good link leads, as expected, to an improved precision, albeit only slightly. Balanced F-score is not affected. Not using this constraint but allowing that one fuzzy link can go to the outside decreases the precision but increases recall, with a slight drop in in F-score. Using both the linking constraint and the near-wellformedness condition leads to the best F-score of 75.9.

Table 6.3 displays the results. For comparative purposes, we also apply the rule on the same set but containing only word alignments (Table 6.4). If we do not consider punctuation as explained previously, recall drops slightly, although precision is hardly affected.

Since the rules only apply to the first level above the terminals, in Table 6.4 the recall is understandably low. However, the accuracy achieved is surprising. With a single rule, we have achieved a precision of 86.2% of aligning first-level subtrees on a word-aligned data set. Adding the near-wellformedness constraint increases both the recall and the resulting F-score. The change in the latter is comparable to the tree aligned data set results.

Our results suggest the following:

- Well-formedness is important, but allowing near-wellformedness to some degree may have some merit. This can be explained by pointing out that

Algorithm	Precision	Recall	F-score
Lingua-Align	93.4	61.3	74.0
Fully well-formed, 0+ good links	88.8	66.0	75.7
Fully well-formed, 1+ good links	88.9	65.9	75.7
Nearly well-formed, 0+ good links	83.5	68.4	75.2
Nearly well-formed, 1+ good links	85.5	68.2	75.9

Table 6.3: Non-terminal evaluation scores after application of different forms of the first-level rule on the Lingua-Align set.

Algorithm	Precision	Recall	F-score
Fully well-formed, 0+ good links	85.9	27.5	41.7
Fully well-formed, 1+ good links	86.2	27.3	41.5
Nearly well-formed, 0+ good links	75.3	29.9	42.8
Nearly well-formed, 1+ good links	79.2	29.6	43.1

Table 6.4: Non-terminal evaluation scores after application of different forms of the first-level rule on the word-aligned set.

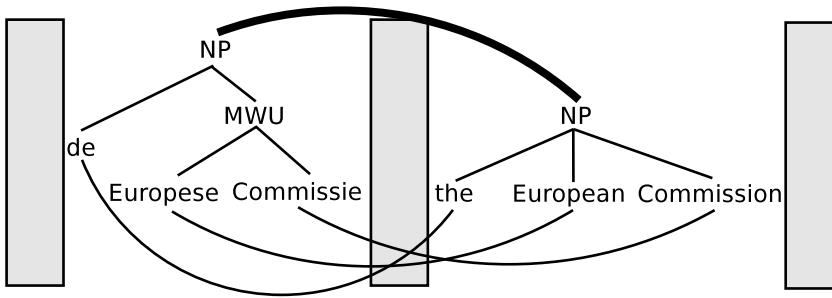
fuzzy word alignments have less confidence and that it can therefore be expected, in a reasonable number of cases, for equivalent subtrees to have some fuzzy alignments to nodes outside the tree. Taking these subtree pairs into account as well may therefore increase recall.

- First-level subtrees can be accurately linked using a hand-crafted rule based on word alignment evidence and completely ignoring metadata such as category and edge labels. This demonstrates the overwhelming importance of word alignment evidence as opposed to other features such as category labels.

Relaxed rules

Since so far we have restricted ourselves to non-terminal nodes on the first level, we construct another rule that is more general than the first, with the exception that children do not necessarily need to be terminal nodes. We call this rule *relaxed*, in opposition to the first rule which is strict in terms of the required height of the subtree roots. Figure 6.3, repeated from figure 2.4, is an example

Figure 6.3: Example of an output of a relaxed form of the first rule: the two NPs in the picture are linked after application of the rule.



where such an alignment is necessary. In this case, the words are perfectly aligned with each other, but note that the NPs involved here would not have been aligned if only the first rule was considered, since the Dutch NP does not only have terminal nodes as children.

We can describe this rule as follows:

- For every unlinked non-terminal source tree node s :
 - If there exists at least one child that is a linked terminal node:
 - Get the set of all the linked leaves of the source-tree node.
 - Get the set of target-side nodes to which these nodes link.
 - Get the lowest common parent t that they share.
 - Get the set of all the linked leaves dominated by t .
 - If these leaves link to source-tree terminals, all of which have s (the current non-terminal source-tree node) as an ancestor:
 - Link s with t .

After applying the relaxed rule, we found that 440 links were added. 214 of them match correctly with good links in the gold standard, 12 of them with fuzzy links and 214 are incorrect. Ignoring the fuzzy links, this amounts to an accuracy of 50%. However, the evaluation score on the final set is a notable improvement (*normal s2t* in Table 6.5), with F-scores higher than those of the

first-level rules. Error analysis suggests that most errors are due to height differences. For example, the Dutch phrase *deze daden begaan* (*perpetrate these deeds*) is erroneously linked to *these deeds*. This in turn suggests that in the context of a greedy linking approach, linking them in the correct order is important. For example, correctly linking *deze daden* to *these deeds* would rule out *these deeds* as an erroneous candidate to be linked to *deze daden begaan* since we only look at unlinked nodes.

In Table 6.5, we also display the results of applying the algorithm from target to source, as well as those of the intersect and union of the previous two sets. We also display the results of allowing for near-wellformedness, both with and without the good link constraint, again with the above-mentioned directions and combinations. Again, as with the first-level rule, for the near-wellformedness rules, we start off with a candidate node-pair, as opposed to starting with just a non-terminal and immediately inspecting the leaf links. To investigate the impact of word alignments more properly, we ignore well-formedness violations at the non-terminal level.⁴

As expected, relaxing the well-formedness constraint leads to better recall but lower precision. However, there are few differences between the F-scores in general. Near-wellformedness has no positive effect here.

When we apply the union of two data sets, note that sometimes, the same node could be linked to a different node in each set. For example, if source node *a* is linked to target node *b* in set *A* but the same source node *a* is linked to target node *c* in set *B*, the union will contain both alignments, $a \leftrightarrow b$ as well as $a \leftrightarrow c$. We regard these as different alignments and, since this is a union, they are allowed, although this means that non-terminals are not strictly linked in a one-to-one fashion anymore.

When applied to the word-aligned data set (Table 6.6), we find more or less expected results: Precision is lower than first-level rules, recall is higher, but changes in F-scores are unimpressive. Of course, we are still restricting ourselves to nodes that have at least one terminal node, and should not expect any surprises. An interesting observation is that in all cases, the union of the sets does better. However, in Table 6.5, this is only true for “Strict union”.

From the results of the two main types of rules that we have investigated, we can conclude the following about alignment for the data set involved:

- Generally, strict well-formedness and good links requirements increase precision, but decrease recall.

⁴Experiments show that this actually leads to slightly better scores.

Data set	Precision	Recall	F-score
Lingua-Align	93.4	61.3	74.0
Normal s2t	80.0	77.0	78.5
Normal t2s	81.6	73.6	77.4
Normal intersect	84.8	72.0	77.9
Normal union	77.4	78.5	77.9
Strict s2t	78.9	79.1	79.0
Strict t2s	81.0	74.5	77.6
Strict intersect	84.1	72.9	78.1
Strict union	76.5	80.7	78.5
Near-wfness (normal) s2t	70.2	84.1	76.5
Near-wfness (normal) t2s	72.8	81.2	76.8
Near-wfness (normal) intersect	74.0	79.9	76.8
Near-wfness (normal) union	69.2	85.4	76.5
Near-wfness (strict) s2t	71.5	83.9	77.2
Near-wfness (strict) t2s	74.2	81.0	77.5
Near-wfness (strict) intersect	75.5	79.7	77.5
Near-wfness (strict) union	70.4	85.2	77.1

Table 6.5: Evaluation scores after application of different forms of the relaxed rule on the data set aligned by Lingua-Align. *normal* means that there is no necessity of a good word alignment link, where *strict* means the opposite. *s2t* denotes source-to-target and *t2s* target-to-source. *near-wfness* denotes *near-wellformedness*. With the others, strict well-formedness is implied.

- Near-wellformedness increases recall with little drop in precision.
- The direction of the algorithm may make a significant difference. The intersection of both directions increases precision and their union increases recall. Neither the union nor the intersection scores consistently better than the other.
- First-level alignments can be linked with a high precision using few features.
- Relaxing the height requirements increases recall but decreases precision. F-scores improve very clearly.

Data set	Precision	Recall	F-score
Normal s2t	67.3	57.3	61.9
Normal t2s	67.2	45.0	53.9
Normal intersect	81.8	42.3	55.8
Normal union	59.7	60.0	59.8
Strict s2t	76.3	67.2	71.5
Strict t2s	68.6	48.5	56.8
Strict intersect	80.5	45.6	58.2
Strict union	68.6	70.2	69.4
Near-wfness (normal) s2t	61.9	68.6	65.1
Near-wfness (normal) t2s	64.2	61.1	62.6
Near-wfness (normal) intersect	67.9	55.7	61.2
Near-wfness (normal) union	59.7	74.1	66.1
Near-wfness (strict) s2t	63.2	68.4	65.7
Near-wfness (strict) t2s	65.7	60.8	63.2
Near-wfness (strict) intersect	69.9	55.4	61.8
Near-wfness (strict) union	60.8	73.8	66.7

Table 6.6: Evaluation scores after application of different forms of the relaxed rule on the word-aligned data set (not processed by Lingua-Align).

- The order of alignment matters: It would seem that a directional approach has merit. Given the good results of the relaxed rule, the relative heights of root nodes seem to be less important than the order in which they are linked.

Given the above, it makes sense to first produce the alignments which render the highest precision. We therefore propose an algorithm in the next section that links subtrees from the bottom up. We also implement two additional similarity constraints in an attempt to deal with the problem regarding height differences.

6.3 Bottom-up tree alignment error correction

Based on our previous conclusions, we implement an algorithm that greedily aligns candidate node-pairs in a bottom-up fashion. In the PaCo-MT project, an implementation of this algorithm led to increased BLEU scores (see chapter 8).

We traverse all source-tree non-terminal nodes, starting on the first level above the terminals, and, for every node considered, inspect all candidate target-tree non-terminals, starting on the the first level. In this way, we ensure that lower-level nodes are linked first before higher-level nodes. If conditions are satisfied, a link is made before moving on to another subtree pair. For a link to be made, all of the following criteria need to be met:

- The subtree pair must be well-formed or nearly well-formed with respect to the links of the nodes dominated by them. As in the previous section, we define near-wellformedness as the condition where there is one and only one fuzzy link going to the outside of the candidate subtrees and there exists at least one good link between them. We ignore the cases of outgoing fuzzy links or shared good links where punctuation is concerned.
- The subtree pair must have a certain degree of similarity. In this case, we define a good enough similarity as one where both the leaf count similarity and the link count similarity are above a certain threshold. Defined below in more detail, the leaf count similarity is based on the ratio between the number of leaves on the source and on the target side. This ratio is also used as a feature in Lingua-Align (chapter 4). Additionally, a link count similarity, similar to the *children_links* feature used in Lingua-Align, is calculated based on the ratio of the linked leaves between the subtrees and the total number of leaves between them.

For the similarity criteria, we also suspected that we needed to take the number of leaves into account. For example, a subtree with one leaf should often be linked to a subtree with two leaves, but perhaps it should be less common that a subtree with five leaves should be linked to one with ten leaves, even though the ratio is the same. In other words, there could be a bias, where shorter strings are viewed more favourably over longer strings. We therefore suspected that the number of unlinked leaves, not just the ratio, should also play a role.

If one assumes that both the ratio and the difference count have the same weight, it would be useful and simple to apply the geometric mean of the numbers, since they occupy different ranges. For both similarity measures, these constitute the leaf count similarity and link count similarity scores respectively.

First, we define the leaf ratio lr , similar as in Tiedemann and Kotzé (2009a):

$$lr(s_i, t_j) = \frac{\min(|x|, |y|)}{\max(|x|, |y|)}$$

where x denotes $s_x \leq s_i$ and y $t_y \leq t_j$. s_i and t_j are the respective candidate source and target-side nodes, and s_x are the leaves (terminal nodes) dominated by s_i and t_y are dominated by t_j .

The link count ratio, similar to the *children_links* feature in Tiedemann and Kotzé (2009a), is calculated as follows:

$$\begin{aligned} lcr(s_i, t_j) &= \frac{\sum_{s_x} \text{linked}(s_x, s_i) + \sum_{t_y} \text{linked}(t_y, t_j)}{\sum_{s_x} \text{leaf}(s_x, s_i) + \sum_{t_y} \text{leaf}(t_y, t_j)} \\ \text{leaf}(l, n) &= \begin{cases} 1 & \text{if } n \text{ dominates } l \\ 0 & \text{otherwise} \end{cases} \\ \text{linked}(l, n) &= \begin{cases} 1 & \text{if } n \text{ dominates } l \wedge l \in L \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where L is the set of all linked leaves (aligned terminal nodes), but only where the links go back to descendents of the candidate node on the other side. The reasoning behind this is that nodes with links going to other subtrees do not contribute to well-formedness and should therefore not contribute to the similarity score. Some terminals may be linked more than once, but since we count the nodes and not the links, we do not take this into account.

We now define the differences as follows:

$lcd(s_i, t_j)$: leaf count difference - the difference in the number of leaves dominated by s_i and t_j

$llcd(s_i, t_j)$: linked leaf count difference - the difference in the total number of leaves dominated by both s_i and t_j and the total number of linked leaves dominated by both s_i and t_j

We can now define the leaf count similarity score lcs as follows:

$$lcs(s_i, t_j) = \sqrt{lr(s_i, t_j)lcd(s_i, t_j)}$$

and the link count similarity score $llcs$ as follows:

$$llcs(s_i, t_j) = \sqrt{lcr(s_i, t_j)llcd(s_i, t_j)}$$

For both similarity measures, a previously determined threshold determines whether or not the subtrees in question are similar enough to be linked. This threshold can be estimated using a training data set and a machine learning method, or it can be manually tuned. Here, we only present the rule-based part of the algorithm, along with the results of using a threshold that we manually determined.⁵

A link is only made if both similarity scores are less than or equal to their respective thresholds and the subtrees in question are at least nearly well-formed. Additionally, we assume that sentence alignment is perfect and therefore change all fuzzy links between root nodes to good. The reasoning behind this is that since sentence alignment is very accurate, this has a good chance to be a better reflection of sentence pair equality and may improve MT performance as well.

Instead of assuming equal weight between the ratios and differences and using their geometric means to calculate the similarity scores, we can subtract a weighted difference from the ratio, which we will call a penalty score. The idea behind this is similar to the previous formula: The larger the difference, the less important the ratio should be, so we have a lower score in this case. However, simply subtracting the difference would result in a skewed figure. We therefore introduce a normalization factor z by which the difference count is divided. The final values now need to be more than the determined thresholds for the trees to be linked. For our experiments, we have settled to set z at the value 80, with the leaf count similarity threshold at 0.35 and the link count similarity threshold at 0.45.

If we assign x to be the leaves dominated by s_i and y to be the leaves dominated by t_j ,

$$lcs(s_i, t_j) = lr(s_i, t_j) - \frac{abs(|x| - |y|)}{z}$$

where x denotes $s_x \leq s_i$ (the count of the source-side subtree terminals), y $t_y \leq t_j$ (the count of the target-side subtree terminals), and z the normalization value. Similarly, our link count similarity score can be expressed as follows:

⁵1.45 for the leaf ratio similarity and 1.4 for the linked leaf ratio similarity.

Data set	Precision	Recall	F-score
Source side first	76.0	79.8	77.9
Target side first	76.9	81.1	78.9
Intersection	81.4	77.4	79.3
Union	72.4	83.4	77.5

Table 6.7: Evaluation scores on the tree aligned data set after application of different bottom-up heuristics

$$llcs(s_i, t_j) = llr(s_i, t_j) - \frac{|x| + |y| - (|x_l| + |y_l|)}{z}$$

where x_l and y_l are the linked leaves in x and y respectively.

We also experimented with using the existence of verbal part-of-speech tags on both sides as a criterion. In some cases, the existence of a verb on only one side indicates the fact that the current nodes would be linked on the wrong level, since verbs are often translated as verbs, at least in the case of the closely related languages involved. The rule states that if a verb is found on only one side, the nodes are not to be linked. We found that imposing this restriction may lead to increased precision, but that recall slightly suffers. Often, a verb on one side is differently lexicalized on the other side, for example through nominalization, leading to a false negative. Since this experiment did not lead to a significant improvement, we decided not to use it for producing the parallel treebanks for the final version of PaCo-MT. However, we do use it as a feature in our transformation-based learning experiments (see chapter 7).

We apply the algorithm using geometric averages on our data set bidirectionally, as before, and then calculate the intersection and the union. The results are displayed in Table 6.7. The F-scores are generally slightly better than those of the relaxed rule. Notable is that in this case, unlike as with the relaxed rule, the union is clearly worse than the intersection.

We also apply the algorithm on the word-aligned data set (Table 6.8). The scores are predictably lower, but we see the same tendencies: The intersection has clearly the highest precision and F-score, and the union has the highest recall but its precision suffers.

The current version of the algorithm allows for unrestricted node height differences, relying on the assumption that more similar subtrees closer to the level of the current source-tree node will be linked first, while being constrained by

Data set	Precision	Recall	F-score
Source side first	67.8	56.9	61.9
Target side first	68.3	57.6	62.5
Intersection	79.3	51.8	62.7
Union	60.9	62.7	61.8

Table 6.8: Evaluation scores on the word-aligned data set after application of different bottom-up heuristics

Height thresholds	Precision	Recall	F-score
no threshold	76.0	79.8	77.9
0 to max	77.3	75.5	76.4
min to 0	77.7	77.1	77.4
-1 to +1	77.3	77.3	77.3
0 to +1	77.9	74.1	76.0
-1 to 0	78.5	76.0	77.2
-2 to +2	76.5	79.3	77.9
0 to +2	77.6	75.2	76.4
-2 to 0	77.9	76.9	77.4
-3 to +3	76.0	79.6	77.8
0 to +3	77.4	75.4	76.4
-3 to 0	77.7	77.0	77.3

Table 6.9: Precision, recall and balanced F-score when applying the bottom-up rule addition algorithm with varying thresholds of height difference between candidate source and target nodes. 0 refers to the height of the currently selected source tree node, *min* is the minimum height (1) and *max* is the maximum height of the sentence tree.

the similarity thresholds. An alternative solution would be to restrict alignment within a certain height threshold. We experiment with different thresholds, using the source to target implementation. Table 6.9 displays the results.

Not surprisingly, precision tends to go up as the thresholds are closer to each other, and recall tends to go up as the height restrictions become more relaxed. However, no F-score exceeds the original setup with no threshold (77.9), but the one with the thresholds of -2 and +2 equals it.

If we produce the target-to-source alignments of the algorithm using no height thresholds (F-score: 79.0) and the resulting intersection, we get the the best F-score of 79.4, with a precision of 81.8 and a recall of 77.1.

Error analysis reveals visibly fewer cases of mismatched tree spans, but there are cases where the verbal rule would have been helpful.

6.4 Conclusion

Our results suggest that the bottom-up algorithm without height restrictions does rather well to increase alignment recall. In general, combining source-to-target and target-to-source heuristics leads to overall better scores. We have also formulated additional features to contribute to increased accuracy of alignment, including two different measures of subtree similarity. It is also worth noting that *Lingua-Align* utilizes a bottom-up alignment procedure with success (chapter 4), which performs better than a top-down approach. However, we have obtained our improved results only after trial and error experimentation using manually defined rules and values, which may change according to the language pair and data set used. It would be more ideal to learn these values automatically. It would be entirely possible to integrate these features into machine learning algorithms such as those used by *Lingua-Align*.

In this chapter, we have experimented on complementing existing alignments by heuristically implementing manually constructed rules. Although we have gained some valuable insights and can report success with this endeavour, automatic acquisition is more convenient and may lead to better results. In the next chapter, we introduce our implementation of a transformation-based learning system for tree alignment and alignment error correction. So far, we have only focused on increasing recall by adding links to existing data sets. Our implementation also removes links, constructs more refined rules and generally produces a more accurate alignment set.

Transformation-based tree alignment and alignment error correction

In this chapter, we describe our experiments using Brill's transformation-based learning (TBL) algorithm on the tree-to-tree alignment problem. We will discuss our motivation for this in the next section, followed by an introduction to TBL in section 7.2 and a description of our adaptation to the problem in sections 7.3, 7.4, 7.5 and 7.6. A presentation of our experiments follows in section 7.7, followed by a discussion of the results in sections 7.8 and 7.9. Finally, we conclude this chapter in section 7.10.

7.1 Motivation

In chapter 6, experiments suggest that a rule-based approach to tree alignment error correction is an effective complement to statistical alignment, affirming the promising results obtained in other systems described in chapter 2. However, these experiments only make use of manual rule-based components with a limited scope. It would make sense to also be able to learn many rules that can be applied with much greater precision without the effort of constructing them manually.

Transformation-based learning has proven to be a simple yet powerful algorithm, applicable to a variety of tasks, also outside the field of NLP. Its error-driven nature makes it an ideal candidate for improving performance on data sets which have already been processed, but can also be successfully implemented as a classifying algorithm in itself. Furthermore, unlike pure statistical algorithms, its rule-based nature provides for a very transparent error analysis procedure. We believe that in view of this, TBL is an excellent candidate for tree alignment error correction. Additionally, TBL has to our knowledge not yet been applied to this particular problem at the time of this writing.

As with chapters 4, 5 and 6, we design and implement our experiments in view of eventual application in syntax-based MT. However, we believe that the results of our experiments also provide some interesting linguistic insights. For example, error analysis can shed light on patterns regarding structural and lexical differences between the languages involved. Conversely, the most successful rules tell us which linguistic or structural generalisations can be made in the alignment process with a high rate of success.

7.2 Transformation-based learning

Transformation-based learning (TBL) is a supervised rule-based machine learning algorithm used to classify previously unseen examples of a data set. Ever since the introduction of the original concept by Eric Brill and its application on part-of-speech tagging (Brill, 1992; Brill, 1994; Brill, 1995), there has been a flurry of publications on the topic, mainly on refining the algorithm (Samuel, 1998; Ngai and Florian, 2001), increasing the scope (Florian and Ngai, 2001), or adapting it to different problems such as parsing (Brill, 1993a; Brill, 1993c; Brill, 1996; Fung et al., 2004), prepositional phrase attachment disambiguation (Brill and Resnik, 1994), base NP chunking (Ramshaw and Marcus, 1995), spelling correction (Mangu and Brill, 1997), dialogue act tagging (Samuel et al., 1998), phrase chunking (Florian et al., 2000), handwritten character segmentation (Kavallieratou et al., 2000), named entity recognition (Black and Vasilakopoulos, 2002), semantic role labeling (Higgins, 2004; Williams et al., 2004), word alignment (Ayan et al., 2005), newspaper headline generation (De Kok, 2008) and syntactic tree transformation to improve convergence (Burkett and Klein, 2012).

Although since the 90s there has been significant progress in optimizing and refining the algorithm, its basic premise remains unaltered: Training occurs by applying a series of rules to a data set. Each is applied separately in order

to find the single best rule which, when applied, improves performance most. This is made possible by comparing the output to a gold standard. That rule is then marked as the best at that point and set aside. The data set is retained, with the changes made by the rule and other possible rules in earlier iterations, up to that point. Then all rules are again tested on the new and updated data set to find the next single best rule which improves performance the most. This continues until no rule can be found that improves the accuracy of the newest data set. The set of learned rules is essentially the model that is to be used on new unseen data.

Rules are generated by a pre-defined set of templates. A rule template can be defined as an abstraction of a rule which can be realised to more than one different instance. A triggering environment realises the rule when conditions specified in the environment hold. Generally, we can write it as an *if...then* statement:

if condition X then do Y

More specifically we can write this as the following example in the context of a tagging application:

if previous element is X then change current tag to Y

Rule templates refer to features that can potentially describe useful relationships between elements in the data. In the example above, the relationship between the current and the previous tag is assumed to have the potential to be a useful feature in helping to determine the identity of the current tag.

After constructing the template list, we generate instantiations, which essentially form the list of all possible rules that can be applied. For the above tag template, this includes a list of possible tags to be included either as the previous or the current tag. Not all tag pairs may be useful, therefore the challenge lies in trying to find an optimal solution lying somewhere between brute force generation and more manual work (for example, using random sampling).

Often, learned rules are initially very general but become more specific as they approach the ground truth represented by the gold standard. An effective analogy is provided by Samuel (1998), who attributes it to Terry Harvey. A painter can decide, upon painting a barnyard scene, to first colour everything blue, since the sky will comprise the majority of the canvas area. After the paint dries, he paints the barnyard which comprises a smaller area, but without

taking care to avoid the windows, roof and doors, which will be painted more precisely at a later stage. Similarly, the first few rules can be applied over a broad area, increasing recall but also making many mistakes. As more rules are applied, some mistakes are corrected and precision generally goes up.

With many TBL applications, the data is first annotated with a so-called *initial state annotator*. To explain this further it in terms of a part-of-speech tagging application, this can range from naively tagging all words as nouns to the output of an already accurate tagger with the intention of further increasing its accuracy. The object of the training phase is to learn the correct order of rules to get as close as possible to the *ground truth* represented by the gold standard.

7.3 Transformation-based learning for tree alignment

In this section, we describe our adaptation of TBL to the tree alignment problem. More specifically, as in chapters 4 and 6, the problem we are attempting to solve is the accurate alignment of non-terminal constituents between a pair of syntactically annotated parallel sentences. Minimally, we assume structurally valid phrase structure trees with existing word alignments. However, any additional metadata can be incorporated and used as features, such as dependency information, POS tags, category labels, lemmas, and so on. Instead of an alignment set that is only word aligned, one containing existing non-terminal alignments can also be used as an initial state annotator.

As discussed in the previous section, in TBL we apply a list of transformations consisting of a so-called *triggering environment* and a *rewrite rule*, which is the action undertaken when the required conditions specified by the triggering environment are met. In the case of tree alignment, there can always be only two general courses of action: add or remove an alignment. Whether an alignment can be added or removed depends on whether the currently inspected node pair is in fact aligned.

For every iteration, best rules are globally applied - currently, no feature extraction exists until the full data set has been updated by the rule. Adding or removing alignments may of course change the actual feature values of ancestors or descendents of particular node pairs, possibly leading to errors. However, assuming good feature selection and proper implementation, we can rely on later rules to correct these errors.

At the time of writing, we do not distinguish between adding different types of links, instead assuming that all links to be added are confident (good, not fuzzy - see section 7.4). Therefore, our rule templates consist solely of gener-

alisations of possible conditions under which to add or remove an alignment. Since most existing links in our gold standards are confident, not adding fuzzy links does not significantly affect our results.

As mentioned in chapter 2, a transformation can be represented in an abstract form called a *template*. The following is an example of a possible template in the context of tree alignment:

- if current node pair has a height difference of less than x , then change link type to y

where the link type can in this case only be ADD or REMOVE. However, the triggering environment may consist of a number of possible features, which we will discuss in the next section.

7.4 Features

Thanks to our experience in building training data and previous tree alignment experiments, we have identified a number of features that could be useful to act as triggering environments. We explain how we incorporate them into our system in section 7.5. All features are optional, although at least one is needed for training.

Tree features

Unary nodes

Some phrase-structure trees contain constituents that have only a single child node. Depending on the alignment scheme, one may prefer aligning either the parent or the child (or even the unary grandchild, if it exists). For example, in our gold standards we preferred to align the lowest possible node, since the information is most specific. For example, a VP can also be an S. Aligning the VP is more informative since there are also other types of S having different unary daughters.

Apart from checking if a specific node is in a unary relation, one may specify certain special conditions or constraints. For example, an S may be in a unary relationship with a ROOT node, apart from the fact that the ROOT node is also the parent of a full stop. One may decide that punctuation should be ignored in some contexts and make an exception in this case.

Subtree similarity

In the same chapter (section 6.3), we describe two key similarity features that we call leaf ratio similarity scores and linked leaf ratio similarity scores. Given a subtree pair, a leaf ratio similarity score is a measure of the ratio between the number of terminals (leaves) of the subtree with the least number of leaves and that of the subtree with the most number of leaves, as well as the leaf count difference between the two sides. The idea is that together, they constitute a measure of similarity, and the more similar the subtrees are, the more likely it is that their roots should be linked (given certain other conditions such as the condition that they share at least one word alignment). Using the geometric mean of the two numbers (ratio and count difference), reflects the degree to which they share the same weight. It has the effect that the lesser score is weighted more heavily. If we use a penalty score, we assume that the weights are different. We attempt to model these differences using the value of the denominator.

A similar measure is calculated by the linked leaf ratio similarity score. In this case, the lowest value is the number of leaves in both subtrees that are linked, and the highest value is simply the total number of leaves in both subtrees. The idea behind this is that the more leaves are linked between two subtrees, the more likely it is that the trees should be linked. The ratios between these two numbers and the difference between them are used in exactly the same way as above (chapter 6) to calculate two alternative ways of representing subtree similarity. The formulae for these measures are presented in section 6.3.

Height

We have found it useful to define the *height* of a node as the longest path from any of its leaves to the current node. Relative height difference between a given subtree pair could conceivably be a feature in helping to determine whether or not they should be linked. For example, it seems unlikely that a subtree at the first level above the terminal and spanning fewer words should be linked to a more complex subtree containing many recursive subtrees and typically spanning more words.

Link features

Link well-formedness

We use the *well-formedness* feature as a measure of whether or not there are links from any of the nodes in the subtree pair that are not shared by both of the trees. If one or more such “outgoing” links, which does not have the root of the other tree as an ancestor, exist, the trees are not well-formed with respect to its links. As stated in chapter 2, our definition of well-formedness is based on the following axiom as stated by Hearne et al. (2007):

- A node in a tree may only be linked once.
- Descendants of a source linked node may only be linked to descendants of its linked node on the target side and vice versa (target to source).
- Similarly, ancestors of a source linked node may only be linked to ancestors of its linked node on the target side and vice versa (target to source).¹

Similar to previous chapters, we will only refer to the second point when mentioning well-formedness, as it is a significant indication of equivalence and an important discriminating feature.

As with the work presented in Zhechev and Way (2008), we distinguish between so-called *good* and *fuzzy* links, which denote confident and less confident links respectively. This distinction is used for a number of features described below.

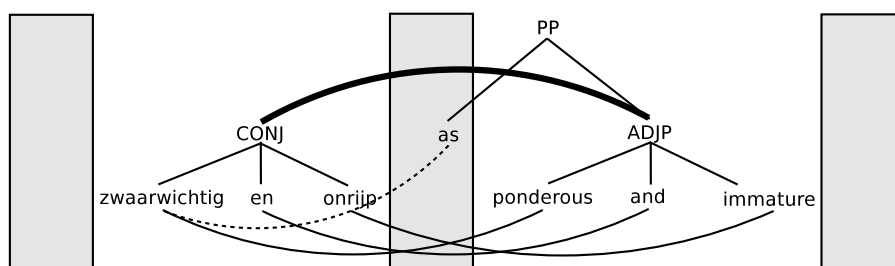
In chapter 6, we have noticed the value of allowing for *near-wellformedness*. Restricting linking subtrees to only those that are strictly well-formed may lower recall significantly. Data sets containing high-recall word alignments often have incorrect links between the terminals which should ideally not play a role in determining whether or not the subtree pair in question should be aligned. We have noticed that in some cases, allowing for one or two less confident word alignments to violate the principle increases both the recall and the resulting F-score. Figure 7.1 explains this graphically.

For our TBL system, it is possible to check for near-wellformedness in a number of possible ways:

- checking for n number of good outgoing links

¹As mentioned in chapter 2, one side is often called the source side and the other the target side, even if eventual translation from the former to the latter is not the end application one has in mind.

Figure 7.1: An example of a non- (but nearly) well-formed linked subtree pair, indicated by the thick line connecting the nodes CONJ and ADJP. The fuzzy (and incorrect) alignment between *zwaarwichtig* and *as* is a less confident link leading to non-wellformedness of the subtrees, since *as* does not have ADJP as an ancestor. However, if we ignore this alignment and allow near-wellformedness, we can still link the trees in question. The grey areas represent “no go areas” for word alignments which render the subtrees non-wellformed whenever a descendent of any of the subtree roots is linked to a node there.



- checking for n number of fuzzy outgoing links
- checking for n number of non-terminal outgoing links
- checking for all or any of the above, but ignoring punctuation - in other words, any links involving terminal nodes with punctuation are not counted

where by “outgoing” links we mean links between nodes where one of the nodes that are being linked has a relevant source or target subtree root node as an ancestor and the other one does not. For example, in figure 7.1, repeated from figure 6.2, the subtrees CONJ and ADJP are linked. With this in mind, we note that there is an outgoing link aligning *zwaarwichtig* with *as*. The word *zwaarwichtig* has a relevant ancestor - the parent node CONJ - whereas the word *as* has not, since ADJP, which is aligned to CONJ, is not an ancestor of *as*.

This can also easily be changed to count the number of nodes that have these links instead of the links themselves. In the case of multiply aligned nodes, these two numbers can differ significantly.

Based on manual inspection, we regard even one outgoing link from a non-terminal node to be significant evidence to discard a candidate node

pair for alignment. Therefore, in our experiments, we did not allow for near-wellformedness in this case. However, it is still possible to implement this if one so chooses.

Link types

If a subtree pair shares only a few low probability (fuzzy) word alignments, it seems reasonable to conclude that perhaps the trees are not that equivalent to each other. The converse - that they are fairly equivalent if they share at least a few strong word alignments - seems to be true. Of course, the lack of any shared word alignments whatsoever would seem to be present strong evidence against the alignment of the subtree pair in question. We therefore include checking for the existence of a specified link type as a possible feature.

Orthographic features

It seems reasonable that if two strings are exactly equivalent, they might also be semantically equivalent, and should therefore be aligned, especially in the case of closely related languages. Additionally, if the span of a string includes the first or last token of the sentence, the same may be true of its translational equivalent.

One may additionally define a wide range of possible conditions or constraints such as the presence of only lower case letters, no punctuation, and so on.

Other features

Element on only one side

In a significant number of cases, we have found that if a verb occurs in either the source or target subtree but not on the other, they should not be aligned. The verb is conceivably the most important word class with which one describes an action. If one assumes the sentence pair to be a more or less equivalent translations of each other, it stands to reason that an action described on the one side must also occur on the other side. Of course, there are exceptions, and in many cases, the equivalent of a verb can be realized as another class such as a noun. However, it has still proven to be a useful discriminating feature. Likewise, other discriminating features such as numbers and capitalized words may also be used in a similar fashion.

Element at beginning or end of span

We have found that if any of the subtrees have a punctuation mark at either end, they are likely to be either at the beginning or end of the sentence or a clause, which in itself can be a useful discriminating feature.

Of course, this can be generalized to any element. Another useful element has proven to be the existence of an alignment that ensures well-formedness - in other words, an alignment from the beginning or end token that links to one on the other side that has the the root of the subtree as an ancestor. If this occurs on both source and target-sides and both at the beginning and end tokens, this seems to be a good indication, along with well-formedness, that the source and target-side spans are equivalent. If one side is not linked, this could mean that perhaps an ancestor ought to be linked instead. Figure 7.2 provides an example.

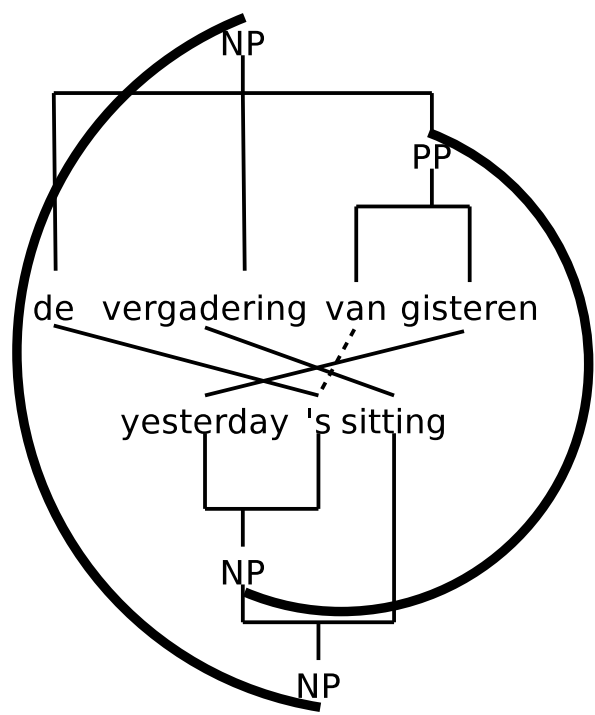
Rule template with feature set

The implementation (see next section) allows any combination of features to be applied. Given a subtree pair, a specific feature is either applicable or not, meaning that we have a hypothesis space consisting of boolean values with the size of 2^n where n is the number of features. Furthermore, features themselves may contain a possibly infinite range of values. For example, one may check for a height difference of 4, or 5, or 6, ad infinitum. This is only restrained by its practical application, which differs depending on the nature of the data being processed.

Given this, we proceed to specify our rule template which contains abstracted versions of the features described before. Since the rewrite rule is only ADD or REMOVE - or more simply put, changing the link status - we do not mention this here. Note that for our final experiments, we have only used a very small set of all possible feature values. We split the template up into different tables, one for each feature type (Tables 7.1, 7.2 and 7.3).

In Table 7.1, please note the following at the similarity scores using geometric means: If the difference in leaf/link counts is 0, then the geometric mean will of course also be 0, since 0 is multiplied by the ratio before taking the square root. However, since this is a special case where the similarity is perfect, we return a very high number instead of 0. The similarity scores using the penalty scores are indicated as potentially infinite since the penalty score consists of a division of a count difference by an arbitrary denominator. This may lead to positive and negative numbers of any real value. If the denominator is 0, the

Figure 7.2: An example of an equivalent subtree pair where the first and last words of both sides (*de, gisteren, yesterday* and *sitting*) are linked to a descendent word of the respective other side. The Dutch phrase literally means *the sitting of yesterday*. The dotted word alignment indicates a fuzzy alignment, while the solid lines are confident.



value returned is conventionally also 0. Linked leaf ratios can include 0 since there may not be any links. Conversely, leaf ratios are always greater than 0 since a non-terminal node dominates at least one leaf.

7.5 Implementation

The system is implemented in four distinct phases:

- training
- rule cutoff selection
- testing
- processing new data

For the first three phases, a gold standard is needed, and for each also a version of the same data set which has been processed by the *initial state annotator*. This can, for example, be the output of a tree aligner or a set containing only word alignments. This we will call the *automatic* data set.

Our tool of choice for creating gold standards, described in previous chapters, is the Stockholm TreeAligner (Lundborg et al., 2007). This tool provides a graphical UI allowing the user to draw lines between nodes in parallel trees. These lines are stored as alignments in an XML file which can be processed by automatic alignment software. This file is in what we will call *STA-XML format*. The Stockholm TreeAligner only accepts trees in TIGER-XML format, and for this reason, it is also our format of choice.

Training

Training initiates by extracting various features from both treebanks and the automatic STA-XML alignment file. The features are specified by the user in a text file using a special syntax. These are used to assign rule statistics to all possible non-terminal node pairs. The rule statistics essentially assign a profile of boolean feature values to every node pair according to the specified rule instantiation. In other words, every unique set of binary numbers refers to the feature profile of the node pairs having those values. These have the potential to function as triggering environments during actual rule application.

Figure 7.3 illustrates an example of the deletion of a link using a remove rule. The tree pair in question is the Dutch NP (s5_4) which is erroneously

Triggering environment	Range of x	Range of y
source node has unary child under condition x	none / ignore punctuation	N/A
target node has unary child under condition x	none / ignore punctuation	N/A
leaf ratio similarity score of range x using penalty score and normalization value of y	∞	∞
linked leaf ratio similarity score of range x using penalty score and normalization value of y	∞	∞
leaf ratio similarity score of range x using geometric means	> 0	N/A
linked leaf ratio similarity score of range x using geometric means	> 0	N/A
source and target node height difference is in range x	≥ 0	N/A
leaf ratio of range x	> 0 and ≤ 100	N/A
linked leaf ratio of range x	≥ 0 and ≤ 100	N/A

Table 7.1: Rule template with list of possible features acting as triggering environments, for tree features.

Triggering environment	Range of x	Range of y	Range of z
x number of word alignments of link type y leads to non-wellformedness under condition z	≥ 0	good/fuzzy	none / ignore punctuation
x number of non-terminal alignments leads to non-wellformedness under condition y	≥ 0	none / ignore punctuation	N/A
link of type x exists	good/fuzzy	N/A	N/A
start and end leaves on both sides are linked under condition x	links are well-formed	N/A	N/A

Table 7.2: Rule template with list of possible features acting as triggering environments, for link features.

Triggering environment	Range of x	Range of y
elements x on source and target-sides are equivalent	whole string / a word containing one or more capital letters or numbers	N/A
elements x exists on both source and target-sides	capitalized word / number	N/A
element x exists on either the source or target-side	punctuation at beginning / punctuation at end	N/A
element x exists on only one side	word containing either a number or any of: <>&*~\$#@+_/\\~{}[] / word containing number / word containing one or more capital letters / any part-of-speech tag in the tagset	N/A

Table 7.3: Rule template with list of possible features acting as triggering environments, for orthographic and other features.

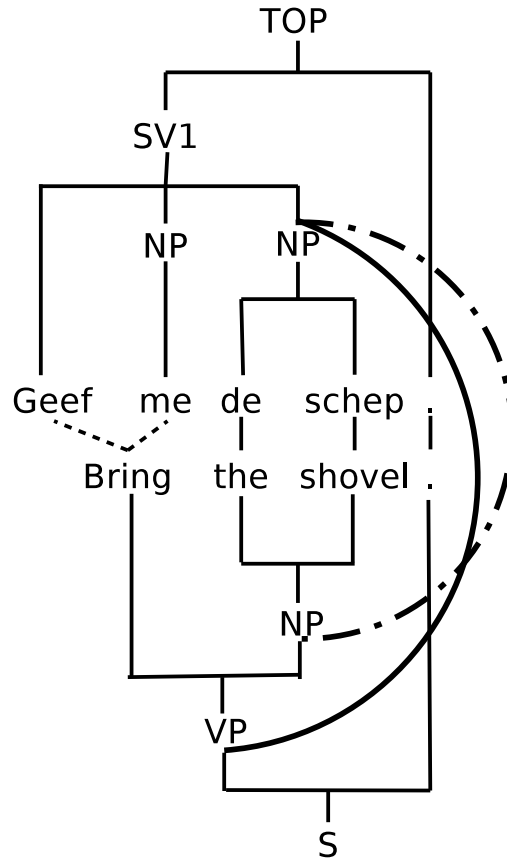
linked to the English VP (s5_503). The NP dominates *de schep* (*the shovel*) while the VP dominates *Bring the shovel*. These two phrases are clearly not equivalent. A remove rule associated with a specific set of feature values could lead to such cases being identified and removed. In a later stage, an addition rule with the feature values of the correct NP pair (indicated by the dashed line) could lead to these tree pairs being correctly linked.

The following feature values could be extracted from the abovementioned node pair to be associated with a remove rule:

Node pair (s5_4)(s5_503) has the following properties:

- source node has a unary child = FALSE

Figure 7.3: An example of a tree pair having the feature values as described in the example list. The tree pair is the Dutch NP which is erroneously linked to the English VP. A remove rule reacting to the current feature values as triggering environment could lead to the current link being removed, while a later addition rule could add the correct link between the two NPs. For simplicity, only the relevant non-terminal links are shown.



- target node has a unary child = FALSE
- source node has a unary child, ignoring children with punctuation = FALSE
- target node has a unary child, ignoring children with punctuation = FALSE
- less than 1 confident (good) word alignment(s) going outside = TRUE
- less than 3 less confident (fuzzy) word alignment(s) going outside = TRUE
- less than 1 non-terminal alignment(s) going outside = FALSE
- less than 1 confident (good) word alignment(s) going outside, ignoring punctuation = TRUE
- less than 3 less confident (fuzzy) word alignment(s) going outside, ignoring punctuation = TRUE
- leaf ratio similarity score is less than 40; penalty score denominator is 80 = FALSE
- height difference is less than 4 = TRUE
- verb exists on only one side = TRUE
- pronoun exists on only one side = FALSE
- subtrees share at least one confident word alignment = TRUE
- source-side string has punctuation at beginning or end of span = FALSE
- target-side string has punctuation at beginning or end of span = FALSE
- begin and end words on both sides have links leading to well-formedness = FALSE

The aforementioned list can be clarified by looking at the example in the figure, with the NP and VP as the subtree roots:

- Clearly, these nodes do not have unary children, with or without taking punctuation into consideration. Therefore, all relevant values are 0.

- There are no confident word alignments going outside (value 0). However, between the subtrees, two fuzzy word alignments go outside: the ones linking *Bring* to *Geef* and *me*. The word *Bring* is a descendent of the English VP, but *Geef* and *me* are not descendents of the Dutch NP, meaning that the tree pair is not wellformed.
- There are less than 3 fuzzy word alignments going outside, so this value is 1.
- The leaf ratio similarity score is not less than 40 if the penalty score denominator is 80. In the example, the leaf ratio is 2/3. Using the formula discussed in chapter 6, we subtract the absolute difference (1) divided by the denominator (80) from the leaf ratio, giving us a value of roughly 0.65 (65). The feature value is therefore 0.
- The source tree height is 1, while the target tree height is 2. The difference is 1, which is less than 4. Therefore, the feature value is 1.
- There exists a verb on only one side, in the form of *Bring*. (value=1)
- It is not the case that a pronoun exists on only one side. (value=0)
- The trees share at least one confident word alignment, in the form of *de*↔*the* and *schep*↔*shovel*. (value=1)
- Neither the source nor the target trees have punctuation at either the beginning or at the end of their spans. The respective values are therefore 0.
- The beginning and end words on both sides do not all have links leading to well-formedness. The word *Bring* has two links that lead to non-wellformedness, since the words to which it links (*Geef* and *me*) are not dominated by either of the trees. (value=0)

During training, we count all occurrences of specific feature value sets in the automatic data set in order to compare them to the gold standard. We assign numbers to each unique feature profile and count whether or not the action ADD (in the case of unlinked node pairs) or REMOVE (in the case of linked node pairs) compares favourably with their equivalents in the gold standard, for every such feature profile.

For example, if there are 20 node pairs in the automatic set having the same set of binary values 00011011000111010, with 12 of them unlinked and 8 linked,

then the rule “if 00011011000111010 then ADD” can be applied 12 times and the rule “if 00011011000111010 then REMOVE” can be applied 8 times.

Since the gold standard has the same trees, it follows that the same node pairs exist as in the automatic data set. Some of them may be linked and some not. Transformation rule “if 00011011000111010 then ADD” are applicable 12 times to the automatic set. If we look at the same node pairs in the gold standard, we might notice that only 8 of them are linked, meaning that 8 rule applications are right and 4 are wrong. Similarly, transformation rule “if 00011011000111010 then REMOVE” can be applied 8 times to the automatic set, but we might notice that only 2 of the relevant node pairs in the gold standard are unlinked, meaning that 2 rule applications are right and 6 are wrong.

We proceed to discover possible rules and compare them to the gold standard until we have a complete list, each having a count of matches with their occurrences in the gold standard as well as mismatches. We proceed to sort them in descending order of the number of mismatches deducted from the number of matches.

For clarity, we present a possible example of a “best” rule:

163 00011011000111010 ADD 51 38 13

This means that the best rule for this iteration is the 163rd rule to be found. The binary number corresponds to the truth values of the features in the order that they are specified in the text file. ADD specifies that with all unlinked node pairs that are encountered where this combination of feature values are true, a link is to be added. 51 is the number of correct cases found in the gold standard, 38 are the number of incorrect cases, and 13 is the result of the number of incorrect cases subtracted from the number of correct cases. If no other rules have been found with a difference of more than 13, this rule would be selected to be applied for the current iteration.

In the iteration, the best rule found is added to the list of transformations and applied to the automatic data set to create a new one, from which we newly extract features and discover rules. This continues until no rule can be found where there is a positive difference between matches and mismatches.

Rule cutoff selection and testing

Experiments (see section 7.7) have indicated that this approach can easily lead to overtraining. This means that training scores tend to be high but score much lower on development test sets. To counter this, we introduce a phase where

the learned transformation rules are applied to a held-out set which has its own gold standard. We then select only the rules up until the one which, when applied, leads to the highest F-score when evaluated against the gold standard. This subset of rules will then be used for testing and/or aligning/correcting new data.

For example, 80 rules might have been learned. On the cutoff set, however, application of rule 60 has the highest F-score. We therefore select only the set up to and including rule 60. The order of the rules stay the same, but selecting a cutoff rule means that not all of them will eventually be applied to new data.

It may happen that with the cutoff rule, the next few rules lead to no change. In that case, we include them as well, since together with the training data set they still lead to a net improvement.

One may wish to apply the newly selected list of rules to a test set for evaluation purposes. Obviously, a gold standard for this set is needed as well. Instead of a test set, one may apply a development test set which can be used for various experiments, so that the test set is only used for the final evaluation process.

Processing new data

Any set of learned rules can be applied to new unseen data. It is important to take care that the input set has been processed to the same degree as the training, cutoff and test sets. For example, if one uses TBL to improve the output of an existing tree aligner, all automatic data sets, including the new one, must ideally contain the output of that tree aligner. Of course, it would be even better if all these sets were produced by the same method or model.

The current implementation is too slow for sequential processing of large data sets such as Europarl (proceedings of the European Parliament, see chapter 3, for example). However, we have adapted the system for use on our local high performance computing (HPC) cluster so that many alignment sets can be processed at once. For the sentence aligned Europarl 3 corpus, we have split the sentence pairs up into 650 files containing roughly 3000 sentence pairs each. Using TBL, it took less than two days for the entire parallel corpus to be aligned on the cluster.

7.6 Feature selection

While developing and testing the system on our data (see section 7.7), we have learned a few interesting properties of the system. We have found it useful to

construct a few complex rules by implementing a relatively small combination of features in order to limit the time for training and testing. This deviates from the approach of Brill (1995) who applies a large set of simple rules to the part-of-speech tagging problem. Nevertheless, our approach has some advantages and disadvantages. Some advantages are:

- The system runs more quickly because of the limited number of features.
- It is quite powerful and high scores are reached very quickly, since all feature values are combined, creating complex rules.

Some disadvantages are:

- Unlike simple rule systems, our system overtrains easily. Since the rules are complex, it is conceivable that some of them can only be applicable on the currently investigated data set. This can result in significantly lower scores in the testing phase on sets that are different from the current one.
- Feature selection must be done carefully in order to keep the number of features to a minimum.

We attempt to lessen the impact of the disadvantages by doing the following:

- Include a variety of sentence pairs from different domains in the training data set.
- Concentrate on the most significant features and only include these.
- Make use of a held-out set in order to select a cutoff point to minimise overtraining.

In our experiments, longer feature lists always led to overtraining issues, even when using a cutoff point. Therefore, our lists tended to be small, so we experimented with including and excluding various different features.

Some of the feature lists led to output with a high measure of precision, but had a relatively low coverage, such as checking whether a number exists on only one side, which usually means that the subtrees should be unlinked. Our data sets do not contain many numbers, so including this feature did not lead to much overall improvement. Therefore, this was not included in our final best list.

We also experimented with using different initial state annotators, which we will describe in section 7.7 in more detail.

Our best list which we will report here was already noted above in the form of an example of a feature profile. For clarity, we list them again without the true and false values (note that the specific order does not matter):

- source node has a unary child
- target node has a unary child
- source node has a unary child, ignoring children with punctuation
- target node has a unary child, ignoring children with punctuation
- less than 1 confident (good) word alignment(s) going outside
- less than 3 less confident (fuzzy) word alignment(s) going outside
- less than 1 non-terminal alignment(s) going outside
- less than 1 confident (good) word alignment(s) going outside, ignoring punctuation
- less than 3 less confident (fuzzy) word alignment(s) going outside, ignoring punctuation
- leaf ratio similarity score is less than 40; penalty score denominator is 80
- height difference is less than 4
- verb exists on only one side
- pronoun exists on only one side
- subtrees share at least one confident word alignment
- source-side string has punctuation at beginning or end of span
- target-side string has punctuation at beginning or end of span
- begin and end words on both sides have links leading to well-formedness

7.7 Experiments

Data

For our experiments, we make use of data sets that are part of large parallel corpora which are freely distributed online. These corpora are the same as those described in chapter 3, which were used in the context of the Parse and Corpus-Based Machine Translation (PaCo-MT) project. The data sets are in the same stage of processing as those used in our Europarl experiments described in chapters 4 and 6. In other words, we make use of strictly one-to-one aligned sentences that are parsed and then also word aligned by GIZA++ using the Moses heuristics of *intersect* (high precision) and *grow-diag* (high recall) alignments.

From the processed data, we constructed various alignment sets, all of them with Dutch as the source and English as the target language, using the Stockholm TreeAligner. We have kept the automatic word alignments intact, as we seek to emulate input conditions as they would occur when applying our models. In constructing our training data, we have tried to adhere to the following principles:

- We took the previously defined well-formedness constraints into account. However, we also made alignment decisions based on real-world knowledge of concepts to which the text refers. Additionally, we considered it important whether or not it would make sense to translate the source-side string into the target-side string in at least some contexts. We have applied the constraint involving one-to-one links only to those between non-terminals. Many-to-one and even many-to-many links between terminals occur frequently in cases where there is otherwise no satisfactory way to represent translational equivalence. Because of this and because we prefer high-recall over high-precision alignments (see discussion on previous extrinsic evaluations in section 7.10), we have allowed this.
- As mentioned before, we distinguish between so-called *good* and *fuzzy* links, which denote confident and less confident links respectively.
- A good link is made between subtrees where the string dominated by the target subtree would be considered a good translation in at least some contexts of the string dominated by the source subtree.
- A fuzzy link is made between subtrees where the string dominated by the target subtree would be considered a conceivable but less optimal trans-

English phrase	Dutch phrase	Translation	Reason
its activities	de activiteiten die daar plaatsvinden	the activities tak- ing place there	loose translation
to join us	tot de Unie toe- treden	to join the Union	loose but correct in context
the European Union	Europese Unie	European Union	incomplete but plausible translation
the story	de verhalen	the stories	loose translation
create opinion	onze opinies vor- men	create our opin- ion	more general translation

Table 7.4: Some phrases which have been linked using the fuzzy (less confident) link. Note that the translation direction is from Dutch to English, which makes the last example a more general translation instead of a more specific one.

lation of the string dominated by the source subtree. Examples would be where the target string (*the vehicle*) is more general than the source string (*the car*). A more general translation is still in a sense correct, but a more specific translation can easily be wrong, since the knowledge about the specifics is not necessarily present on the source-side. However, in cases where the target-side is more specific and this particular translation seems to be relatively frequent, we still align them using a fuzzy link. We also make fuzzy links, for example, where the translation is loose or where less important words, such as determiners, are left out on one side.

Figure 7.1 gives a good example of a subtree pair which should be confidently aligned. The phrases *zwaarwichtig en onrijp* and *ponderous and immature* are word-for-word translations where every terminal has a good link to a terminal governed by the other subtree. Conversely, there may be a number of reasons to prefer aligning subtrees using a fuzzy link. Generally, though, they are pairs which we have deemed as translationally less optimal, but acceptable. Table 7.4 presents such examples of phrases which we have deemed to align using fuzzy links.

Function	Domain(s)	Sentence pairs
Training set	EP (50), OPUS (150), DGT (50)	250
Cutoff set	EP (50), OPUS (35), DGT (15)	100
Development test set	EP (50), OPUS (150), DGT (50)	250
Test set	EP	200

Table 7.5: Summarisation of the data used in the experiments, where EP = Europarl = European Parliament Proceedings, OPUS = Open Parallel corpUS and DGT = Directorate-General for Translation translation memories of the JRC (Joint Research Centre).

As mentioned in the previous section, there are a number of different data sets that can be used in the experimental process, each having its own gold standard. We therefore proceeded to construct four gold standard sets from Lingua-Align output which was trained on a set of 140 parallel sentences from Europarl which we used in the PaCo-MT project. The sets consist of:

- A training data set comprising 250 sentence pairs from four different domains.
- A held-out data set comprising 100 sentence pairs from the same four different domains, which we use to determine a rule cutoff point, as discussed in the previous section.
- A development test set comprising 250 sentence pairs from the same four different domains.
- A test set comprising 200 sentence pairs from one domain (Europarl).

Our opinion is that the training and both test sets are the most important, therefore they all contain a relatively high number of sentence pairs. The development test set should be big enough to be a reliable guide for the feature engineering process. The test set, however, should be notably different in its domain composition than the training data set, in order to be a proper test of the flexibility of the model (see below). We regard the cutoff data set as the least important, therefore it contains the least sentences.

Table 7.5 summarises the above.

For the training and the development test set, the OPUS (Open Parallel corpUS) data (Tiedemann, 2009) is subdivided into OpenSubtitles (movie

subtitles, 100 sentence pairs) and EMEA (documentation from the European Medicines Agency, 50 sentence pairs). DGT (50 sentence pairs) denotes the DGT (Directorate-General for Translation) Multilingual Translation Memory of the JRC (Joint Research Centre) Acquis Communautaire.² In the cutoff set, the OPUS data are subdivided into 20 OpenSubtitles and 15 EMEA sentence pairs. We have aligned more sentences in OpenSubtitles since most of them are very short, although we have tried to include enough variation and negative examples as well (for example where parses are such that many constituents cannot be aligned).

Since all four data sets have both a corrected (gold) and an uncorrected version, this brings the total up to eight sets.

Initial state annotators

We have also experimented using different combinations of alignment steps. One alignment step is a bottom-up algorithm which we describe in chapter 6, which mainly uses the similarity features using geometric averages as described in the previous section. It can be described as follows:

- For every unlinked source-side non-terminal node, starting at the lowest level:
 - Inspect every unlinked target-side non-terminal node, starting at the first level.
 - If conditions hold between the current two subtrees, link them, else continue to the next unlinked target-side non-terminal node on the same level.
 - If no target-side non-terminal node has been linked on the current level, continue to the next level, and repeat the previous steps up to and including the highest possible level.
 - If the current source-side node has been linked, move on to the next source-side node on the same level, and start inspecting target-side nodes on the first level again.
 - If all source-side non-terminal nodes have been inspected on the current level, move on to the next level and repeat by inspecting unlinked target-side nodes on the first level, repeating all previous steps.

²<http://langtech.jrc.it/DGT-TM.html>

- Stop when all unlinked source-side non-terminal nodes have been processed.

The algorithm can also be applied in the other direction by starting from the target-side nodes instead of the source-side nodes. One can create different data sets by applying both directions to an existing set. Furthermore, one can create a higher precision set of the latter two sets by calculating and applying their intersection, or a higher recall set by doing the same with the union.

In our experiments, using the union set as input somehow always led to many more rules being learned, with relatively little increase in performance. We have therefore continued to experiment with these combinations (all sets were already word aligned):

- TBL
- bottom up + intersection + TBL
- Lingua-Align + TBL
- Lingua-Align + bottom up + intersection + TBL

This Lingua-Align model is the same one as mentioned before: a high precision but low recall model trained on a separate Europarl set, which we applied to all corpora in the PaCo-MT project. We have subsequently applied a version of the bottom-up algorithm to it, which increased recall as well as the BLEU score (see chapter 8). The fact that experiments with intersection data sets increased F-scores even further suggested that it would be a good idea to experiment with alignment in that order.

7.8 Results

We would like to point out that for our evaluation scores, we do not distinguish between good and fuzzy links, since we do not yet attempt to make that distinction in our output. Therefore, we use the standard balanced F-measure as it occurs in many other NLP publications, instead of the more refined definition as is used in the word alignment literature (Och and Ney, 2003), where the recall of only good links is taken into consideration.

The following formulae are used to determine F-scores. P denotes all correct links (“possible”, both good and fuzzy) and A refers to the automatically produced links.

Alignment steps	Precision	Recall	F-score
Intersect	91.2	47.6	62.6
TBL	88.3	78.3	83.0
Intersect+TBL	87.4	81.2	84.2
Lingua-Align+TBL	89.0	80.0	84.3
Lingua-Align+intersect+TBL	87.8	82.7	85.2

Table 7.6: Evaluation results on the development test set

$$Precision(A, P) = \frac{|P \cap A|}{|A|} \quad Recall(A, P) = \frac{|P \cap A|}{|P|} \quad F = \frac{2 * Precision * Recall}{Precision + Recall}$$

Table 7.6 displays our test results on the development test set using all evaluated combinations, while table 7.7 displays the same results on our Europarl test set. We do not include Lingua-Align scores here, because in order to be fair, we would also need to train a Lingua-Align model on the same training data set, which comprises four different domains. However, for Lingua-Align training we would also need to specify the correct GIZA++ models, and we do not have any single model that was trained on all four these domains. If we attempt to train a Lingua-Align model on the training data set specifying any single GIZA++ model, for example the Europarl model, recall drops substantially.

Therefore, we proceed to select sentence pairs from the domain which is represented the most in all our data sets, Europarl, and implement ten-fold cross validation using both Lingua-Align and TBL on its word-aligned version. We do the same with the pipeline that has led to the best scores on average: Lingua-Align + bottom-up + intersection + TBL. The set consists of 350 sentence pairs, on which for Lingua-Align we train 90% - i.e. 315 sentence pairs - and test on the rest, whereas for TBL, we train on 250 sentence pairs, use 75 sentence pairs as a cutoff set and test on 25. For Lingua-Align, we use the same set of parameters that we used in the PaCo-MT project, with an improved set of features that resulted in better scores for later experiments. For TBL, we use the same set of template instantiations that led to the scores above. We present the results of our evaluation in Table 7.8.

A number of things stand out when looking at our results. First of all, applying the intersection of the bottom-up algorithm generally leads to a high precision but a low recall. As suggested by experiments described in chapter 6,

Alignment steps	Precision	Recall	F-score
Intersect	79.3	51.8	62.7
TBL	78.2	79.3	78.7
Intersect+TBL	77.4	82.5	79.9
Lingua-Align+TBL	79.0	82.3	80.6
Lingua-Align+intersect+TBL	77.3	83.9	80.5

Table 7.7: Evaluation results on the Europarl test set

System / Pipeline	Precision	Recall	F-score
Lingua-Align	81	77.6	79
TBL	84.9	80.1	82.5
Lingua-Align+bottom-up+intersect+TBL	84.2	84.2	84.2

Table 7.8: Ten-fold cross validation scores applied on Europarl data for TBL and Lingua-Align

however, it is quite suitable for increasing the recall of data sets produced by Lingua-Align.

Secondly, all TBL experiments lead to very good results on the development test set and worse results on the test set. This is expected since the rules were learned while training on a set with similar data than the development test set, whereas the test set consists of only Europarl data. Generally, the combination consisting of Lingua-Align, the bottom-up intersection and TBL fares the best although the F-score of Lingua-Align+TBL is a 0.1 improvement on the test set. The scores of Lingua-Align+intersect+TBL and Lingua-Align+TBL are generally very close to each other, suggesting that the combination of Lingua-Align and TBL is more important than using the intersection with TBL.

However, the score from the Lingua-Align model with the intersection without using TBL is also quite high for the test set, as Table 7.9 indicates. Performance is inconsistent, however, as the F-score for the same combination on the development test set is much lower. Performance of Lingua-Align varies greatly between the sets as well. This is not surprising as the Lingua-Align model was trained on sentences from Europarl. However, note that Lingua-Align precision stays very high throughout.

System / Pipeline	Data set	Precision	Recall	F-score
Lingua-Align	devtest set	91.3	41.6	57.2
Lingua-Align+intersect	devtest set	85.1	57.5	68.6
Lingua-Align	test set	90.1	60.5	72.4
Lingua-Align+intersect	test set	81.3	76.9	79.0

Table 7.9: Evaluation scores of Lingua-Align with and without adding the intersection of the bottom-up algorithm, on both the test and development test sets.

Thirdly, the TBL experiments produce very high precision results on the development test set with lower but still good recall scores. But when applied to the test set, recall is always as least as good as precision. The differences are not that much but still noticeable. We do not yet have a clear explanation for this, but we speculate that the generalizing nature of the algorithm, together with the use of a cutoff data set, may be at least partly the reason for the relatively high recall scores.

Finally, it is clear that our ten-fold cross validation results favour TBL over Lingua-Align. The very high scores of the error correcting model (Lingua-Align+bottom-up+intersection+TBL) are consistent with those of the development test set and the test set. As a tree aligner, TBL achieves surprisingly high scores using only a very small set of features.

Figure 7.4 is a graphical presentation comparing the test set evaluation scores for the TBL experiments that we reported in Tables 7.6 and 7.7 as more rules are applied for each set. Note that the plots are of different lengths because of the fact that a different number of rules was applied in each case. It is clear that the first few rules are very effective in the case of TBL and intersection+TBL. Unlike with training, where the points are usually plotted in a shape resembling a logarithmic curve, with these two experiments, there are three distinct jumps in performance increase at more or less the same points on the x-axis. Note that we do not include the original scores in the graph (at $x=0$).

In the case of Lingua-Align+intersection+TBL, it starts off with a high initial score which increases only slightly after application of the rules. This is not that surprising since before applying TBL, Lingua-Align+intersect already has a very high F-score of 79.0 (see Table 7.9). However, Lingua-Align has a lower score of 72.4, but after just one rule (Lingua-Align+TBL), it achieves almost the

Figure 7.4: Comparison of the F-scores of the four different test sets.

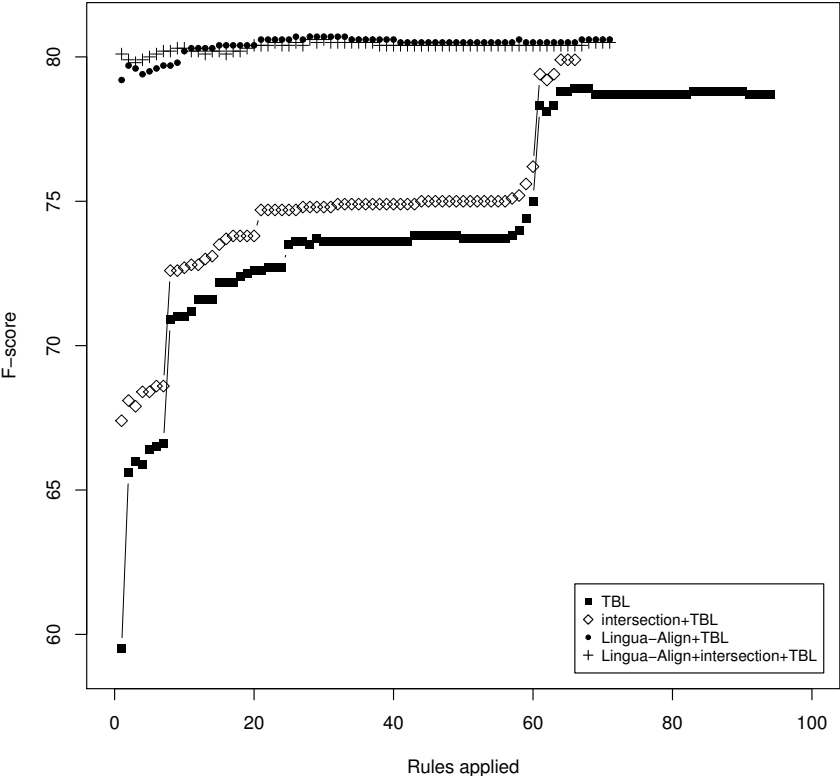
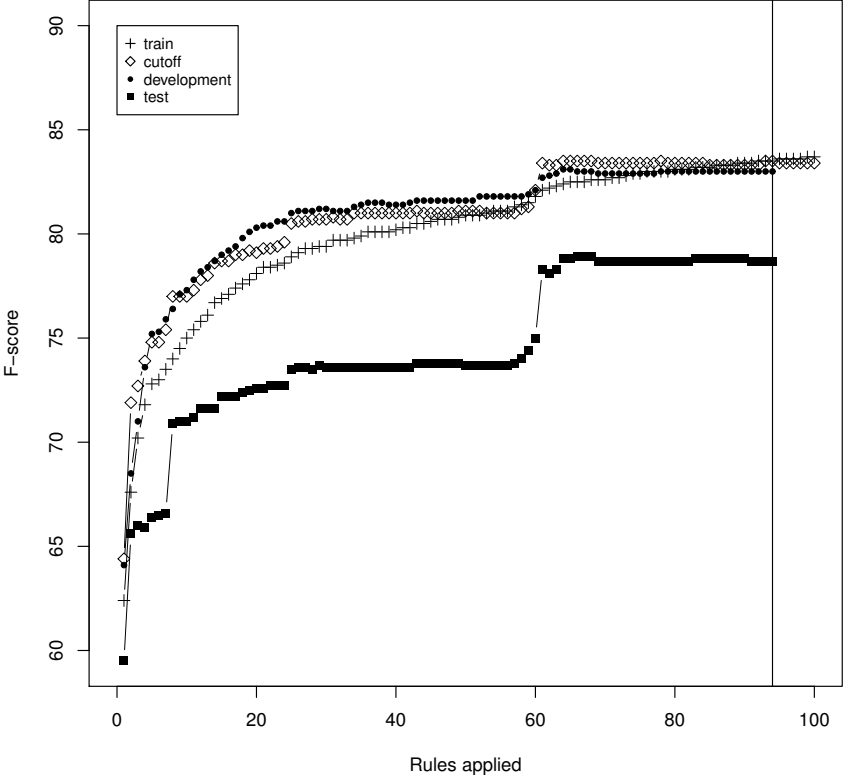


Figure 7.5: Comparison of the F-scores of the four different evaluation sets when the system is used as a tree aligner.



same as Lingua-Align+intersect+TBL, and also manages to stay on par during subsequent rule applications.

Figure 7.5 gives us some insight to what happens when we use the system as a tree aligner. We plot the F-scores for not only the test set, but also for the training, cutoff and development test sets. The vertical line shows where the cutoff point is, which is the maximum F-score obtained in the cutoff data set.

It is clear that the scores for the other sets are not only higher than those of the test set, but that they also more closely resemble a logarithmic function, the closest of which is displayed by that of the training data set. However, the points of the cutoff and development sets mimic the last jump in F-score improvement in the test set graph to a degree. It is also striking that after only a single rule application, F-scores jump to between 59.5 and 64.4. During training, the first rule adds 910 alignments of which only 132 are incorrect. However, test set results are consistently lower, which suggests overtraining, in spite of the measures undertaken to minimize its effect.

The first few rules applied are very similar, even when applied to the different experimental data sets. Most rules add alignments and significantly fewer remove them. This obviously holds more strongly when TBL is applied to a set containing just word alignments. With all four experiments, the first rule learned is the same. These are the features associated with this rule:

- The source and target roots are not in a unary relation, even when accepting unary parents that have punctuation as children (see section 7.4).
- No more than two fuzzy word alignments exist that lead to non-wellformedness (see section 7.4).
- No more than two fuzzy word alignments exist that lead to non-wellformedness, ignoring punctuation.
- No good word alignments exist that lead to non-wellformedness.
- No good word alignments exist that lead to non-wellformedness, ignoring punctuation.
- No non-terminal alignments exist that lead to non-wellformedness.
- The leaf ratio score is 40 or more, with a penalty score denominator of 80 (see section 7.4). Recall that the higher the leaf ratio score, the higher the similarity. However, we use a penalty score to take the leaf count difference into account, using a normalization value to avoid subtracting a too large amount. For more information, see section 6.3.
- The height difference is less than 4 (see section 7.4).
- A verb is either present on both the source and target-side, or none (see section 7.4).

- A pronoun is either present on both the source and target-side, or none.
- The subtrees share at least one good word alignment (see section 7.4).
- The first and last terminal of each subtree has a link that goes to a descendant terminal of the root of the other subtree (“inside”). In other words, these four terminals - the start and end terminals of the source and target-sides - all link to terminals dominated by the subtrees with the current source and target non-terminals as roots (see section 7.4).

Figure 7.6 - which also appears in the introductory chapter - displays an example of a set of trees conforming to the requirements set by the first rule.

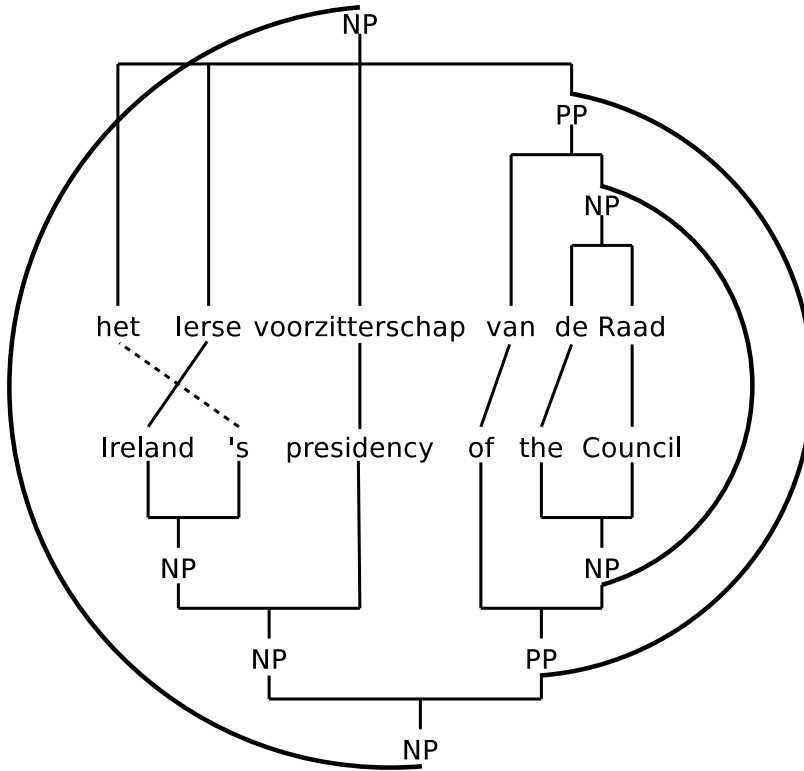
In all cases, the second rule is almost exactly the same. Subsequent rules follow similar patterns, allowing for certain conditions to be true where it was not the case in the first rule, for example that a verb is present on only one side. With all four experimental sets, the first REMOVE rule (situated at position 4, 5 or 6) is exactly the same. In addition, this rule differs in only one respect to the first ADD rule which is the removal of links where there is at least one non-terminal alignment leading to non-wellformedness. The fact that they do not differ that much makes sense, because in many cases we remove alignments that have been added before, meaning that they share a significant number of features that are also good for adding. REMOVE rules also noticeably become more frequent towards the end, but the scope of their application also drastically diminishes.

As an example, we look at the first few rules learned when training TBL on word-aligned data only. Five ADD rules are applied before the first REMOVE rule. The first ADD rule has the same values as described in the list above. We will use this as a point of reference.

Next, we look at the rules applied in subsequent iterations.

- The second rule adds links to subtrees with the same conditions as those described by rule 1, except that in this case, a pronoun exists on only one side. This is a bit counterintuitive, since one should expect correctly linked trees containing pronouns to have them on both sides. Apparently though, this is often not the case for this data set.
- The third rule is the same as rule 1, with the exception that there is only a verb on one side. Again, it seems that there were enough positive hits for this to be selected as a best rule.

Figure 7.6: An example of a set of subtrees linked by application of the first rule. Solid lines are confident (good), the dashed line is less confident (fuzzy).



- The fourth rule is the same as rule 1, with the exception that one or more good word alignments leading to non-wellformedness are allowed.
- The fifth and final ADD rule - before the first REMOVE rule - also adds links where the target-side node has a non-terminal unary child, where the source-side has a non-terminal unary child ignoring punctuation, and where both the source and target-side strings have punctuation at either the beginning or end of their spans. Here we can see that subtrees involving unary relationships and punctuation are globally speaking in a relative minority but that at some point, a significant number of them can also be correctly linked.

One way of viewing this is that the values of rule 1 are the feature archetype of a correctly linked node pair, where rule 2 to 5 are slight deviations which are less typical but still mostly correct, in decreasing levels of importance.

As mentioned before, the first REMOVE rule removes links between subtrees where there is at least one non-terminal node leading to non-wellformedness. Interestingly, none of the ADD rules described above have explicitly linked such subtrees. What happened here was that the addition of previous links changed the feature profiles of some ancestor node pairs so that after the iteration, they now have descendants with links going to the outside. This was possible since there is no well-formedness check on a global scale. In other words, when we link a subtree pair, we do not check whether or not this action would lead to their respective ancestors becoming non-wellformed, instead assuming that later REMOVE rules would take care of this.

Next, we look at the important performance jumps, which are clearly visible in figures 7.4 and 7.5, and focusing on using TBL on word-aligned data only. The first one occurs at the 8th rule. This addition rule leads to an F-score increase of 4.3 (from 66.6 to 70.9) in the test set. The features that differ from the first rule are the existence of punctuation at either the beginning or end of the string for both the source and the target-sides. Punctuation does not seem to be a negative feature in most cases - as opposed to outgoing links, for example - rather, it helps to identify certain types of trees such as those spanning the start and/or end of the sentences. In this case, we could view the tree pairs described by this particular feature set as a special case of those described by rule 1.

The last big jump occurs at rule 61, which is also an addition rule, leading to an F-score increase of 3.3 in the test set (from 75 to 78.3). In the development test set, this rule also leads to a jump, albeit smaller, with a 0.6 increase in F-score (from 82.1 to 82.7). It differs with rule 1 with respect to four feature values:

- The source-side node has a non-terminal unary child.
- The target-side node has a non-terminal unary child.
- The source-side has punctuation at either the beginning or end of the span.
- The target-side has punctuation at either the beginning or end of the span.

This clearly describes the sentence root nodes, many of which have unary children on both sides. Obviously, nodes dominating strings spanning the

whole sentence mostly include punctuation at either the beginning or end of the strings. Although we have mentioned before that we prefer to link unary children instead of their parents, for the sake of completeness we also link the parents if both of their children are already linked. Since in our experience this occurs in a significant number of sentences, this may explain the jump in performance.

In the cutoff data set, instead of one big jump, there are two smaller jumps at the same place: at rule 60 (0.8 F-score increase, from 81.3 to 82.1) and rule 61 (1.3 increase, from 82.1 to 83.4). Rule 60 differs from rule 61 only at the height difference feature value. This means that rule 60 links (mostly) top-level nodes where there is a height difference of 4 or more.

7.9 Error analysis

Apart from the evaluation scores that we have already discussed above, we would also like to present a qualitative analysis of the output errors with the help of the Stockholm TreeAligner.

In our opinion, the most interesting results would be those that were produced by applying TBL on the word aligned data sets, since all non-terminal alignments and omissions are made only by the system. In this way, we can rule out such errors made by Lingua-Align and the bottom-up algorithm. Furthermore, it would have been interesting to compare the different domains on a qualitative level, but most domains are only represented in the development test set, which we used as a means to improve our scores. Therefore, we will restrict our analysis to the Europarl domain which is the only one present in the test set.

In a number of cases, a link is made to the parent of the correct source or target node, meaning that a longer than necessary string is included on that side. This may be the result of the fact that non-wellformedness is allowed to a degree. Words can be wrongly aligned, for example in the case of a lack of a proper translation, ultimately leading to errors such as the above.

Figure 7.7 gives an example of this, which is an extract from the following sentence pair:

- Geachte dames en heren, geachte collega's, op 23 juni 1996 is de heer Andreas Papandreou overleden.
- Ladies and gentlemen, on 23 June 1996 Andreas Papandreou died.

The Dutch phrase literally reads: *Dear ladies and gentlemen, dear colleagues* , but only the words *Ladies and gentlemen* appear on the English side. One could argue that these two phrases could be aligned using a fuzzy link, but a better and more equivalent translation of *Ladies and gentlemen* already exists in the form of the string under CONJ, *dames en heren* (ignoring the punctuation). In the gold standard, therefore, we have decided to align CONJ with NP, but TBL model has aligned the two NPs instead.

In the figure, it would appear that the high recall word alignment scheme aligned the untranslated words, such as *collega's* (colleagues), using fuzzy links, as indicated by the dotted lines. This leads to errors such as *collega's* being linked to *gentlemen*. If punctuation is ignored, the NP pair is completely well-formed. The pair of CONJ/NP can be regarded as nearly well-formed with three illegal fuzzy links, and two if punctuation is ignored - a still quite conceivable pairing.

In other cases, the causes of errors are initially less clear. For example, figure 7.8 displays a series of embedded subtree pairs which are linked in the gold standard but not in the TBL output. The sentences are:

- De heer Papandreou heeft er in beslissende mate zijn deel aan bijgedragen om Griekenland na de donkere jaren van het kolonelsregime terug te leiden naar vrijheid en democratie.
- Andreas Papandreou played a vital part in restoring freedom and democracy in Greece after the dark years of military rule.

All these tree pairs are well-formed, contain confident links and are very similar with respect to their leaf counts, but they remain unlinked. The Dutch only differs with *het kolonelsregime*, which literally translates to *the colonel's regime* instead of the more general *military rule*. Probably not coincidentally, these words are the only ones that are not aligned, although fuzzy links shared between *kolonelsregime* and *military rule* would have been correct.

This state of affairs leads to the fact that *het kolonelsregime* is not aligned with *military rule*, since they do not share any word alignments. The node pair comprising their respective parents (PP/PP) has a few good features: they share at least one confident word alignment, they have the same height, a perfect leaf ratio and they are well-formed. In fact, they share all feature values of the first addition rule apart from the requirement that the first and last word in both strings (*kolonelsregime* and *rule*) should also be linked to descendants of the respective other side (the last feature listed above). Their respective parents and grandparents (NP/NP and PP/PP) also share exactly the same features.

Figure 7.7: An example of a wrongly linked subtree pair, where the parent (NP) of the correct node (CONJ) is linked. The solid lines between the words are confident alignments and the dotted lines are fuzzy.

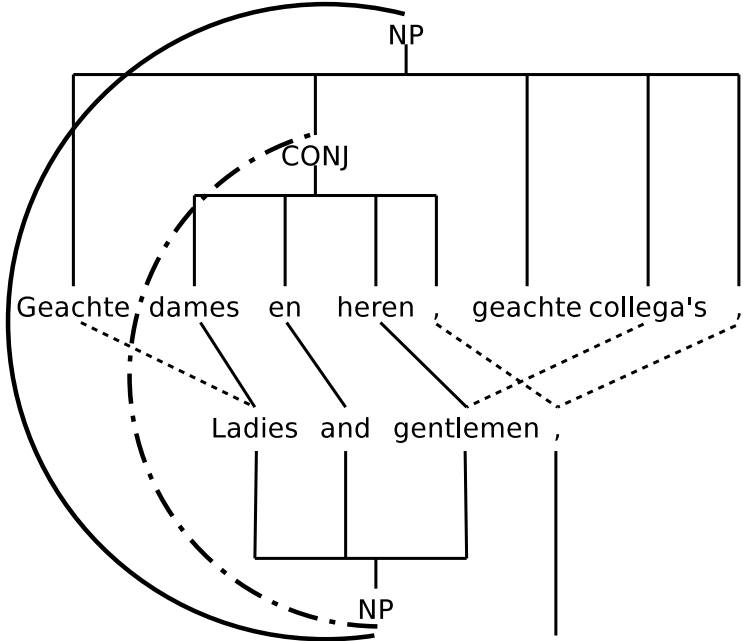
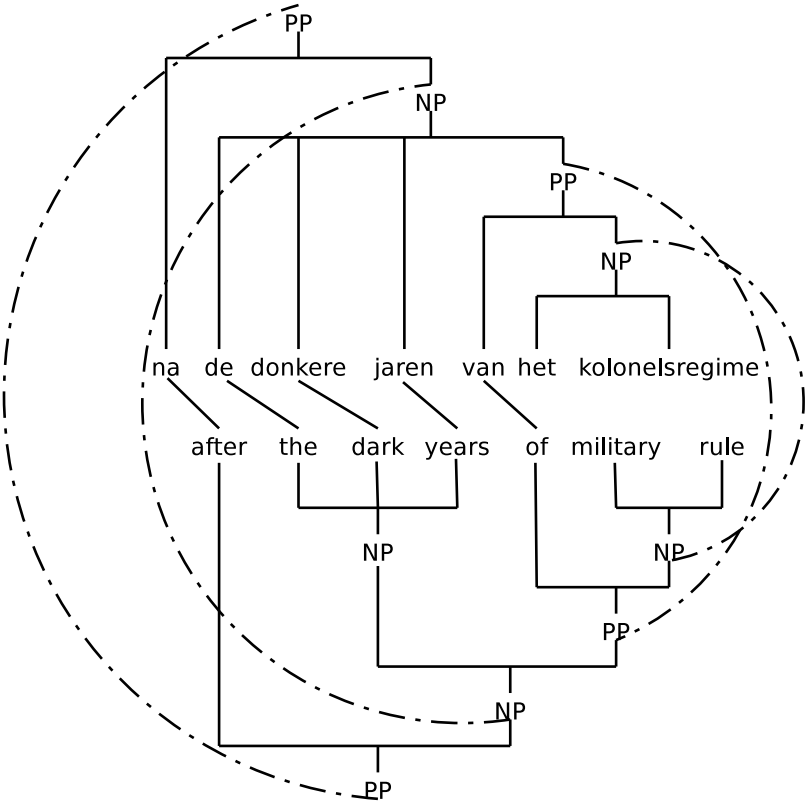


Figure 7.8: In this figure, non-terminal links indicated with dotted lines were linked in the gold standard but not by TBL.



This particular difference was in fact the deciding factor which kept these trees unlinked, since no addition rule actually exists which are applicable to these trees.

More such examples can be found such as the lack of a link between the NP *democratic Greece* and the NP *het democratische Griekenland* (*the democratic Greece*) with all words but *het* being aligned. Again, *het* as the first word in the source-side string being unlinked leads to this particular subtree pair not being considered for alignment during the application of addition rules.

As discussed in section 7.4, we have found that this particular feature may be a useful indication of equivalence. However, it naturally also excludes many possible candidates since not all equivalent trees have to be linked in this way. Since in our experience, using high precision, low coverage features such as checking for numbers may lead to a drop in overall performance, perhaps excluding or changing this feature may be a better option in the future.

Finally, there are errors which are very difficult to solve for various reasons. The most obvious errors are those that have occurred in previous stages of the parallel treebank creation pipeline: during tokenization, part-of-speech tagging, parsing, sentence alignment or word alignment. Words may be spelled or translated incorrectly or there may even be encoding issues. The only solutions for this are to focus on improving the accuracy of these tasks and to minimize their impact if possible.

However, some errors will persist even if one takes care of these problems. Many if not most sentences are not exact translations of each other: words are left out or added, even entire phrases, references are made to entities not in the text yet they are explicitly named on the other side, one language may make use of more idiomatic expressions than the other, and so on. Sometimes one needs to make the decision on whether or not to link a tree pair based on one's real world knowledge of the concepts to which these phrases refer. In these cases, it is very hard if not impossible to formulate a concrete set of features, apart from those that we have described before, in order to automatically make these decisions.

With this in mind, we recognize that a lack of deep semantic information in the data may limit general alignment accuracy to a degree. In this regard, we suggest the future incorporation of external resources such as wordnets.

7.10 Future work and conclusions

We have presented a system which adapts the powerful transformation-based learning algorithm to the problem of tree-to-tree alignment. In this chapter, we have described a set of experiments that were run on various data sets. All experiments involving our TBL implementation produced very satisfying results.

We must stress that it might be possible to train a better Lingua-Align system on the same data using the right feature and parameter sets. However, extensive feature engineering on Lingua-Align has not yet produced the desired results, whereas with our TBL system, we managed to reach higher scores relatively quickly. We can conclude that to the best of our knowledge, the results of our TBL experiments rival and in some cases surpass the best scores that were produced by Lingua-Align.

At this point in time, it is premature to compare our system to other aligners such as the Dublin aligner, which has been evaluated on different data sets. However, a series of ten-fold cross validation experiments using the same alignment sets and word alignment files as used for Table 7.8 would provide more clarity on the strength of its performance.

We have gone to reasonable lengths to ensure that experimental data sets contain a variety of parsed sentences from different domains. Based on the good results discussed above, we are reasonably confident that we have a working system with the ability to produce quality alignments for Dutch to English, which should also be fairly easily adaptable for other language pairs.

Currently, our system can still be better optimized for faster experimentation. Every iteration involves much input and output in the form of feature extraction and writing statistics to files. Various possibilities exist to optimize this step.

Our system also does not currently generate template instantiations. This means that there is much more manual work to be done when experimenting. Methods to achieve this without significant sacrifice of performance include random sampling using Monte Carlo methods (Samuel, 1998) and making more independence assumptions (Ngai and Florian, 2001).

Currently, our system does not add links with differing degrees of confidence, which may play a significant role in the performance of statistical MT systems.

Obviously, experimenting with additional features may improve performance in general. For example, classifying links that lead to non-wellformedness may be useful. A verb being linked this way - in other words, where the link

goes outside - may be regarded as a more negative feature than a preposition, for example.

An example of a feature that can be extended is the one which checks for the existence of good word alignments between the subtrees. Instead, shared non-terminal alignments can be included, with the option of not only allowing confident alignments. In fact, checking for such shared links was one of the strong positive features learned by Lingua-Align in our experiments described in chapter 4.

We could also incorporate GIZA++ word alignment probability features which are used by Lingua-Align and the Dublin aligner. Lingua-Align also makes use of features including looking at the position of the nodes with respect to the whole tree span, normalized instead of absolute height differences, more general category label and part-of-speech combinations, parents, grandparents, children, grandchildren, sisters, edges, and so on.

At some point, we would be interested to create separate templates for improving word alignments, which we can use in conjunction with those that we use for non-terminal alignments. The idea is to extend the system to one that can be applied to general alignment correction in a single step. Manipulation of trees for improved alignment convergence is an additional interesting idea that could be implemented, as described in Burkett and Klein (2012).

In the next chapter, we discuss the impact of our data on the performance of two machine translation systems: Moses and PaCo-MT.

Translation quality evaluation based on alignment output

8.1 Introduction

In previous chapters, we have discussed various approaches to the problem of tree-to-tree alignment. For Dutch to English, we have established that the flexibility of using maximum entropy models is well complemented by both rule-based heuristics and, especially, the rule-based machine learning approach called transformation-based learning. TBL has also been shown to work very well on its own, assuming relatively accurate high-recall word alignments and given proper feature engineering.

Using these methods, we have produced a number of different versions of a set of large parallel treebanks, both during the PaCo-MT project (chapter 3) and after. These resources can be used in a variety of ways for research and development. Of particular interest to us is the effect on syntax-based MT. Although our focus in this work is to improve alignment quality, by and large such improvement should also lead to an improvement in translation quality. For this reason, we present all relevant results achieved by our MT system using various evaluation metrics in this chapter. We also compare our results with

those achieved by a phrase-based SMT model trained on the same data, using Moses (Koehn et al., 2007).¹

In the next section, we present all relevant evaluation scores. After that, in section 8.3, we discuss the results of a qualitative analysis on some example sentences. Finally, we conclude in section 8.4.

8.2 Translation quality evaluation

Our chosen MT evaluation metrics are BLEU, NIST and TER, which are widely used in the evaluation of MT worldwide.

BLEU (Papineni, 2002), short for *bilingual evaluation understudy*, is a metric of which the authors claim high correlation with human evaluation. The general idea is that the closer the automatic translation is to the human translation equivalent, the better it is. For calculation, it requires a reference corpus of human translations. A numerical metric is applied which is based on n-gram occurrences of matching words.

Over the years, BLEU has garnered some criticism. Koehn (2011) lists some of the main arguments:

- The fact that some words are more important than others is ignored.
- No attention is paid to grammaticality.
- The BLEU score is very dependent on factors “such as the number of reference translations, the language pair, the domain, and even the tokenization scheme used to break up the output and reference into words”. (Koehn, 2011, p. 2)
- Experiments have shown that human translations do not score higher than MT output according to BLEU, even though the human translations are clearly better.

Because it is still the most widely used metric, we will also use it here.

The NIST (National Institute of Standards and Technology) metric is based on the BLEU metric, but some changes are implemented (Doddington, 2002). For example, lesser occurring n-grams are perceived as more informative and are scored higher.

¹Our sincere thanks go to Dr Vincent Vandeghinste of the Catholic University of Leuven in Belgium who ran the evaluation metrics and did most of the training, including for Moses.

Translation Edit Rate (TER), also called Translation Error Rate,² is a measurement of the amount of editing that has to be done to an automatic output to equal the correct human translation (Snover et al., 2006). Since fewer edits suggests a better translation, the lower scores are better.

Original versions of the PaCo-MT system produced relatively poor evaluation results, as, for example, reported in Vandeghinste and Martens (2009). Using a bottom-up transfer approach instead of a top-down approach led to much better scores (Vandeghinste and Martens, 2010). A tree-based target language generation component was also developed during the project (Vandeghinste, 2009). Over time, newer and better versions of these systems were developed.

During the project, we have produced parallel treebanks for the language pairs of Dutch to English, English to Dutch, Dutch to French and French to Dutch. For all MT experiments, we used the same data sets. As described in chapter 3, parallel corpora were preprocessed by tokenizing, sentence aligning, word aligning and parsing the texts. A few structural changes were made to Dutch trees to improve convergence with the English and French trees and to make them more uniform. The last step was to align the non-terminal nodes in Alpino-XML format.

We will only report the Dutch-to-English results for this work, since in the project, we have spent much more time working on refining our approach for this language pair, and as a result, the scores for the other language pairs are somewhat lower. For more information, the abovementioned papers can be consulted. As reported in chapter 3, a bilingual dictionary was used which in most cases has led to slight improvements. All reported evaluation scores are calculated against a test set consisting of 500 sentences and three reference translations.

To illustrate the improvement of various versions of the system during the project, we present in table 8.1 the best evaluation scores reported during those stages of models trained on Europarl, as well as the results of a 5-gram phrase-based model of Moses trained on the same data, using the default settings without optimization.³ The third line denotes the scores derived from a model trained on parallel treebanks created by the high-precision Lingua-Align model described in chapter 4, while the fourth line is associated with high-recall

²<http://www.cs.umd.edu/~snover/tercom>

³Although there exist other state-of-the-art candidates in the SMT paradigm such as Moses Chart or HIERO-style systems such as Joshua, the phrase-based approach of Moses remains among them as a worthy competitor.

Transducer	Alignment approach	BLEU	NIST	TER
V&M (2009)	Precision	0.1353	5.70	70.36
V&M (2010)	Precision	0.2065	6.44	63.72
PaCo-MT	Precision	0.2228	6.43	64.14
PaCo-MT	Recall	0.2548	7.36	61.12
Moses	N/A	0.4174	8.79	42.92

Table 8.1: Evaluation scores on various models trained on Dutch-to-English Europarl

alignments produced by a source-to-target version of the bottom-up algorithm described in chapter 6.

The improvement resulting from the switch from top-down to bottom-up transfer (first two lines) is quite striking. However, as expected, the state-of-the-art system Moses outperforms our efforts by a large amount.

Note that the model trained on the high-recall alignments leads to a significant improvement. We have not had the opportunity to train models based on other heuristics such as the intersection or the union of the source-to-target and target-to-source approaches of the algorithm.

After implementing our transformation-based learning system (chapter 7), we have experimented with training PaCo-MT models on various alignment combinations. They are:

- Lingua-Align (high precision) + bottom-up algorithm, using the intersection of source-to-target and target-to-source alignments + transformation-based learning (LABIT)
- Lingua-Align + bottom-up algorithm, using the intersection of source-to-target and target-to-source alignments + transformation-based learning + union of source-to-target and target-to-source alignments (LABITU)
- Lingua-Align + bottom-up algorithm, using the intersection of source-to-target and target-to-source alignments + transformation-based learning + union of source-to-target and target-to-source alignments + all fuzzy non-terminals turned to good (probabilities changed from 0.4 to 0.8; LABITU-FG)

The reason for the creation of the last alignment set is that the high-precision Lingua-Align model often seemed to assign fuzzy links with a low probability

to subtree pairs which should have higher probabilities (and none to those that should have fuzzy links). Since the TBL system does not work with probabilities and only attempts to remove wrong alignments (if discovered), these are mostly intact. We have felt that this might lead to a reduced performance in translation, and we therefore decided to change all probabilities between non-terminal nodes to 0.8. The plan was to apply this to both LABIT and LABITU (i.e. LABIT-FG and LABITU-FG), but since LABITU-FG scored much worse, we did not finish the training on LABIT-FG.

The reason for the creation of the second last alignment set (LABITU) was that high-recall alignments seem to lead to better scores. The union of the bottom-up algorithm added the most number of new alignments, while TBL seemed to be quite accurate. We have therefore decided to choose, according to our intrinsic evaluation experiments, the most successful TBL setup (LABIT) and adding the union alignments to its output.

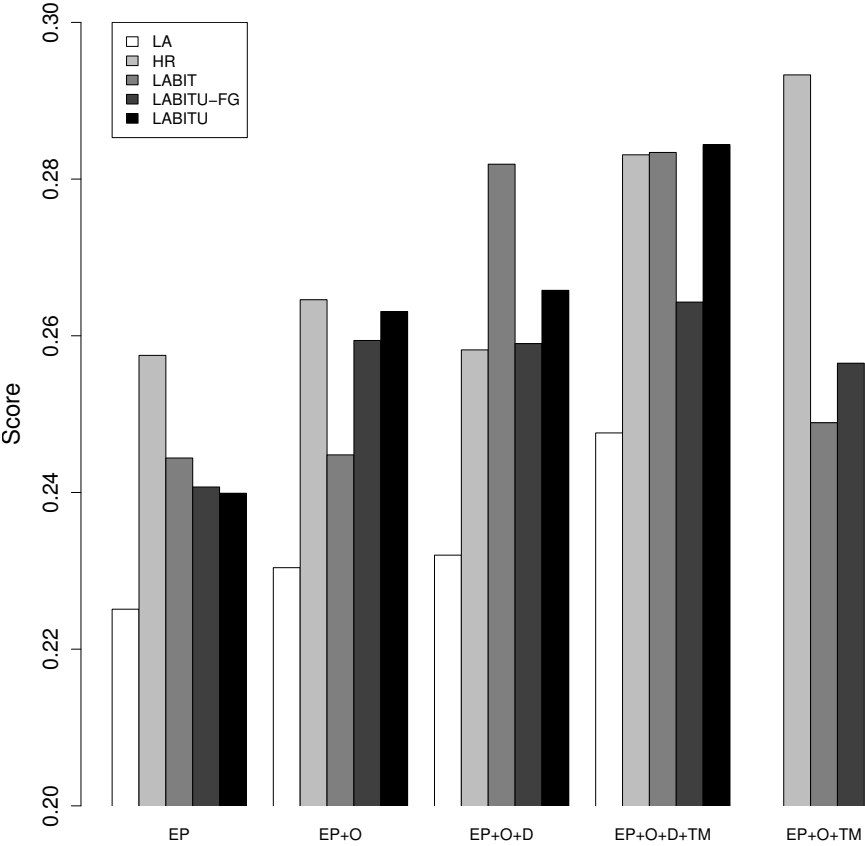
We now present evaluation scores for all the alignment sets, including the high-precision alignments from Lingua-Align (LA) and the high-recall bottom-up alignments (HR). Separate scores are also presented for models trained on the following corpus combinations:

- Europarl (EP)
- Europarl + OPUS (EP+O)
- Europarl + OPUS + DGT translation memories (EP+O+D)
- Europarl + OPUS + DGT translation memories + OneLiner translation memories (EP+O+D+TM)
- Europarl + OPUS + OneLiner translation memories (EP+O+TM) (not including DGT)

In Table 8.2 we present the BLEU scores. Note that some scores for equivalent setups are different than those presented in Table 8.1 because of the fact that better versions of the transducer and target language generator were developed after the conclusion of the project. “N/A” values are reported for evaluations we were not able to obtain. A bar chart of the same scores is presented in figure 8.1.

The high-precision approach (LA) is inferior in comparison and the bottom-up approach (HR) clearly better, producing the best score of 0.2933. Interestingly, adding DGT (third line) to the model leads to a decrease for HR. With the

Figure 8.1: Barcharts of the BLEU scores for models trained on different combinations of alignments and corpora.



Alignment set	LA	HR	LABIT	LABITU-FG	LABITU
EP	0.2251	0.2575	0.2444	0.2407	0.2399
EP+O	0.2304	0.2646	0.2448	0.2594	0.2631
EP+O+D	0.2320	0.2582	0.2819	0.2590	0.2658
EP+O+D+TM	0.2476	0.2831	0.2834	0.2643	0.2844
EP+O+TM	N/A	0.2933	0.2489	0.2565	N/A

Table 8.2: BLEU scores for models trained on different combinations of alignments and corpora. N/A means that the scores for those particular setups were not calculated. The best score for each line is in bold.

exception of LABITU-FG, where the score decreases from 0.2594 to 0.2590, this pattern does not repeat itself with other alignment combinations, where more data leads to better scores. Conversely, adding DGT to LABIT leads to a relatively large increase of 0.0371. It is especially interesting considering that for other models on EP+O+D, there is very little increase or even a decrease in performance. The reason for this discrepancy remains unclear, but the fact that we have used some material from DGT as training data for our TBL system may explain some of it.

Some other noticeable jumps in performance are:

- adding TM to HR with EP+O+D (0.0249)
- adding TM to HR with EP+O (0.0351)
- adding OPUS to LABITU with EP (0.0232)
- adding TM to LABITU with EP+O+D (0.0186)

On average, the scores increase by the following when these corpora are added to the training:

- adding OPUS to EP: 0.01094
- adding DGT to EP+O: 0.00692
- adding TM to EP+O+D: 0.01318

We exclude mentioning the scores where TM to EP+O was added since not all models have a corresponding score. Taking all models into account, adding

Alignment set	LA	HR	LABIT	LABITU-FG	LABITU
EP	6.48	7.43	7.22	7.2	7.25
EP+O	6.61	7.44	7.22	7.32	7.43
EP+O+D	6.70	7.28	7.64	7.29	7.39
EP+O+D+TM	6.93	7.58	7.66	7.37	7.62
EP+O+TM	N/A	7.71	7.27	7.31	N/A

Table 8.3: NIST scores for models trained on different combinations of alignments and corpora. N/A means that the scores for those particular setups were not calculated. The best score for each line is in bold.

TM to EP+O+D has on average the largest positive effect. This is closely followed by adding OPUS to EP. However, we did not test for all possible conditions such as training on EP and DGT first and then adding OPUS or TM, for example. The performance of TM is especially interesting when one takes into account the fact that this is the smallest parallel corpus from the four⁴ and being a corpus of translation memories, often does not even contain full sentences.

The BLEU results also suggest that high recall generally works. Not only is HR better than LA but the very high recall LABITU produces the second highest score (28.44) and the highest when DGT is incorporated into the model.

Finally, the experiment where the probabilities of fuzzy non-terminal alignments were increased did not succeed. In most cases, its BLEU score is lower than other alignment sets involving TBL, in spite of adding the union of the bottom-up algorithm in order to increase its recall.

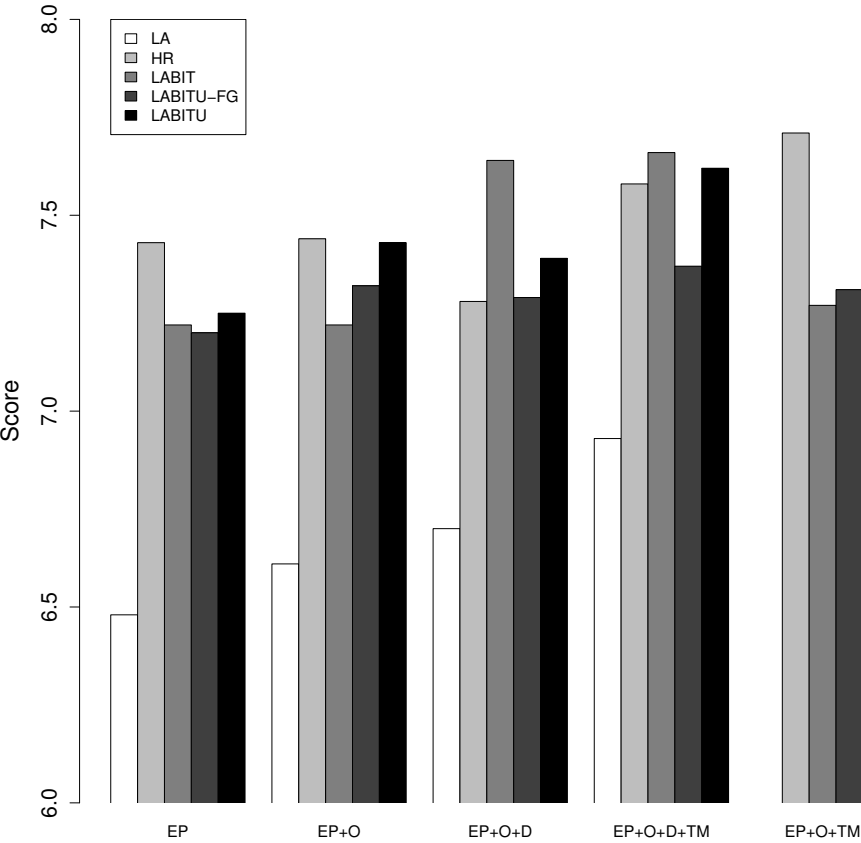
Table 8.3 displays the NIST scores for the same models, represented graphically by figure 8.2.

The results are almost identical in comparison. In this case, however, LABIT (and not LABITU) scores higher than HR when trained on all data, inching ahead by 0.08. But yet again, when an HR model is trained on all corpora excluding DGT, it scores better, with 7.71, a difference of 0.05. Interestingly, this time only LA and LABIT experience an improvement with the addition of DGT. The latter experiences a relatively big boost with a difference of 0.42. With the exception of HR, though, all models still obtain their best scores when trained on all corpora.

Here are the average improvements per line for the NIST scores:

⁴366,161 sentence pairs, as opposed to 1,180,706 sentence pairs in Europarl, 731,673 in OPUS and 478,972 in DGT.

Figure 8.2: NIST scores for models trained on different combinations of alignments and corpora.



Alignment set	LA	HR	LABIT	LABITU-FG	LABITU
EP	64.02	60.83	61.26	61.35	61.45
EP+O	64.53	61.42	61.43	61.34	60.98
EP+O+D	65.04	61.83	59.32	61.50	60.58
EP+O+D+TM	63.71	60.41	59.04	61.3	59.12
EP+O+TM	N/A	59.98	61.24	61.09	N/A

Table 8.4: TER scores for models trained on different combinations of alignments and corpora. N/A means that the scores for those particular setups were not calculated. The best score for each line is in bold.

- adding OPUS to EP: 0.088
- adding DGT to EP+O: 0.056
- adding TM to EP+O+D: 0.172

Again, TM seems to have the most positive influence, followed by OPUS and then DGT.

Finally, we present the TER evaluation scores and their graphical equivalent in Table 8.4 and figure 8.3.

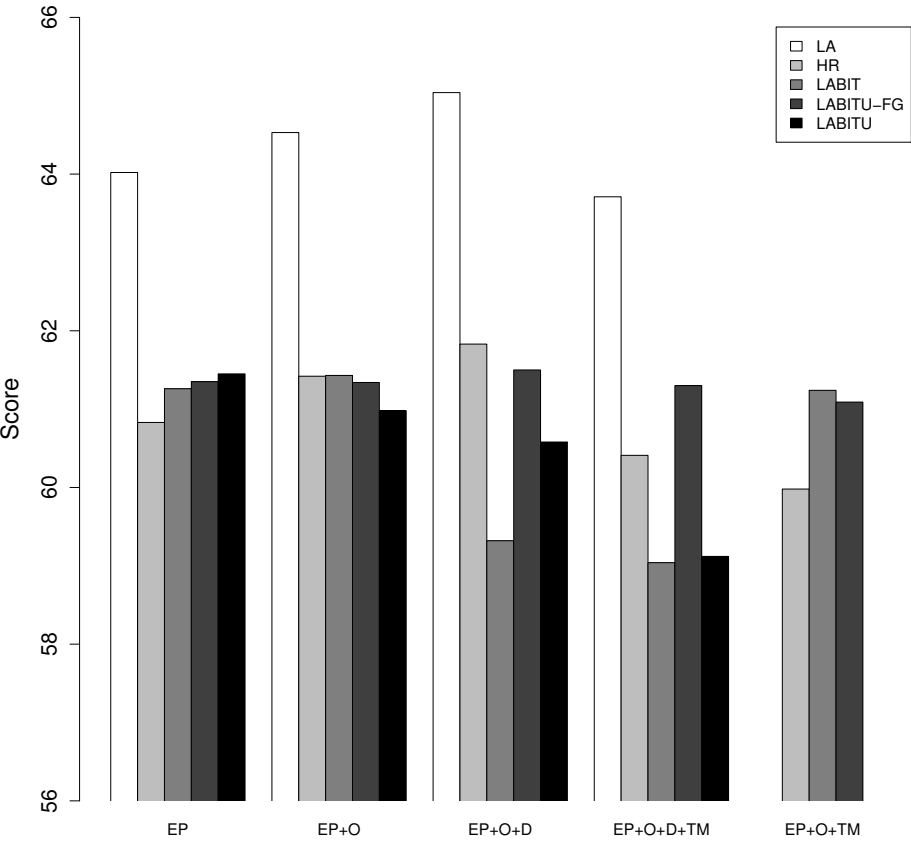
For this metric, the best scores are produced by LABIT and LABITU - this includes HR trained on everything but DGT (59.98). This time, adding OPUS also worsens the score in some cases (LA, HR and LABIT), just as with DGT (LA, HR and LABITU-FG). However, the best scores for all models are still produced by training on all the data.

Here, we find the average contribution of adding corpora to be somewhat different:

- adding OPUS to EP: -0.79
- adding DGT to EP+O: 0.286
- adding TM to EP+O+D: 0.938

Yet again, TM wins, and the difference is even more pronounced. Furthermore, adding OPUS has a net negative effect. It would be interesting to observe TER scores when the models are trained on everything but OPUS (EP+DGT+TM). In any case, this shows the effect that using different metrics

Figure 8.3: TER scores for models trained on different combinations of alignments and corpora.



can have on evaluation results. It is not clear why TER and BLEU/NIST differ so much in this regard.

To lend strength to our conclusions, we run significance tests on a few MT output sets. Zhang and Vogel (2004) present a method whereby confidence intervals for BLEU/NIST scores can be calculated using a bootstrap method where sentences in the hypothesis and reference sets are replaced with others in the same sets, leading to slightly different sample sets for each iteration. For each different set, the score is calculated, producing a distribution of different scores.

Our top three models that are based on high-recall alignments, HR, LABIT and LABITU, are similar in terms of their performance. We therefore suspect that their differences would not be significant. However, using the bootstrap method, we can show that each of them is significantly better than the high-precision Lingua-Align model. Formally, we aim to reject the null hypothesis that the three models are not significantly better than the high-precision model using a confidence interval of 95%.

First, log files are created for each translation hypothesis file associated with a model. These are used as input for the bootstrap script. As a pre-processing step, text normalization is applied. Since different normalization methods are required for the BLEU and NIST metrics, we create separate log files for each one.

The log files also produce BLEU and NIST scores for each model. In all cases, BLEU scores are slightly lower than those reported by our script, version 11b of "mteval.pl", which itself is widely used for evaluation. The script which produces the log files for bootstrap⁵ is "in accordance with mteval-v11 of NIST", according to the tutorial page.⁶ However, it is not clear if this means that it should produce the same results as our script. It also contains two distinct text normalization subroutines, one for BLEU and one for NIST, as opposed to just one used in "mteval". Replacing them with the one in "mteval" leads to only slight changes. It therefore seems that there are other factors at play here. We have decided instead to just note the differences and report the results of our analysis consistently. We will therefore compare only the scores produced by "generateLog" to each other, instead of the ones produced by "mteval".

We proceed to apply the bootstrap method to create a sample set of 1,000 hypothesis sets for each model, for which the BLEU and NIST scores are calculated. Using a confidence interval of 95% and assuming a normal distribution,

⁵<http://projectile.sv.cmu.edu/research/public/tools/bootStrap/generateLog-v11.pl>

⁶<http://projectile.sv.cmu.edu/research/public/tools/bootStrap/tutorial.htm>

Model	HR (EP+O+TM)	LABIT (all)	LABITU (all)
Median	0.2783	0.2712	0.2727
Mean	0.2786	0.2713	0.2724
95% Conf. Int.	[0.2604,0.2971]	[0.2537,0.2890]	[0.2523,0.2914]
STDEV	0.0097	0.0090	0.0095

Table 8.5: Statistics extracted from the results of the 1,000 iterations created using the bootstrap method, where “STDEV” denotes standard deviation and “Conf. Int.” denotes “Confidence interval”. BLEU scores are used.

Model	HR (EP+O+TM)	LABIT (all)	LABITU (all)
Mean	7.7075	7.6550	7.6062
95% Conf. Int.	[7.4577,7.9550]	[7.3391,7.9366]	[7.3033,7.8937]
STDEV	0.1274	0.1500	0.1522

Table 8.6: Statistics extracted from the results of the 1,000 iterations created using the bootstrap method, where “STDEV” denotes standard deviation and “Conf. Int.” denotes “Confidence interval”. NIST scores are used.

we produce the statistics displayed in Tables 8.5 and 8.6 for the high-recall models.

Table 8.5 shows that the medians and means of the different models are very similar, showing that skew is not a problem. Therefore, we omit reporting the medians for the other metrics.

For the model based on the high-precision Lingua-Align alignments, a BLEU mean of 0.2361 and a NIST mean of 6.9375 are reported. These are well below the minimum values of the confidence intervals of all three high-recall models in Tables 8.5 and 8.6, meaning that we can reject the null hypothesis with a p-value of less than 0.025. In fact, since the means of the high-recall systems are more than four standard deviations better than the baseline system, we can reject the null hypothesis at a level of $p < 0.001$.

For the sake of completeness, we also bootstrap the output of our best Lingua-Align model, trained on all corpora. The results are displayed in Table 8.7.

As for TER, we have not been able to run tests of significance. Although LABIT and LABITU outperform HR in our TER evaluation, the differences remain relatively small.

Model	LA BLEU	LA NIST
Mean	0.2361	6.9375
95% Conf. Int.	[0.2185, 0.2537]	[6.5471, 7.3081]
STDEV	0.0087	0.1934

Table 8.7: Statistics extracted from the results of 1,000 iterations on BLEU and NIST log files of high-precision LA, created using the bootstrap method.

Our results show that high-recall alignments improve PaCo-MT models. Although HR seems to be the best model, it is not better than the other high-recall models to a statistically significant degree. It also contains fewer phrases than LABIT and LABITU (see next chapter). It may well be that there is more variation in performance of our TBL system between the treebanks of different domains, whereas the performance of the simplistic bottom-up heuristic should not vary too much between the domains and therefore produce more consistent results. This seems to be true for almost all data except the DGT treebanks, the effect of which remains a mystery for now.

8.3 Qualitative analysis

We conduct a short qualitative analysis of the MT output in the test set, by selecting a few random translations with differing themes and grades of complexity. Translations are selected from the best models: Moses, HR (trained on all but DGT), LABIT and LABITU.

Note that punctuation is not handled explicitly by PaCo-MT and this is assumed to be done at a later stage. For both PaCo-MT and Moses, the same holds for capitalization.

Table 8.8 displays the first example. The first line contains the Dutch sentence to be translated. The following two lines contain two English reference translations, made by professional translators. The rest contains the automatic MT output.

In this case, most translations are almost correct. HR and LABIT contain the erroneous *itself*, a direct translation from *zelf*, which in this case should correspond to *own*. However, note that the Moses output is ungrammatical.

In Table 8.9, we present an example of a sentence where Moses provided the better translation. However, it still errs with the selection of the words *fidelity*, *cut* and *in progress*.

Model or version	Sentence
Original	Elke lidstaat behoudt de vrijheid zelf zijn behoeften te bepalen.
Ref1	Every member state remains free to determine its own needs.
Ref2	Every member state retains the freedom to determine its own needs.
Moses	each member state shall retain the freedom they themselves can identify their needs .
HR	each member state maintains the freedom itself to determine its needs
LABIT	each member state retains the freedom itself to determine its needs
LABITU	each member state maintains the partner freedom to determine its needs

Table 8.8: First sentence comparing different versions of an English translation for a Dutch sentence.

Table 8.10 displays an example where Moses seems to have produced an almost correct output, but where the placement of the auxiliary verb *been* completely changes the meaning of the sentence. This is a case where an evaluation metric that does not rely on syntax or semantics may produce a higher score than necessary. Note that the Moses output is otherwise still relatively close to the second reference translation. Unfortunately, the PaCo-MT output is very ungrammatical in this case.

In Table 8.11, the PaCo-MT output is again better than that of Moses. In this case, Moses is missing a main verb. The output of LABIT, although structurally unusual, is acceptable, apart from *a already* that should be *an already*. HR and LABITU are mostly perfect, although notice that the reference translators have decided to translate *de werklozen*, which is perfectly translatable to *the unemployed*, to the perhaps more stylistically suitable *the unemployed persons* and *the unemployed citizens*. It is also interesting to note that LABIT and LABITU put *in 2005* at the end of the sentence instead of fronting it such as is the case in the Dutch sentence.

Model or version	Sentence
Original	De beeldkwaliteit van de uiterst betrouwbare E-3, geknipt voor professionals en fotografen in spe, is gewoonweg subliem.
Ref1	Designed for professional and aspiring photographers alike, the E-3 boasts amazing image quality combined with reliability.
Ref2	The image quality of the highly reliable E-3, designed for professionals and budding photographers alike, is simply amazing.
Moses	the fidelity of the highly reliable e-3 , cut for professionals and photographers in progress , is quite superb .
HR	the quality of the extremely reliable e-3 be for professionals in progress and photographers is simply superb
LABIT	the perfect for professionals in progress and photographers quality of the extreme reliable e-3 is simply superb
LABITU	the quality of the extremely reliable e-3 perfect for photographers and professionals in progress is simply superb

Table 8.9: Second sentence comparing different versions of an English translation for a Dutch sentence.

Model or version	Sentence
Original	Ik zou het niet aandurven, omdat ik het nooit eerder heb gedaan.
Ref1	I would be afraid of that because I have never done it before.
Ref2	I wouldn't dare to take the risk, because I have never done it before.
Moses	i would not dare , because i have never before been done .
HR	i would hold not it because i never it did been have
LABIT	i would courage not i have never it did been because it
LABITU	i would courage do because i have never it did been it

Table 8.10: Third sentence comparing different versions of an English translation for a Dutch sentence.

Finally, we looked at the translations of a longer sentence, presented in table 8.12. Clearly, translation quality is very bad in all cases here. Moses seems to make slightly more sense, but in the second and third clauses, main verbs are missing again.

In our experience, large and structurally complex sentences may often be parsed incorrectly. In general, it also becomes more likely for non-terminal nodes to remain unaligned, as it becomes less likely for large tree pairs to remain structurally similar. All this results in less positive examples of training data for the system, among much noise.

8.4 Conclusion

Comparing the results with Moses shows that there is a long way to go for our syntax-based approach until we par with phrase-based SMT. One reason could be the fact that phrase-based SMT systems are known to be optimized to maximize BLEU. The effect of the n-gram modeling of Moses is quite clear,

Model or version	Sentence
Original	In 2005 waren 60,5% van de werklozen al meer dan één jaar werkloos.
Ref1	In 2005, 60.5% of the unemployed persons had been unemployed for more than one year.
Ref2	In 2005, 60,5% of unemployed citizens had been unemployed for over a year.
Moses	in 2005 , 60.5 % of the unemployed for more than one year unemployed .
HR	in 2005 60.5 % of the unemployed were already more than one year unemployed
LABIT	60.5 % of the unemployed . were at a already more than one year unemployed level in 2005
LABITU	60.5 % of the unemployed were already more than one year unemployed in 2005

Table 8.11: Fourth sentence comparing different versions of an English translation for a Dutch sentence.

where the occurrence of unusual word combinations is limited, whereas with PaCo-MT, odd combinations such as *did been* may occur. Nevertheless, our system has not yet reached its full maturity and there are several ways to improve the approach, as discussed in Vandeghinste et al. (2013).

It is difficult to say why some parallel treebanks lead to better scores and others not. All but Europarl consist of a variety of different domains, while Europarl is also by far the biggest collection. The test set also consists of sentences from a variety of different domains, with a certain amount of overlap. For example, some sentences also contain examples of legalese or formal constructions such as those found in Europarl or the DGT data that we worked with in chapter 7.

Our translation results do not currently stand up to the state-of-the-art in MT. Although some might argue that this supports the rejection of syntax-based MT as a serious alternative for phrase-based SMT, this approach is still in its infancy and there are still many avenues for future work to explore. In the meantime, the parallel corpora of Europarl, OPUS and DGT have been

Model or version	Sentence
Original	Hij rijdt dus niet tot aan het laatste station, maar vat meteen de terugreis aan om dan wel op tijd te zijn, aangezien er daar meer reizigers de trein willen nemen.
Ref1	As a result, it does not proceed to the final station but is diverted along its route and starts its return journey immediately, thus accommodating the larger number of passengers who are waiting for it at the return destination.
Ref2	It therefore does not run all the way up to the final station, but returns immediately to get there on time, because more passengers are waiting for it there.
Moses	it is not therefore up to the last station , but vessel the return journey to or to be on time , since more passengers the train .
HR	, but aanvat the return journey immediately to to be then be in time since more passengers want take there are take the train to will do until the last drive not are who
LABIT	to the last station up not are who but aanvat the return journey immediately to to since more passengers want to take there are on-board will be be in time
LABITU	far to the last drive do therefore he but aanvat the return immediately to to be in time then be since more passengers want to there the train take notice

Table 8.12: Fifth sentence comparing different versions of an English translation for a Dutch sentence.

significantly expanded, and other corpora are probably available as well. As a source of training data, this bodes well for the future development of MT systems in these languages. There may be merit to further research into the manipulation of tree structures for improved convergence, in order to counter the recall problem of tree-to-tree alignment approaches.

In the next chapter, we summarize the work that we have presented in this book, putting it into a bigger context. We discuss the meaning of our results, how they may impact future research, and the way forward.

Discussion and conclusion

9.1 Summary of the results

In this work, we have presented a flexible means for the alignment of phrase-structure trees in order to create parallel treebanks in the context of improving results for a hybrid syntax-based MT system. Although one of our main goals was to improve MT in the syntax-based paradigm, we have gone a step further in developing an easily accessible and flexible platform for a wide variety of needs.

We have shown that both stochastic and rule-based alignment systems can be successfully implemented. We have conducted quantitative and qualitative analyses of alignment output illuminating some problems, some of which we have solved and some of which are very much solvable in the near future. In the case of *Lingua-Align*, we have also conducted a statistical analysis in order to determine which features have the most impact on affecting performance.

During the course of our alignment experiments, the author has developed alignment gold standards consisting of 1406 parsed and aligned sentence pairs. The bitexts in these alignment sets were extracted from parallel corpora that are freely available online. Our gold standards are also freely available for further use in research and development purposes.

Alignment output sets were used as training data for our MT system, creating various models and evaluated against a state-of-the-art SMT system and against each other. Results show that although phrase-based SMT still outperform syntax-based MT, our high-recall systems produce promising results.

9.2 Research questions

In the introduction, we present three research questions that are addressed in this work. Here, we discuss whether or not, or to which degree, we have managed to answer them. For the reader’s convenience, we list them again.

Can we improve constituent alignment?

Precision is very important, but given the context of the application of aligned parallel treebanks to MT, some effort was also made to increase coverage and align as many constituents as possible. Therefore, we have given special attention to increasing recall.

We have shown that we can improve upon the results of a high-precision model produced by the statistical aligner *Lingua-Align*, presented in chapter 4. Although this does not prove that our rule-based approaches are better than *Lingua-Align*, our results are clearly promising. Since we have improved upon a decent baseline, we can say that at least in this respect we can answer the research question in the affirmative.

In chapter 2, we have listed some other systems, some of which obtain very good results as well, some better than reported in this work. It remains to be seen if those systems are in fact better, since we were not able to make any direct comparisons. In the future, we would like compare our TBL system with the following rival approaches more concretely:¹

- the Dublin Aligner (Zhechev, 2009)
- the system of Lavie et al. (2008) which uses the novel Prime Factorization and Alignments algorithm (PFA)
- alignment of packed forests as described by Liu et al. (2009)
- the system of Sun et al. (2010) which uses Bilingual Tree Kernels (BTKs)
- the implementation of the PFA algorithm by Araújo and Caseli (2011)

Although we have not won any shared tasks awards — an intriguing thought for the future — our results suggest that, given the variety of different domains used in our experiments, we have achieved near-state-of-the-art results, and

¹For obvious reasons, we only include systems which can produce parallel treebanks of phrase structures.

we may be optimistic about finding opportunities to improve. As far as we are aware, we are also the first to introduce rule-based learning to this paradigm, offering an exciting alternative to stochastic and manual rule-based systems.

Can we determine which factors influence constituent alignment?

If so, can we use this information to improve our alignment approach?

To answer this, we conducted qualitative and quantitative analyses to study the effect of various features on alignment quality. We showed how alignment improves as we apply what we learn from these studies.

First, our qualitative study presented in chapter 4 suggests that the quality of word alignment may have a large effect on the quality of constituent alignment, especially with recall. In order to be more flexible in the face of erroneous word alignments, we introduce relaxed versions of the well-formedness constraint in chapter 6. In the context of our manual rule-based experiments, when we allow for a less confident word alignment to violate the constraint but not for more confident word alignments, the effect is quite positive. This is also true of our bottom-up rule-based heuristic which is designed to increase alignment coverage of already aligned parallel treebanks. We implement this as a feature in our TBL system (chapter 7), where the user may choose any number of confident or less confident links to violate well-formedness.

In the bottom-up heuristic, we discover that applying a simple set of conditions is often sufficient for complementary alignment where the aim is to increase recall. A directional bottom-up approach which can be applied from source to target or from target to source provides extra flexibility, as well as extra precision or recall when combined.

Secondly, we conduct a statistical study in chapter 5 which shows that differences in sentence length are an important predictor of constituent alignment quality. The effects are more visible on the source-side trees (Dutch), suggesting that differences in tree structure may be influential. Although there is not much that can be done in terms of sentence length, tree manipulation may be a suitable course of action to improve alignment accuracy.

As for the diminishing the impact of bad parse trees, we admit that more can be done. Apart from improving the parsers, using an n -best list of parse trees instead of the first-best trees may help to counter the effect of bad parse trees.

Apart from improving the parsers, a very important issue is proper convergence of the syntactic structures of the two languages. In other words, the

parsers must be somewhat compatible. In many cases, incompatibility of the tree structures leads to a significant drop in recall for the extraction of phrase pairs. As an example, the number of phrases extracted from LABITU amounts to 36,467,596, whereas with Moses, a total of 59,558,940 phrases are extracted (Table 9.1). The fact that our Moses model also outperforms PaCo-MT by a fair amount suggests that this may be significant. The tree restructuring script of Vandeghinste, described in chapter 3, increases syntactic convergence but still assumes good parse trees. We have also applied it to the Dutch data aligned with French but without much success.

Stochastic inversion transduction grammars (Wu, 1997) as described in chapter 2, attempt to solve this problem. We already mentioned that this may fail in cases where segments do not correspond precisely. In addition, as reported in chapter 2, Wellington et al. (2006) report experiments showing that bitexts exhibit translational patterns that are more complex than reported in the literature. On the basis of this, they suggest that STIDs do not have the ability to generate some of these equivalence relations.

Although their criticism extends to tree-to-tree alignment, we argue in this work for the potential merit in using specialized monolingual parsers. In PaCo-MT, we utilize not only the structure, words and POS tags but also labeled dependencies, lemmas and other useful metadata and discriminating features for training the model. We remain positive about the potential merits of retaining as much knowledge as possible obtained from these parsers and prefer rather to work on the pipeline problem, improving convergence and, of course, using more training data.

The problem of complex translational patterns may be addressed by implementing packed forests. An implementation leading to increased BLEU scores is described by Liu et al. (2009), although they do not explicitly produce parallel treebanks. Using not only the best-first parse trees but rather an n -best list of candidates would be computationally more expensive, but it may also increase the chances of capturing a higher proportion of translational patterns in the text. This may have the additional advantage of further increasing recall.

We believe that the impact of bad sentence alignment, albeit always possible to improve, is minimal, since sentence alignment is generally very accurate. We have therefore not paid attention to this.

Taking all this into account, we can safely conclude that we have learned which factors influence constituent alignment, although perhaps not exhaustively. For example, a statistical study on the output of our TBL system in the same vein as the one presented in chapter 5 may also prove informative.

We may also affirmatively answer the question of whether we can use this information to improve alignment.

How does the quality of tree alignment influence the performance of syntax-based machine translation?

It is generally assumed that better tree alignment leads to better MT. However, in constructing our various alignment sets, we have discovered that using parallel treebanks produced by models focusing on high recall generally leads to better MT results than those produced by high-precision alignment models, at least for the cases that we have inspected. This has prompted us to research ways in which to increase alignment recall in general, as discussed before. Table 9.1 presents the total number of phrase pairs extracted from our produced alignment sets for Dutch to English. The names of the sets are the same as those used in the previous chapter. We also include the number of phrase pairs extracted by the best Moses model for comparison. Note that LABI was not processed by PaCo-MT, although we include it here for comparative purposes.

It is clear from the table that low-recall alignments have worse scores, even though the alignments themselves are accurate. However, the model resulting in the most alignments (LABITU) fares the best in only one of the metrics, TER. HR, which is not the highest recall model by a fair amount, is better in both BLEU and NIST, albeit not by much.

Our statistical significance tests suggest that the top scoring models do not differ significantly from each other, although they outperform the high-precision model. Given what we have, correlation between high-recall alignments and better MT results does not seem to be completely linear, although we need more data to make any definite statements. Logically, precision remains important, and we can assume that it varies between different alignment sets.

LABITU, which is the model which produces the most alignments, still produces relatively good scores on all three metrics. We can conclude that there seems to be a general tendency for high-recall alignments to produce better MT results.

This also means that there is no indication yet of a performance ceiling for PaCo-MT. First of all, adding more data generally leads to better scores. At the time of writing, much more parallel data has been added to the Europarl corpus, OPUS as well as the DGT corpora of the JRC Acquis. We daresay that it is almost certain that incorporating a few million more sentence pairs will significantly increase performance.

Data set	Phrase pairs	Alignments per s.p.	Best BLEU/NIST/TER
LA	7,724,500	2.80	0.2476/6.93/63.71
HR	14,868,119	5.39	0.2933/7.71/59.98
LABI	12,549,241	4.56	N/A
LABIT	16,784,348	6.09	0.2834/7.66/59.94
LABITU	18,233,798	6.61	0.2844/7.62/ 59.12
Moses	29,779,470	10.80*	0.4174/8.79/42.92

Table 9.1: Counts of phrase pairs extracted from all alignment sets for Dutch to English, with average constituent alignments per sentence pair and best MT results. For each set, the same bitexts were used, totaling 2,757,512 sentence pairs. s.p. = “sentence pair”; LA = Lingua-Align; HR = High Recall (source-to-target bottom-up heuristic applied to LA); LABI = LA+Bottom-up+Intersection; LABIT = LABI+Transformation-based learning error correction; LABITU = LABIT + bottom-up+Union. *Since for Moses we have used a phrase-based model, there was no tree alignment process. The number to the left is simply the number of phrase pairs extracted. We include the number-per-sentence-pair number only to give an idea of the increased coverage of the Moses model. Note that “phrase” in the context of Moses is simply a string of an arbitrary length and does not need to be linguistically motivated.

The positive trend for high-recall alignments means that we are probably able to improve quality by implementing additional means of aligning even more nodes. The most obvious next step is changing the tree structure for improved convergence, as in the previously discussed work by Burkett and Klein (2012), but more research is also needed in the application of different word alignment approaches to maximize constituent alignment recall.

On the basis of this, we may conclude that our work is probably suitable for use in syntax-based MT, but that more research is needed to determine whether or not this would ultimately lead to syntax-based MT being considered a serious alternative to phrase-based SMT.

9.3 Contribution

As a result of our research, we have developed a flexible alignment framework allowing the user the freedom to apply a variety of different alignment ap-

proaches, features and parameters for experiments. Our framework has the following attributes:

- the ability to choose a high-precision or a high-recall alignment approach, or something in between
- the ability to choose different alignment strategies and apply various constraints
- the ability to choose from a wide variety and combination of features and parameters
- the ability to experiment with different learning approaches as well as heuristics
- the ability to combine different alignment approaches
- a common platform facilitating the interaction of all alignment software

We would also like to address each of these points in more detail.

- For both Lingua-Align and our transformation-based learning (TBL) system, it is possible to train high-precision or high-recall models, or something in between. For Lingua-Align, it is a question of feature engineering and experimenting with different parameters. For our TBL system, we have generally found that using more features leads to higher precision scores, but also increases the risk of overtraining. The use of precise features that have a high chance to correlate positively with accuracy, such as removing a link when a number is only present on one side, generally leads to higher precision as well.
- Using the Lingua-Align toolkit, one can choose a variety of different alignment strategies, as described in chapter 4 and the software documentation. With our TBL system, we do not yet have this variety, but since rules are applied globally in one go, we do not have to worry about issues such as alignment directionality. Both systems offer the opportunity to impose the well-formedness constraint, which can be shortly defined as the condition where descendants of a candidate node pair are only linked to descendants of the node on the other side, and ancestors are only linked to ancestors. Lingua-Align offers the use of other constraints such as keeping to 1:1 alignments, only linking the same type of node, and so on.

- Both systems offer a variety and combination of different features. In the case of Lingua-Align, the ability to use contextual and history features and combining them using different mathematical operators creates an almost endless supply of variety. In the case of our TBL system, the number of features is more limited, but they can employ values of mostly infinite ranges. For both software applications, anyone with some skill in Perl would be able to extend the number of features that can be used in training and aligning new data.
- One may experiment using different learning approaches. Lingua-Align implements a maximum entropy classifier. Our rule-based system implements transformation-based learning. One may choose one system over the other for different experiments. For example, Lingua-Align is currently faster and can employ more features. On the other hand, output from our TBL system is relatively easy to inspect and to correct because the exact rules applied are visible for each iteration. The heuristics that we have used in our experiments in chapter 6 can be employed using stand-alone Perl scripts with their own set of easily understandable parameters.
- For our experiments using TBL described in chapter 7, we have combined various different alignment approaches in the form of initial state annotators and even as post-processing steps for the system. Eventually, combinations of alignments from Lingua-Align, TBL and our bottom-up rule-based algorithm were created. The simpler manual rules as described in the former part of chapter 6 can be just as easily combined with the other alignment approaches.
- All our tree alignment software is implemented in Perl. Makefiles and bash scripts are available for running a number of scripts on a more global scale, for example to train and test the TBL system in one go, with the potential for relatively effortless extension. For the most part, there is no need to manually install additional modules.

Additionally, we have also developed a set of 1406 sentence pairs for four different language pairs. They are freely available for further use in research and development.

9.4 Scientific significance

Whether or not tree-to-tree alignment will turn out to be important for machine translation in the long run, both manually and automatically created parallel treebanks remain an important resource for the study of language pairs. The evaluation and analysis of automatically created parallel treebanks are also useful endeavours in the process of improving the tools that created them.

Moreover, parallel treebanks can actually improve phrase-based SMT systems. Tinsley (2010) uses an implementation of the Dublin Aligner to create a series of parallel treebanks. The author finds that syntactically motivated phrase pairs and the information encoded in this resource can be exploited to improve PBSMT models (Tinsley, 2010, p. 140).

The technologies developed for this work can be applied to similar tasks, such as aligning dependency structures, for example. Our successful implementation of TBL to tree alignment also suggests that there is merit to the further investigation of applying rule-based learners to tree alignment.

9.5 Future work and conclusion

Much can still be done to improve the alignment tools that we have described in this work. Perhaps the most obvious next step is the addition of new features. Some features in Lingua-Align can also be implemented in the TBL system and vice versa.

For both systems, the biggest bottleneck in terms of the speed of execution is feature extraction. This is an important avenue for further research.

We can run more experiments to test the effect of different combinations of features, parameters, alignment strategies, data set sizes, textual domains, parsers and word aligners. We believe that statistical analyses should go hand-in-hand with investigating and understanding the linguistic motivation behind our conclusions. Both will help us to improve our systems, while the latter will also contribute to furthering our knowledge of the language pairs involved, and may ultimately also help to improve the other tools used for our research.

As mentioned in chapter 7, it may be profitable to employ TBL to correct both word and constituent alignments in a single step. This might be a suitable addition to the literature surrounding the improvement of word alignment using syntactic constraints (see for example, Ma et al. (2008)).

Another important step would be the extension to other language pairs. Computational processing of more distant language pairs remains a challenge

for many kinds of tasks. For tree alignment, this has been hinted at by the lower performance of aligning Dutch to French. However, we suspect that a lack of convergence between the two parsers plays a significant role in this case.

As a means to improve MT, we need to apply our parallel treebanks to other systems as well in order to obtain more robust results. If we manage to achieve consistently good results on a variety of systems, this will solidify the importance of tree-to-tree alignment for MT purposes.

We hope that the reader has found our work stimulating, and that it will prove to be a boost to both the research surrounding automatic parallel treebank creation and syntax-based machine translation.

Bibliography

- B. Aarts and SA Wallis. 2006. *The Diachronic Corpus of Present-Day Spoken English (DCPSE)*. (Version ICECUP 3.1). [Software]. Survey of English Usage, UCL: London.
- A. Abeillé, L. Clément and F. Toussenel. 2003. Building a treebank for French. In A. Abeillé (ed.): *Treebanks*. Kluwer, Dordrecht.
- L. Ahrenberg. 2007. LinES: An English-Swedish Parallel Treebank. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA'07)*, Tartu, Estonia. pp. 270–274.
- A.V. Aho and J.D. Ullman. 1969. Syntax directed translations and the pushdown assembler. In *Journal of Computer and System Sciences*, vol. 3, pp. 37–56.
- L. Ahrenberg, M. Andersson and M. Merkel. 1998. A Simple Hybrid Aligner for Generating Lexical Correspondences in Parallel Texts. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*. pp. 29–35.
- V. Ambati and A. Lavie. 2008. Improving Syntax Driven Translation Models by Re-structuring Divergent and Non-isomorphic Parse Tree Structures. In *Proceedings of the Student Research Workshop at the Eighth Conference of the Association for Machine Translation in the Americas (AMTA'08)*, Waikiki, HI.
- J.G. Araújo and H.M. Caseli. 2010. Alignment of Portuguese-English syntactic trees using part-of-speech filters. In *Workshop on Natural Language Processing and web-based technologies (IBERAMIA-2010)*, Bahia Blanca, Argentina.

- J.G. Araújo and H.M. Caseli. 2011. Combining Models for the Alignment of Parallel Syntactic Trees. In *Brazilian Symposium in Information and Human Language Technology (STIL)*, Cuiabá, Brazil. pp. 169–173.
- N.F. Ayan, B.J. Dorr and C. Monz. 2005. Alignment link projection using transformation-based learning. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Vancouver, British Columbia, Canada. pp. 185–192. Association for Computational Linguistics: Stroudsburg, PA, USA.
- D. Bamman, F. Mambrini and G. Crane. 2009. An Ownership Model of Annotation: The Ancient Greek Dependency Treebank. In *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories (TLT 2009)*, pp. 5–15.
- S. Bangalore and A.K. Joshi. 2010. *Supertagging*. MIT Press.
- W.J. Black and A. Vasilakopoulos. 2002. Language Independent Named Entity Classification by modified Transformation-Based Learning and by Decision Tree Induction. In *Proceedings of CONLL2002*. pp. 159–162.
- R. Bod. 1992. A Computational Model of Language Performance: Data-Oriented Parsing. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING)*, Nantes, France. pp. 855–856.
- C. Boitet and M. Tomokiyo. 1995. Ambiguities and ambiguity labelling: towards ambiguity data bases. In: R. Mitkov and N. Nicolov (eds.) *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*. Tsigov Chark, Bulgaria.
- C. Bosco, M. Sanguinetti and L. Lesmo. 2012. The Parallel-TUT: a multilingual and multiformat treebank. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).
- E. Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings, Third Conference on Applied Natural Language Processing, ACL*. Trento, Italy.
- E. Brill. 1993a. Automatic grammar induction and parsing free text: A transformation-based approach. In *Proceedings, 31st Meeting of the Association of Computational Linguistics*. Columbus, OH.
- E. Brill. 1993b. *A Corpus-Based Approach to Language Learning*. Doctoral dissertation. Department of Computer and Information Science, University of Pennsylvania.
- E. Brill. 1993c. Transformation-based error-driven parsing. In *Proceedings, Third International Workshop on Parsing Technologies*. Tilburg, The Netherlands.

- E. Brill. 1994. Some advances in rule-based part of speech tagging. In *Proceedings, Twelfth National Conference on Artificial Intelligence (AAAI-94)*. Seattle, WA.
- E. Brill. 1995. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics*. 21(4):543–565.
- E. Brill. 1996. Chapter: Learning to Parse with Transformations. In *Recent Advances in Parsing Technology*. Kluwer.
- E. Brill. and P. Resnik. 1994. A transformation-based approach to prepositional phrase attachment disambiguation. In *Proceedings, Fifteenth International Conference on Computational Linguistics (COLING-1994)*. Kyoto, Japan.
- P. Brown, F. Cocke, S. Della Pietra, V.J. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer and P. Roossin. 1990. A statistical approach to machine translation. In *Computational Linguistics*. Vol. 16, nr. 2, pp. 79–85.
- D. Burkett and D. Klein. 2012. Transforming Trees to Improve Syntactic Convergence. In *Proceedings of EMNLP*. Jeju Island, South Korea.
- L. Burnard. 2000. *User Reference Guide for the British National Corpus*. Technical Report, Oxford University Computing Services.
- M. Čmejrek, J. Cuřín, J. Havelka, J. Haijič and V. Kuboň. 2004 Prague Czech-English Dependency Treebank: Syntactically Annotated Resources for Machine Translation. In *Proceedings of LREC 2004*, Lisbon, Portugal.
- M. Carl, M. Melero, T. Badia, V. Vandeghinste, P. Dirix, I. Schuurman, S. Markantonatou, S. Sofianopoulos, M. Vassiliou and O. Yannoutsou. 2008. METIS-II: Low Resources Machine Translation : Background, Implementation, Results, and Potentials. In *Machine Translation*. Vol. 22, nr. 1, pp. 67–99.
- D. Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the ACL*. pp. 263–270.
- D. Chiang. 2006. *An introduction to synchronous grammars*. ACL Tutorial.
- D. Chiang. 2007. *Hierarchical phrase-based translation*. In *Computational Linguistics*, 33(2):201–228.
- M. Colhon. 2012. Language engineering for syntactic knowledge transfer. In *Computer Science and Information Systems*, 9(3):1231–1247.
- J. Cuřín, M. Čmejrek, J. Havelka, J. Haijič, V. Kuboň and Z. Žabokrtský. 2004. *Prague Czech-English Dependency Treebank Version 1.0*.
- L. Cyrus. 2006. Building a resource for studying translation shifts. In *In Proceedings of the 5th Conference of Language Resources and Evaluation (LREC '06)*, pp. 1240–1245. Genoa, Italy.

- H. Daumé III. Notes on CG and LM-BFGS optimization of logistic regression. Implementation available at <http://pub.hal3.name#daume04cg-bfgs>, implementation available at <http://hal3.name/megam/>, 2004.
- H. Daumé III. 2006. Practical Structured Learning Techniques for Natural Language Processing. Ph.D. Thesis, University of Southern California, Los Angeles, CA, August.
- M. de Marneffe, B. MacCartney and C. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the 5th edition of the International Conference on Language Resources and Evaluation (LREC)*.
- D. De Kok. 2008. Headline generation for Dutch newspaper articles through transformation-based learning. Master's Thesis. University of Groningen.
- T.G. Dietterich. 2002. Machine learning for sequential data: A review. In T. Caelli, editor, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 2396 of *Lecture Notes in Computer Science*, pp. 15–30. Springer, 2002.
- Y. Ding, D. Gildea and M. Palmer. 2003. An Algorithm for Word-Level Alignment of Parallel Dependency Trees. In *Proceedings of MT Summit IX*, pp. 95–101. New Orleans, LA, USA.
- S. Dipper, T. Brants, W. Lezius, O. Plaehn and G. Smith. 2001. The TIGER Treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, pp. 24–41.
- G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram cooccurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*. pp. 138–145. Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA.
- K. Dukes, E. Atwell and A.M. Sharaf. 2010. Syntactic Annotation Guidelines for the Quranic Arabic Dependency Treebank. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. pp. 19–21.
- J. Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the 41st Annual Meeting of the ACL*. pp. 205–208.
- C. Fellbaum (ed.) 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.
- R. Florian, J.C. Henderson and G. Ngai. 2000. Coaxing confidences from an old friend: probabilistic classifications from transformation rule lists. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing (EMNLP '00) and very large corpora: held in conjunction with*

- the 38th Annual Meeting of the Association for Computational Linguistics, Hong Kong. Volume 13, pp. 26–34. Association for Computational Linguistics: Stroudsburg, PA, USA.
- R. Florian and G. Ngai. 2001. Multidimensional transformation-based learning. In *Proceedings of the 2001 workshop on Computational Natural Language Learning (ConLL '01)*, Volume 7. pp. 1:1-1:8. Association for Computational Linguistics. Stroudsburg, PA, USA.
- H.J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of the 2002 conference on Empirical Methods in Natural Language Processing*, Philadelphia, US. pp. 304–311.
- A. Fraser and D. Marcu. 2007. Measuring Word Alignment Quality for Statistical Machine Translation. In *Computational Linguistics*. Vol. 33, nr. 3, pp. 293–303.
- P. Fung, G. Ngai, Y. Yang and B. Chen. 2004. A maximum-entropy chinese parser augmented by transformation-based learning. In *ACM Transactions on Asian Language Information Processing (TALIP)*. Vol. 3, nr. 2, pp. 159–168. ACM: New York, NY, USA.
- W.A. Gale and K.W. Church. 1993. A program for aligning sentences in bilingual corpora. In *Computational Linguistics*. Vol. 19(1), pp. 75–102. MIT Press: Cambridge, MA, USA.
- M. Galley, M. Hopkins, K. Knight and D. Marcu. 2004. What’s in a translation rule? In *Proceedings of the HLT Conference of the North American Chapter of the ACL (NAACL)*, Boston, USA. pp. 273–280.
- A. Göhring and M. Volk. 2011. The Text+Berg Corpus: An Alpine French-German Parallel Resource. In *Proceedings of Traitement Automatique des Langues Naturelles (TALN 2011)*. Montpellier, France.
- D. Gildea. 2003. Loosely Tree-Based Alignment for Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association of Computational Linguistics (ACL-03)*, pp. 80–87. Sapporo, Japan.
- Y. Graham. 2010. Sulis: An Open Source Transfer Decoder for Deep Syntactic Statistical Machine Translation. In *The Prague Bulletin of Mathematical Linguistics*, vol. 93, pp. 17–26.
- Y. Graham. and J. van Genabith. 2010a. Deep Syntax Language Models and Statistical Machine Translation. In *Proceedings of the 4th Workshop on Syntax and Structure in Statistical Translation (SSST-4)*, Beijing, China. pp. 118–126.
- Y. Graham. and J. van Genabith. 2010b. Factor Templates for Factored Machine Translation Models. In *Proceedings of the 7th International Workshop on Spoken Language Translation (IWSLT)*. Paris, France.

- S. Greenbaum and G. Nelson. 1996. The International Corpus of English (ICE) Project. In *World Englishes*, vol. 15(1), pp. 3–15. Blackwell Publishing Ltd.
- R. Grishman. 1994. Iterative Alignment of Syntactic Structures for a Bilingual Corpus. In *Proceedings of the Second Annual Workshop for Very Large Corpora*. Tokyo, Japan.
- D. Groves, M. Hearne and A. Way. 2004. Robust Sub-Sentential Alignment of Phrase-Structure Trees. In *Proceedings of the 20th International Conference on Computational Linguistics (CoLing 2004)*, pp. 1072–1078. Geneva, Switzerland.
- Y. Guo, J. van Genabith and H. Wang. 2008. Dependency-based N-gram Models for General Purpose Sentence Realisation. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING)*, pp. 297–304. Manchester, UK.
- S. Gustafson-Čapková, Y. Samuelsson and M. Volk. 2007. *SMULTRON (version 1.0) - The Stockholm MULtilingual parallel TReebank. An English-German-Swedish parallel Treebank with sub-sentential alignments*. Department of Linguistics, Stockholm University.
- J. Hajič. 1998. Building a Syntactically Annotated Corpus: The Prague Dependency Treebank. In *Issues of Valency and Meaning*. pp. 106–132, Karolinum, Prague.
- J. Hajič, E. Hajičová, J. Panevová, P. Sgall, O. Bojar, S. Cinková, E. Fučíková, M. Mikulová, P. Pajas, J. Popelka, J. Semecký, J. Šindlerová, J. Štěpánek, J. Toman, Z. Urešová and Z. Žabokrtský. 2012. Announcing Prague Czech-English Dependency Treebank 2.0. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).
- S. Hansen-Schirra, S. Neumann and M. Vela. 2006. Multi-dimensional Annotation and Alignment in an English–German Translation Corpus. In *Proceedings of the workshop on Multi-dimensional Markup in Natural Language Processing (NLPXML'06)*. pp. 35–42. Trento, Italy.
- B. Harris. 1988. Bi-Text, a new concept in translation theory. *Language Monthly*. 54:8–10.
- H. Hassan, K. Sima'an and A. Way. 2007. Supertagged Phrase-based Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, Czech Republic. pp. 288–295.
- M. Hearne and A. Way. 2003. Seeing the wood for the trees. Data-Oriented Translation. In *MT Summit IX*.

- M. Hearne and A. Way. 2006. Disambiguation strategies for data-oriented translation. In *MT Proceedings of the 11th Conference of the European Association for Machine Translation 2006 (EAMT 2006)*. pp. 59–68.
- M. Hearne, J. Tinsley, V. Zhechev and A. Way. 2007. Capturing Translational Divergences with a Statistical Tree-to-Tree Aligner. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI '07)*. pp. 85–94. Skövde, Sweden.
- U. Hermjakob. 2009. Improved word alignment with statistics and linguistic heuristics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP'09)*, Singapore. pp. 229–237. Association for Computational Linguistics: Stroudsburg, PA, USA.
- D. Higgins. 2004. A transformation-based approach to argument labeling. In *H.T. Ng and E. Riloff (editors): HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*. pp. 114–117. Association for Computational Linguistics: Boston, Massachusetts, USA.
- R. Hudson. 1984. *Word Grammar*. Basil Blackwell, Oxford and New York.
- K. Imamura. 2001. Hierarchical Phrase Alignment Harmonized with Parsing. In *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium*. pp. 377–384. Tokyo, Japan.
- D. Higgins. 2004. A transformation-based approach to argument labeling. In *H.T. Ng and E. Riloff (editors): HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*. pp. 114–117. Association for Computational Linguistics: Boston, Massachusetts, USA.
- H. Kaji, Y. Kida and Y. Morimoto. 1992. Learning Translation Templates from Bilingual Text. In *Christian Boitet (editor): Proceedings of the 14th International Conference on Computational Linguistics (CoLing'92)*. vol. 2, pp. 672–678. Nantes, France.
- E. Kavallieratou, E. Stamatatos, N. Fakotakis and G. Kokkinakis. 2000. Hand-written Character Segmentation Using Transformation-Based Learning. In *Proceedings of International Conference on Pattern Recognition (ICPR)*. pp. 2634–2637. IEEE Computer Society.
- M. Kay, J.M. Gawron and P. Norvig. (eds.) 1994. *Verbmobil. A Translation System for Face-to-Face Dialog*. CSLI.
- S. Ker and J.S. Chang. 1997. A class-based approach to word alignment. In *Computational Linguistics*. Vol. 23, pp. 313–343.
- D. Klein and C.D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pp. 423–430.

- D. Klein and C.D. Manning. 2003b. Fast Exact Inference with a Factored Model for Natural Language Parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pp. 3–10. Cambridge, MA: MIT Press.
- P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*.
- P. Koehn. 2010. *Statistical Machine Translation*. Cambridge.
- P. Koehn. 2011. What is a better translation? Reflections on six years of running evaluation campaigns. *Tralogy 2011*.
- P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin and E. Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of ACL-07, Prague, Czech Republic, 2007*.
- G. Kotzé. 2011. Finding Statistically Motivated Features Influencing Subtree Alignment Performance. In *Bolette Sandford Pedersen, Gunta Nešpore and Inguna Skadiņa (Eds.) Proceedings of the 18th Nordic Conference of Computational Linguistics*, pp. 332–335. Riga, Latvia.
- G. Kotzé. 2011. Improving syntactic tree alignment through rule-based error correction. In *Proceedings of ESSLLI 2011 Student Session*. pp. 122–127, Ljubljana, Slovenia.
- G. Kotzé. 2011. Rule-Induced Error Correction of Aligned Parallel Treebanks. In *Proceedings of Corpus Linguistics*, pp. 35–40. Saint Petersburg, Russia.
- G. Kotzé. 2012. Transformation-based tree-to-tree alignment. In *Computational Linguistics in the Netherlands Journal*. Vol. 2, pp. 71–96.
- G. Kotzé, V. Vandeghinste, S. Martens and J. Tiedemann. 2012. Large Aligned Treebanks for Syntax-based Machine Translation. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. pp. 467–473. European Language Resources Association (ELRA).
- A. Kroch and A. Taylor. 2000. *The Penn-Helsinki Parsed Corpus of Middle English (PPCME2)*. Department of Linguistics, University of Pennsylvania. CD-ROM, second edition, (<http://www.ling.upenn.edu/hist-corpora/>).
- A. Kroch, B. Santorini and L. Delfs. 2004. *The Penn-Helsinki Parsed Corpus of Early Modern English (PPCEME)*. Department of Linguistics, University of Pennsylvania. CD-ROM, first edition, (<http://www.ling.upenn.edu/hist-corpora/>).
- A. Kroch, B. Santorini and A. Dierani. 2010. *The Penn-Helsinki Parsed Corpus of Modern British English (PPCMBE)*. Department of Linguistics, University of

- Pennsylvania. CD-ROM, first edition, (<http://www.ling.upenn.edu/hist-corpora/>).
- H. Kučera and W. Francis. 1967. *Computational Analysis of Present-day English*. Providence: Brown University Press.
- Y. Lü, M. Zhou and S. Li. 2001. Automatic Translation Template Acquisition Based on Bilingual Structure Alignment. In *Computational Linguistics and Chinese Language Processing*. Vol. 6, nr. 1, pp. 83–108.
- J.T. Lønning, S. Oepen, D. Beerman, L. Hellan, J. Carroll, H. Dyvik, D. Flickinger, J. Johannessen, P. Meurer, T. Nordgård, V. Rosén and E. Velldal. 2004. LOGON. A Norwegian MT effort. In *Proceedings of the Workshop in Recent Advances in Scandinavian Machine Translation*. Uppsala, Sweden.
- P. Lambert and R.E. Banchs. 2012. BIA: a Discriminative Phrase Alignment Toolkit. In *Prague Bulletin of Mathematical Linguistics*. pp. 43–54.
- A. Lavie. Stat-XFER: A general search-based syntax-driven framework for machine translation. In *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics*. pp. 362–375. Haifa, Israel.
- A. Lavie, A. Parlikar and V. Ambati. 2008. Syntax-driven learning of sub-sentential translation equivalents and translation rules from parsed parallel corpora. In *Proc. SSST-2*, pp. 87–95.
- P.M. Lewis and R.E. Stearns. 1968. Syntax-directed transduction. In *Journal of the ACM*. Vol. 15, pp. 465–588.
- W. Lezius, H. Biesinger and C. Gerstenberger. 2002. *TIGER-XML Quick Reference Guide*. (Tech. Rep.) IMS, University of Stuttgart.
- Z. Li, C. Callison-Burch, C. Dyer, J. Ganitkevitch, S. Khudanpur, L. Schwartz, W. Thornton, J. Weese and O. Zaidan. 2009. Joshua: an open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, Athens, Greece. pp. 135–139. Association for Computational Linguistics: Stroudsburg, PA, USA.
- X. Li, S. Strassel, S. Grimes, S. Ismael, M. Maamouri, A. Bies and N. Xue. 2012. Parallel Aligned Treebanks at LDC: New Challenges Interfacing Existing Infrastructures. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).
- Y. Liu, Y. Lü and Q. Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*

- of the AFNLP, Suntec, Singapore. Volume 2, pp. 558–566. Association for Computational Linguistics, Stroudsburg, PA, USA.
- Y. Liu, Q. Liu and S. Lin. 2010. Discriminative word alignment by linear modeling. In *Computational Linguistics*, vol. 36(3). pp. 303–339. Cambridge, MA, USA.
- J. Lundborg, T. Marek, M. Mettler and M. Volk. 2007. Using the Stockholm TreeAligner. In *Proceedings of the 6th Workshop on Treebanks and Linguistic Theories*, pp. 73–78. Bergen, Norway.
- Y. Ma, S. Ozdowska, Y. Sun and A. Way. 2008. Improving word alignment using syntactic dependencies. In *Proceedings of the Second Workshop on Syntax and Structure in Statistical Translation*, Columbus Ohio. pp. 69–77. Association for Computational Linguistics: Stroudsburg, PA, USA.
- L. Macken. 2010. *Sub-sentential alignment of translational correspondences*. Ph.D. Thesis, University Press Antwerp.
- L. Mangu and E. Brill. 1997. Automatic Rule Acquisition for Spelling Correction. In *Proceedings of the 14th International Conference on Machine Learning*, pp. 734–741. Morgan Kaufmann.
- D. Marcu, W. Wang, A. Echihabi and K. Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Sydney, Australia.
- M.P. Marcus, B. Santorini and M.A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. In *Computational Linguistics*, vol. 19(2). pp. 313–330.
- M. Marcus, G. Kim, M.A. Marcinkiewicz, R. Macintyre, A. Bies, M. Ferguson, K. Katz and B. Schasberger. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *ARPA Human Language Technology Workshop*. pp. 114–119.
- Mark, D. 2008-. *The Corpus of Contemporary American English: 450 million words, 1990-present*. Available online at <http://corpus.byu.edu/coca/>.
- Y. Matsumoto, H. Ishimoto and T. Utsuro. 1993. Structural matching of parallel texts. In *31st Annual Meeting of the ACL*, pp. 23–30.
- J. May and K. Knight. 2006. Tiburon: A weighted tree Automata Toolkit. In *Proceedings of the 11th International Conference on Implementation and Application of Automata (CIAA)*. Taipei, Taiwan.
- B. Megyesi, B. Dahlqvist, É.Á. Csató and J. Nivre. 2010. The English-Swedish-Turkish Parallel Treebank. In *Proceedings of the Seventh International Conference*

- on *Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).
- I.D. Melamed. 2000. Models of translational equivalence among words. In *Computational Linguistics*. Vol. 26, nr. 2, pp. 221–249.
- I.D. Melamed. 2001. *Empirical Methods for Exploiting Parallel Texts*. MIT Press.
- A. Menezes and S.D. Richardson. 2001. A Best-first Alignment Algorithm for Automatic Extraction of Transfer Mappings from Bilingual Corpora. In *Proceedings of the ACL 2001 Workshop on Data-Driven Methods in Machine Translation*, pp. 39–46. Toulouse, France.
- A. Mengel and W. Lezius. 2000. An XML-based encoding format for syntactically annotated corpora. In *Proceedings of LREC-2000*. pp. 121–126. Athens, Greece.
- T. Meyer. 2011. Multilingual Annotation and Disambiguation of Discourse Connectives for Machine Translation. In *Proceedings of 12th SIGdial Meeting on Discourse and Dialog*. pp. 194–203. Portland, OR, USA.
- T. Meyer and A. Popescu-Belis. 2012. Using Sense-labeled Discourse Connectives for Statistical Machine Translation. In *Proceedings of the EACL-2012 Workshop on Hybrid Approaches to Machine Translation (HyTra)*. Avignon, France.
- T. Meyer, A. Popescu-Belis, N. Hajlaoui and A. Gesmundo. 2012. Machine Translation of Labeled Discourse Connectives. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*.
- A. Meyers, R. Yangarber and R. Grishman. 1996. Alignment of shared forests for bilingual corpora. In *Proceedings of COLING-96*. pp. 460–465.
- A. Meyers, R. Yangarber, R. Grishman, C. Macleod and A. Moreno-Sandoval. 1998. Deriving Transfer Rules from Dominance-Preserving Alignments. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (CoLing-ACL'98)*. Vol. 2, pp. 843–847. Montreal, QC, Canada.
- G.A. Miller. 1995. WordNet: A Lexical Database for English. In *Communications of the ACM*. Vol. 38(11), pp. 39–41.
- M. Nagao. 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*. pp. 173–180, New York, NY, USA. Elsevier North-Holland, Inc.

- T. Nakazawa and S. Kurohashi 2011. Bayesian Subtree Alignment Model based on Dependency Trees. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, Chiang Mai, Thailand. pp. 794–802. Asian Federation of Natural Language Processing.
- R. Nesson, S. Shieber and A. Rush. 2006. Induction of Probabilistic Synchronous Tree-Insertion Grammars for Machine Translation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (ANTA 2006)*. pp. 128–137.
- H. Ney. 2005. One Decade of Statistical Machine Translation 1996-2005. *Proceedings of MT Summit X*. i-12 - 1–16. Phuket, Thailand.
- G. Ngai and R. Florian. 2001. Transformation-Based Learning in the Fast Lane. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies (NAACL '01)*. pp. 1–8, Association for Computational Linguistics, Stroudsburg, PA, USA.
- F.J. Och and H. Ney. 2001. A systematic comparison of various statistical alignment models. In *Computational Linguistics*, 29(1):19–51.
- A. Okumura, K. Muraki and S. Akamine. 1991. Multi-lingual sentence generation from the PIVOT interlingua. In *Proceedings of MT Summit III*. pp. 67–71.
- K. Papineni, S. Roukos, T. Ward and W.J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual meeting of the Association for Computational Linguistics*. pp. 311–318.
- S. Petrov and D. Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of NAACL HLT 2007*, pp. 404–411.
- A. Popescu-Belis, T. Meyer, J. Liyanapathirana, B. Cartoni and S. Zufferey. 2012. Discourse-level Annotation over Europarl for Machine Translation: Connectives and Pronouns. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association (ELRA): Istanbul, Turkey.
- A. Poutsma. 2000. Data-oriented translation. In: *Proceedings of COLING*, pp. 635–641.
- A. Poutsma. 2003. Machine Translation with Tree-DOP. In R. Bod, R. Scha, K. Sima'an (eds.): *Data-Oriented Parsing*. Chap. 18, pp. 339–358. CSLI Publications.
- K. Probst, L. Levin, E. Peterson, A. Lavie and J. Carbonel. 2002. MT for Minority Languages Using Elicitation-Based Learning of Syntactic Transfer Rules. In *Machine Translation*. Vol. 17, nr. 4, pp. 245–270.

- R Core Team. 2012. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. <http://www.R-project.org>.
- L.A. Ramshaw and M.P. Marcus. 1995. Text Chunking using Transformation-Based Learning. In *CoRR*. pp. 82–94.
- R. Reppen, N. Ide and K. Suderman. 2005. *American National Corpus (ANC) Second Release*. Linguistic Data Consortium, Philadelphia.
- S. Riezler and J.T. Maxwell III. 2006. Grammatical Machine Translation. In *Proceedings of the HLT Conference of the North American Chapter of the ACL (NAACL)*, New York, USA. pp. 248–255.
- A. Rios, A. Göhring and M. Volk. 2009. A Quecua-Spanish Parallel Treebank. In *Proceedings of the 7th Workshop on Treebanks and Linguistic Theories*.
- M. Rosetta. 1994. Compositional Translation. *International Series in Engineering and Computer Science*. Springer.
- M. Saers and D. Wu. 2011. Principled Induction of Phrasal Bilexica. In: M. Forcada, H. Depraetere, V. Vandeghinste (eds.) *Proceedings of the 15th International Conference of the European Association for Machine Translation*. pp. 313–320. EAMT, Centre for Computational Linguistics. Leuven, Belgium.
- K. Samuel. 1998. Lazy Transformation-Based Learning. In Diane J. Cook (ed.): *Proceedings of the Eleventh International Florida Artificial Intelligence Research Society Conference*. pp. 235–239, AAAI Press.
- K. Samuel, R. Carberry and K. Vijay-shanker. 1998. Dialogue Act Tagging with Transformation-Based Learning. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*. pp. 1150–1156.
- Y. Samuelsson and M. Volk. 2006. Phrase Alignment in Parallel Treebanks. In J. Hajič and J. Nivre (eds.): *Proceedings of the 5th Workshop on Treebanks and Linguistic Theories (TLT '06)*. pp. 91–102. Prague, Czech Republic.
- Y. Samuelsson and M. Volk. 2007. Alignment Tools for Parallel Treebanks. In: *Proceedings of GLDV Frühjahrstagung 2007*.
- Y. Samuelsson and M. Volk. 2007. Automatic Phrase Alignment: Using Statistical N-Gram Alignment for Syntactic Phrase Alignment. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories*, pp. 139–150. NEALT Proceedings Series. Geneva, Switzerland.
- M. Sanguinetti and C. Bosco. 2011. Building the multilingual TUT parallel treebank. In *Proceedings of The Second Workshop on Annotation and Exploitation of Parallel Corpora*. pp. 19–28. Hissar, Bulgaria.

1990. *Mathematical and Computational Aspects of Lexicalized Grammars*. University of Pennsylvania.
1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK.
- B. Schrader. 2007. *Exploiting Linguistic and Statistical Knowledge in a Text Alignment System*. PhD thesis, Universität Osnabrück.
- M. Snover, B. Dorr, R. Schwartz, L. Micciulla and J. Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*.
- R. Steinberger, B. Pouliquen, A. Widiger, C. Ignat, T. Erjavec, D. Tufis and D. Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*, Genoa, Italy.
- E. Steiner. (ed.) 1991. Special Issues on EUROTRA, *Machine Translation*, vol. 6.
- A. Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*. Denver, US.
- J. Sun, M. Zhang and C.L. Tan. 2010. Exploring Syntactic Structural Features for Sub-Tree Alignment Using Bilingual Tree Kernels. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden. pp. 306–315. Association for Computational Linguistics.
- H. Telljohann, E.W. Hinrichs, S. Kübler, H. Zinsmeister and K. Beck. 2009. *Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z)*. University of Tübingen.
- J. Tiedemann. 2003. OPUS - an open source parallel corpus. In *Proceedings of the 13th Nordic Conference on Computational Linguistics*. University of Iceland, Reykjavik.
- J. Tiedemann. 2003. *Recycling Translations - Extraction of Lexical Data from Parallel Corpora and their Application in Natural Language Processing*. Doctoral Thesis, Studia Linguistica Upsaliensia. Uppsala, Sweden.
- J. Tiedemann. 2004. Word to Word Alignment Strategies. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*. pp. 212–218.
- J. Tiedemann. 2009. News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces. In *Proceedings of Recent Advances in Natural Language Processing (RANLP-2009)*. pp. 237–248.

- J. Tiedemann. 2010. Lingua-Align: An Experimental Toolbox for Automatic Tree-to-Tree Alignment. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*. Valletta, Malta.
- J. Tiedemann. 2011. Bitext Alignment, Synthesis Lectures on Human Language Technologies, Graeme Hirst (ed), Morgan & Claypool Publishers.
- J. Tiedemann and G. Kotzé. 2009a. A Discriminative Approach to Tree Alignment. In Iustina Ilisei and Viktor Pekar and Silvia Bernardini (eds.): *Proceedings of the Workshop on Natural Language Processing Methods and Corpora in Translation, Lexicography, and Language Learning (in connection with RANLP'09)*. pp. 33–39, Association for Computational Linguistics, Borovets, Bulgaria.
- J. Tiedemann and G. Kotzé. 2009b. Building a Large Machine-Aligned Parallel Treebank. In Passarotti, M. and Przepiórkowski, A. and Raynaud, S. and Van Eynde, F. (eds.): *Proceedings of the 8th International Workshop on Treebanks and Linguistic Theories (TLT'08)*. pp. 197–208, EDUCatt, Milan, Italy.
- J. Tinsley. 2010. *Resourcing Machine Translation with Parallel Treebanks*. Ph.D. Thesis. Dublin City University, Ireland.
- J. Tinsley, V. Zhechev, M. Hearne and A. Way. 2007. Robust Language Pair-Independent Sub-Tree Alignment. In *Proceedings of Machine Translation Summit XI*. pp. 467–474. Copenhagen, Denmark.
- H. Uchida. 1986. Fujitsu machine translation system: ATLAS. In *Future Generations Computer Systems*. Vol. 2, pp. 95–100.
- K. Uchimoto, Y. Zhang, K. Sudo, M. Murata, S. Sekine and H. Isahara. 2004. Multilingual Aligned Parallel Treebank Corpus Reflecting Contextual Information and its Applications. In *Proceedings of the Workshop on Multilingual Linguistic Resources (MLR'04)*, Geneva, Switzerland. pp. 63–70.
- H. Uibo, K. Liin and M. Volk. 2005. Phrase alignment of Estonian-German parallel treebanks. Paper presented at *JRC EU-Enlargement Workshop 'Exploiting Parallel Corpora in up to 20 Languages'*, Arona, Italy.
- S. Ullah, M.A. Khan and K.S. Kwak. 2009. A discourse based approach in text-based machine translation. In *CoRR*, vol. abs/0911.1516.
- A. Van den Bosch. 1997. Learning to pronounce written words: A study in inductive language learning. Ph.D. thesis, Maastricht University.
- G.J. van Noord. 2006. At Last Parsing Is Now Operational. In P. Mertens, C. Fairon, A. Dister and P. Watrin (editors): *TALN'06. Verbum Ex Machina. Actes de la 13e conference sur le traitement automatique des langues naturelles*. pp. 20–42.

- G.J. van Noord, I. Schuurman and V. Vandeghinste. 2006 Syntactic Annotation of Large Corpora in STEVIN. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*. Genoa, Italy.
- V. Vandeghinste. 2007. Removing the distinction between a translation memory, a bilingual dictionary and a parallel corpus. In *Proceedings of Translation and the Computer*. ASLIB, London, UK.
- V. Vandeghinste. 2008. *A Hybrid Modular Machine Translation System. LoRe-MT: Low Resources Machine Translation*. Ph.D. Thesis. Leuven, Belgium.
- V. Vandeghinste. 2009. Tree-based Target Language Modeling. In *Proceedings of the 13th International Conference of the European Association for Machine Translation (EAMT-2009)*. Barcelona, Spain.
- V. Vandeghinste and S. Martens. 2009. Top-down Transfer in Example-based MT. In *Proceedings of the 3rd Workshop on Example-based Machine Translation*. pp. 69–76. Dublin, Ireland.
- V. Vandeghinste and S. Martens. 2010. Bottom-up Transfer in Example-based Machine Translation. *Proceedings of the 14th International Conference of the European Association for Machine Translation (EAMT-2010)*. Saint-Raphaël, France.
- V. Vandeghinste, S. Martens, G. Kotzé, J. Tiedemann, J. Van den Bogaert, K. De Smet, F. Van Eynde and G. Van Noord. 2013. Parse and Corpus-based Machine Translation. *Essential Speech and Language Technology for Dutch: resources, tools and applications*, Springer.
- D. Varga, L. Németh, P. Halácsy, A. Kornai, V. Trón and V. Nagy. 2005. Parallel corpora for medium density languages. In *Proceedings of Recent Advances in Natural Language Processing (RANLP-2005)*. pp. 590–596.
- E. Velldal and S. Oepen. 2006. Statistical Ranking in Tactical Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Sydney, Australia.
- M. Volk, A. Göhring, T. Marek and Y. Samuelsson. 2010. *SMULTRON (version 3.0) - The Stockholm MULTilingual parallel TReebank. An English-French-German-Spanish-Swedish parallel treebank with sub-sentential alignments*. Institute of Computational Linguistics, University of Zurich.
- W. Wang, J. Huang, M. Zhou and C. Huang. 2002. Structure Alignment Using Bilingual Chunking. In *Proceedings of the 19th Conference on Computational Linguistics*. pp. 1–7. Taipei, Taiwan.
- W. Wang, J. May, K. Knight and D. Marcu. 2010. Re-structuring, Re-labeling, and Re-aligning for Syntax-Based Machine Translation. *Computational Linguistics*, vol. 36(2), pp. 247–277.

- J. Weese, J. Ganitkevitch, C. Callison-Burch, M. Post and A. Lopez. 2011. Joshua 3.0: syntax-based machine translation with the Thrax grammar extractor. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, Edinburgh, Scotland. pp. 478–484. Association for Computational Linguistics: Stroudsburg, PA, USA.
- B. Wellington, S. Waxmonsky and I.D. Melamed. 2006. Empirical lower bounds on the complexity of translational equivalence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Sydney, Australia. pp. 977–984. Association for Computational Linguistics: Stroudsburg, PA, USA.
- K. Williams, C. Dozier and A. McCulloh. 2004. Learning Transformation Rules for Semantic Role Labeling. In *H.T. Ng and E. Riloff (editors): Proceedings of CoNLL-2004*. pp. 134–137. Association for Computational Linguistics: Boston, MA, USA.
- T. Witkam. 1988. DLT - an industrial R&D project for multilingual machine translation. In *Proceedings of COLING*. pp. 756–759.
- D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. In *Computational Linguistics*. Vol. 23, pp. 377–403.
- D. Wu. 2000. Bracketing and aligning words and constituents in parallel text using stochastic inversion transduction grammars. In *Parallel Text Processing: Alignment and Use of Translation Corpora*. Kluwer.
- D. Wu. 2010. *CRC Handbook of Natural Language Processing*, chapter Alignment, pp. 367–408. CRC Press, second edition, 2010.
- K. Yamada and K. Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the ACL*. pp. 523–530.
- Y. Zhang and S. Vogel. 2004. Measuring Confidence Intervals for the Machine Translation Evaluation Metrics. In *Proceedings of The 10th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2004)*, Baltimore, MD, USA. October 4-6, 2004.
- V. Zhechev. 2009. *Automatic generation of parallel treebanks: an efficient unsupervised system*. PhD Thesis, Dublin City University.
- V. Zhechev and A. Way. 2008. Automatic Generation of Parallel Treebanks. In *Proceedings of the 22nd International Conference on Computational Linguistics (CoLing)*. pp. 1105–1112.
- A. Zollman and A. Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In *Proceedings of the Workshop on Statistical Machine Translation*, New York City, USA. pp. 138–141.

Samenvatting

In recente jaren vindt er een verschuiving in het domein van data-gedrevene machinevertaling (MT) plaats: Er ontstond een grotere nadruk op de integratie van meer linguïstisch gemotiveerde data voor de bouw van systemen die vloeiende vertalingen kunnen produceren. Een zeer nuttige bron is het *parallelle treebank*. Dat kunnen wij definiëren als een *parallel corpus* dat op verschillende niveaus gealigneerd is. Zo een bron is echter alleen op een zeer grote schaal bruikbaar, en het is dus onpraktisch om handmatig te produceren. Een door computers leesbaar parallel treebank om MT systemen op te trainen moet dus automatisch worden geproduceerd, terwijl men ook daarbij rekening moet houden om aan zekere nauwkeurigheidstandaarden trouw te blijven.

De automatische bouw van parallelle treebanks is wel een tamelijk recent verschijnsel, maar er werd reeds veel onderzoek op gedaan. Onze bijdrage vindt in de context van de ontwikkeling van een hybridisch syntactisch gebaseerd systeem voor automatische vertaling plaats. In dit werk experimenteren wij op en passen wij bestaande en nieuwe methoden van *tree alignment* toe om een verscheidenheid van grote parallelle treebanks in verschillende taalparen te produceren, maar met de focus op het taalpaar Nederlands-Engels. Ons belangrijkste doel is om de kwaliteit van het aligneren van *constituenten* te verbeteren.

Ten eerste bekijken wij het gebruik van zogenaamde *maximum entropy models* om het probleem van alignment binair te classificeren - dat wil zeggen, om voor elk paar knopen in de bomen te besluiten of ze gealigneerd moeten zijn of niet.

Wij gebruiken maximum entropy models om systemen op een *discriminerende* manier te trainen door het gebruik van een log-lineair model: De gebruiker maakt een verzameling trainingsgegevens alsook een stel features aan, die de classifier kan gebruiken om te leren wanneer de paren knopen gealigneerd moeten worden. Gebaseerd op wat hij op de trainingsgegevens heeft geleerd, wordt het opgebouwde model op nieuwe data toegepast. Belangrijke features zijn word alignments, gelijkvormigheid van subbomen en de etiketten van knopen.

Wij onderzoeken welke factoren de kwaliteit van alignment beïnvloeden door middel van een meervoudige regressieanalyse en de berekening van correlaties. Ten tweede bekijken wij hoe wij regels kunnen gebruiken om de prestatie van de classifier te verbeteren. Vooral leggen wij klem erop om *meer* alignments te maken met het doel om *recall* te verhogen, omdat ons statistische model een hogere precisie maar een lagere recall heeft. Wij vinden dat heel simpele regels al positieve effecten op recall kunnen hebben. Wij experimenteren met verschillende handmatige regels alsook een heuristisch bottom-up algoritme. In het laatste geval gebruiken wij ook de relatieve gelijkvormigheid van de subbomen - waar de huidige paar kandidaatknopen als wortelknopen fungeren - als features.

Voor beide benaderingen vinden wij dat de kwaliteit van de word alignments zeer belangrijk is. De toepassing van de zogenaamde *well-formedness constraint* (WFC) heeft een positief effect op de precisie, maar leidt tot een verlaging in recall wanneer de word alignments afwijken. Wanneer wij regels gebruiken om de WFC te verslappen, doordat wij word alignments toelaten die niet door beiden subbomen gedeeld worden, gaan niet alleen de recall maar ook de F-score omhoog.

Uiteindelijk passen wij wat wij uit deze experimenten hebben geleerd toe op een regelgebaseerd systeem dat de zogenaamde methode van *transformation-based learning* gebruikt. Een stel mogelijke regels worden automatisch aangemaakt door een combinatie van features die men handmatig opstelt. Door middel van trainingsgegevens wordt er telkens één beste regel gekozen, die dan aan een lijst wordt toegevoegd, waarop de regel op een ongealigneerde versie van de trainingsgegevens wordt toegepast. Wanneer men geen regel meer vindt die tot een verbetering leidt, stopt de training. De lijst regels past men dan toe op nieuwe data.

Wij vinden dat een tamelijk kleine maar goed gekozen set features genoeg is om hoge scores te bereiken. Zoals met onze regelgebaseerde aanpak laten wij enkele schendingen van de WFC toe voor een verhoogde recall. Behalve de gelijkvormigheid van subbomen bekijken wij ook andere structurele features

zoals hoogte en unaire knopen. Niet alleen word alignments maar ook het type alignment - met meer of met minder vertrouwen - zijn belangrijke features. Ook gebruiken wij knoopenetiketten in beperkte mate.

Wij demonstreren dat een combinatie van statistische en regelgebaseerde methoden, die een aantal benaderingen tot alignment alsook verschillende structurele beperkingen toepast, niet alleen de uiteindelijke kwaliteit van alignment verbetert, maar ook actief het zogenaamde *pijplijnprobleem* aanpakt. Als annotaties in een reeks van modules streng sequentieel worden toegepast, zodat de ene module altijd voor de andere tewerk gaat, spreken wij van een pijplijn. Het pijplijnprobleem ontstaat als latere modules het werk van eerdere altijd overnemen en nooit mogen corrigeren. Het werk in dit proefchrift neemt een stap in de richting van een oplossing. Dat gebeurt doordat onze aanpak het aligneren van constituenten robuuster tegen fouten in het word alignment - hetgeen een direct en meetbaar effect op de kwaliteit van het aligneren van constituenten heeft - maakt, alsook relevante stijgingen in recall teweegbrengt.

Na het bespreken, demonstreren en het analyseren van de door ons gekozen systemen, passen wij onze output op het bovengenoemde syntactisch gebaseerde systeem toe en bekijken wij hoe zijn prestatie met een huidig state-of-the-art frase-gebaseerd statistisch systeem, dat op dezelfde datasets getrained is, vergelijkt. De resultaten laten zien dat het statistische systeem duidelijk beter is. Het syntactisch gebaseerde systeem levert echter soms vertalingen met een betere grammaticale kwaliteit. Ook laten wij zien dat vertaalmodellen die op hoge recall alignments - een combinatie van statistische (hoge precisie) en regelgebaseerde (hoge recall) alignments - getrained zijn, significant beter zijn dan de modellen die alleen op de door het statistische model geproduceerde hoge precisie alignments getrained zijn.

Ten slotte bespreken wij de resultaten, de wetenschappelijke betekenis van ons werk en wat nog in de toekomst ligt.