Exploration and Exploitation of Multilingual Data for Statistical Machine Translation

Simon Carter

Exploration and Exploitation of Multilingual Data for Statistical Machine Translation

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof.dr. D.C. van den Boom
ten overstaan van een door het college voor promoties ingestelde
commissie, in het openbaar te verdedigen in
de Agnietenkapel
op woensdag 5 december 2012, te 12:00 uur

door

Simon Christopher Carter

geboren te Londen, Engeland

Promotiecommissie

Promotor:
Prof. dr. M. de Rijke

Co-promotor:
Dr. C. Monz

Overige leden:
Prof. dr. A. van den Bosch
Dr. M. Dymetman
Prof. dr. P. Koehn
Prof. dr. C. de Laat

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

SIKS Dissertation Series No. 2012-46
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

Cover by Mark Assies
Printed by Off Page, Amsterdam

ISBN: 978-94-6182-197-3

To family, friends and colleagues,
for the love, laughs, and the counsel.

Many thanks go to

Christof,
for instructing me on how to be a scientist.

Christof, Johan, Maarten, and Wouter
for the comments and translations.

My co-authors,
for the collaboration.

Edgar, Manos, Marc and Wouter
for the drinks, laughs, and advice.

The ILPS group,
for the coffee, company, and cheerful conversations.

The ISLA futsal team,
for tolerating my two left feet.

The committee members,
for their valuable input and time.

The Labyrinth

Anthropos apteros for days
Walked whistling round and round the Maze,
Relying happily upon
His temperment for getting on.

The hundredth time he sighted, though,
A bush he left an hour ago,
He halted where four alleys crossed,
And recognized that he was lost.

"Where am I?" Metaphysics says
No question can be asked unless
It has an answer, so I can
Assume this maze has got a plan.

If theologians are correct,
A Plan implies an Architect:
A God-built maze would be, I'm sure,
The Universe in minature.

Are data from the world of Sense,
In that case, valid evidence?
What in the universe I know
Can give directions how to go?

All Mathematics would suggest
A steady straight line as the best,
But left and right alternately
Is consonant with History.

Aesthetics, though, believes all Art
Intends to gratify the heart:
Rejecting disciplines like these,
Must I, then, go which way I please?

Such reasoning is only true
If we accept the classic view,
Which we have no right to assert,
According to the Introvert.

His absolute pre-supposition
Is - Man creates his own condition:
This maze was not divinely built,
But is secreted by my guilt.

The centre that I cannot find
Is known to my unconscious Mind;
I have no reason to despair
Because I am already there.

My problem is how not to will;
They move most quickly who stand still;
I'm only lost until I see
I'm lost because I want to be.

If this should fail, perhaps I should,
As certain educators would,
Content myself with the conclusion;
In theory there is no solution.

All statements about what I feel,
Like I-am-lost, are quite unreal:
My knowledge ends where it began;
A hedge is taller than a man."

Anthropos apteros, perplexed
To know which turning to take next,
Looked up and wished he were a bird
To whom such doubts must seem absurd.

W. H. Auden

Contents

Chapter 1

Introduction

Machine Translation (MT) is the use of computers to translate text or speech from one natural language to another. An MT system is built to expect input in a specific language and translate it into a second language. The idea of using computers to translate text or speech from one natural language to another goes back over half a century. The potential for using computers in this way was first publicly mentioned by Warren Weaver in 1947. Urged to elaborate on his ideas by colleagues, Weaver wrote the memorandum titled, pointedly, "Translation," in 1949 [203]. He noted the relationship between the translation task, and that of code breaking, a field that continued to be an active area after WWII;

> ...it is very tempting to say that a book written in Chinese is simply a book written in English which was coded into the "Chinese code" [203].

Research into use of machines to translate foreign text, was, as much research, originally funded by governmental organisations for military and intelligence purposes, and that still remains true, to a large extent, to this day. Yet, in the last decade, with the prominence of the world-wide-web, there has been an explosion in the amount of content in a diverse amount of languages, ranging from formal, edited, news articles on one end of the spectrum to informal, unedited, user-generated text on the other. Clearly, there is still a desire by national organisations to identify and translate certain foreign texts of various domains and genres, but the desire for translation has also grown to multinational governmental and non-governmental institutions, such as the EU with its 23 official languages, and even large multi-national businesses.

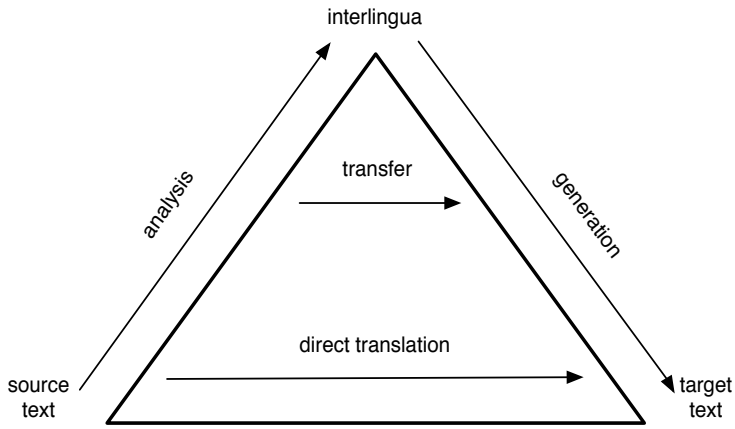


Figure 1.1: Translation Triangle.

1.1 History and Approaches

While there is a clear need for translation for a variety of goals, the difficulty of translating using computers was identified as early as the suggestion of the task itself. In his memorandum, Weaver discusses the different ways in which computers could be used to help tackle a translation task. Figure 1.1 shows a description of a translation triangle, a direct consequence of Weaver's memorandum, proposed in [197]. Looking at the triangle, there are a number of ways to get from the source sentence which we wish to translate on the left corner, to its translation in a target language at the right corner. All approaches since the founding of machine translation fall somewhere on this triangle.

The categorisation of an MT system can in part be defined by how high a level of abstraction the system goes up the triangle. The *direct*, or word by word approach, is the most basic, and suffers from the ambiguity problem, where a word can have multiple meanings; e.g., take the word "bank." The *transfer* based approach goes deeper, that is beyond the surface of the text, using shallow syntactic and semantic representations, such as Part-of-Speech (POS) tags, to translate between two languages. Finally, there is the *interlingual* approach, where a sentence is first translated into an internal logic. From this internal representation, the process is then reversed to arrive at a target translation.

Beyond a description of the representations used by an MT system, they can be

categorised, at a high level, as either being rule based or data driven.

Rule-Based Machine Translation (RBMT) These systems, typified by well-known brands such as Systran or Babelfish, employ human generated translation rules [40], written for a specific language pair and written for a specific translation direction, to translate foreign sentences into other natural languages.

Because of their nature, rule based systems have a number of shortfalls. Firstly, rules are human generated. Not only is this costly in terms of manpower, but also expensive in terms of time. Rules for all possible different contexts and situations must be foreseen and considered. Because of this, a limited number of systems have seen success in restricted domains, where the range of possible inputs are limited. Even then, such systems are limited by specificity to language pair. With over 6400 languages in use worldwide, to build translation systems for each possible language pair one would require 20,476,800 human generated systems, or double that if we take into account the uni-directional aspects of rule based translation software. Rule based systems are thus not feasible, and are unable to react to quickly changing socio-economic needs. Thus empirical, data-driven approaches are increasingly preferred. Empirical approaches can be split into two subcategories; the first based on translation memory and example-based approaches, and then finally, the statistical machine translation approaches.

Translation memory and EBMT In Example Based Machine Translation (EBMT), a large amount of bi/multilingual translation examples has been stored in a textual database, also called a translation memory [5], and input expressions are rendered in the target language by retrieving from the database that example which is most similar to the input [131]. The steps of an EBMT system can be said to include (i) matching fragments against existing examples, (ii) identifying the corresponding translation fragments, and (iii) recombining them to give the target text [95]. Translation memories only do the first step, and so it is said that EBMT systems build on-top of, or extend, translation-memories in that they actually present the user with a single translation, as opposed to just showing the matching translations in the memory [5].

Statistical Machine Translation The aim of Statistical Machine Translation (SMT) is to is to take a foreign sentence, f, and translate it into a source sentence, e, using probabilistic models generated using machine learning techniques. Using a corpus of foreign and target sentences for which we know to be translations of one another, called a parallel text, or bi-text,[1] SMT becomes a problem of constructing accurate

[1] The parallel-text used in an SMT system is similar to a translation memory [5], though it is used differently.

probability distributions, or models, that can be used to translate a collection of foreign sentences into a target language. Such methods exploit co-occurrence statistics to learn translations of words and phrases between different human languages. The more data there is for learning translations, the more source phrases we will be able to translate, and the translations themselves will improve with accuracy.

The idea behind this is not new; learning from parallel text was used to decode the, at the time, unknown ancient Egyptian language from the known Demotic and Ancient Greek on the Rosetta Stone in 1822 [146]. Yet, it was not until the 1990's and the publication of the seminal work [24, 26] that the approach, using machines, really took hold. SMT systems are formed of two core components. The first is a *translation-model*, or *phrase table*, which stores tuples of source phrases, the target phrases they translate to, and a score for each pair. The second is a *language model*, which assigns a score to potential translation candidates, without any knowledge of the source sentence being translated.

The benefits of such data-based approaches over earlier labour-intensive methods are that, given a parallel corpus, almost anyone can build an SMT system without prior linguistic knowledge of either the foreign or target language. Due to the popularity of SMT systems, we use and build upon these throughout this thesis. This is partly due to the availability of well documented, open-source software, but also because these are increasingly used by the largest investors, providers and users of machine translation technology. We leave to Chapter 2 a full description of the modern SMT system that we use and build upon.

Before moving on to the research themes, we note that the RBMT, EBMT and SMT approaches are not mutually exclusive. Hybrid systems do exist, including the pairing of RBMT and EBMT systems [157, 169], RBMT and SMT systems [6, 62, 177], and finally the pairing of EBMT and SMT systems [112, 201]. However, the combination of different systems constitutes a field in itself, and we focus in this thesis on improving widely used, state of the art SMT algorithms.

1.2 Research Themes

The success of statistical MT methods is in many ways dependent on the quantity and quality of parallel text available to build an SMT system. Given that, at the core of many SMT approaches, the exploitation of co-occurrence statistics is used for creating a translation model, the more data there is, the better the statistics become.

Yet, the acquisition of such data can be difficult, depending on the language pair in question. For some language pairs, resources can be mined from governmental and trans-governmental organisations such as the Canadian government, European

Government, or United Nations. Further, many transnational corporations and non-governmental agencies, along with established media corporations, produce multilingual documents that can be exploited for training purposes. Unfortunately, such information either does not always exist or is not available, and even if it is, the domain and genre of the language pair in question may differ greatly from the text of interest, causing mismatches between the training and test data that results in a poorly performing system.

Thus, the first part of this thesis focuses around the following research direction: how can we get (more) relevant data? Specifically, the first theme we address is:

RT 1. Exploration: The quantity and quality of training data have a serious impact on the final quality of an SMT system. We examine ways of correctly identifying the language of unedited, informal, idiomatic online content, and then examine ways of using this content for acquiring new and better translations.

In addition to the core problem of data acquisition, there is a complementary concern of how to build accurate models given resources that are readily available. That is, if we put the problem of acquiring more data to one side, how can we dig deeper into existing resources? For example, one important concern is how to deal with out-of-vocabulary (OOV) types, that is, types that are in the test data, but have not been seen in the training corpora. SMT systems deal with unseen tokens by either ignoring them, or including them as-is into the translation, both of which are sub-optimal. Another concern is the limited context used by the language model for scoring of the fluency of a hypothesis translation. But the use of longer contexts directly within an SMT system causes two main problems. Not only does it increase the size of the space in which we have to search for the best translation, but the statistics on which such high-context language models rely become poor due to sparsity (i.e., lack of enough data).

Thus, the second theme we address in the second part of this thesis is: how can SMT practitioners better exploit the data all ready available available to address these problems? The next research theme is then:

RT 2. Exploitation: We examine ways of better leveraging existing resources. This can be via exploiting more complex features that better explain the data (i.e., syntax), or new methods for estimating existing models.

1.3 Research Questions

Because quantity and quality of data is important to building SMT systems, and because this data is naturally lacking for either low-resource languages or domains of

interest outside of our training data, we turn to the task of mining social media platforms for new translations. The initial growth in the amount of textual content on the web [143] has been followed with a growth in the use of social-media, that is applications that build upon the infrastructure of the web to allow "the creation and exchange of user generated content" [91]. Microblogging platforms offer users ways to post short, real-time messages on events, activities and thoughts, and a large proportion of users post messages in their own native language. We need to use an SMT system to translate these, but unfortunately such systems are often based on out-of-domain, formal texts, so we need to acquire in-domain texts. Before we can do this, we first need to be able to accurately identify the language of a microblog post. Language identification has been studied in the past (see Section 4.1 for previous work in this field), showing successful results on structured and edited documents. However, the use of language in microblog posts is very different from the more formal texts previously studied. People, for example, use word abbreviations or change word spelling so their message can fit in the allotted space, giving rise to a rather idiomatic language that is difficult to match with statistics from external corpora.

Because existing solutions to language identification rely on some form of comparison between n-gram statistics of the text under question and a background corpus representative of a language, the shortage of and mismatch in n-gram statistics make identification hard. We thus propose a number of models that exploit additional information, going beyond the text itself, to increase the sample size from which we can make these n-gram comparisons. Further to proposing new models, we also conduct a thorough analysis of the different models and optimisation strategies proposed, and demonstrate the distribution of language use on Twitter. We ask:

RQ 1. Can we successfully identify the language of a microblog post, given the short, idiomatic and unedited nature exhibited?

- **a.** What is the performance of a strong language identification method for microblogs posts?
- **b.** Does domain-specific training of language models help improve identification accuracy?
- **c.** What is the effect on accuracy of using priors extracted from microblog characteristics?
- **d.** Can we successfully combine semi-supervised priors in a post-independent way?
- **e.** How can we determine confidence of individual priors, and can we use confidence to combine priors in a post-dependent way?

After we have looked at mapping languages to posts, we turn to the task of domain adaptation and examine methods for extracting in-domain parallel data from the language assigned tweets. We compare two popular approaches, one based on using retrieval methods from the Information Retrieval (IR) field for finding potential translations, and the second, simpler, based on directly using the machine translation of the foreign tweets. We ask:

RQ 2. Given a set of language identified microblog posts, how can we learn new phrase translations from this data?

- **a.** Do information retrieval based approaches outperform simpler methods based on self learning?
- **b.** Which ways of combining the in-domain data with the baseline out-of-domain data work best? How much data from the microblog post corpus can we exploit? How important is the common filtering step for the information retrieval method?

Having looked at improving the phrase table via domain adaptation techniques using comparable corpora in Part I, we turn our attention to language models (LMs) in Part II. Specifically, we now examine the potential syntax has as a global reranker for improving the quality of translation output, and specifically for addressing the long distance reordering mistakes induced due to the nature of the stack-based search algorithm. Given the computational requirements of full parsers, we are the first to conduct a thorough analysis of the utility of a range of syntax features. Given that different features contain a varying range of information, and have varying computational extraction costs to deal with, we take into account practical considerations as well as absolute improvements. Given this, we ask:

RQ 3. Can a generative syntactic parser discriminate between human produced and machine output language? Are they useful in an SMT setting for reranking output, where global information may prove beneficial to translation quality?

- **a.** Given that syntactic models are trained only on examples of good text, and are optimised towards parse retrieval as opposed to LM probability, to what extent can they be used for differentiating between human and machine text, and more importantly, to what extent can they differentiate between different machine produced texts?
- **b.** As input to our machine learning algorithms for reranking, we extract and compute many different feature types from differing syntactic models. What

features help most, and from which syntactic models are they best extracted? Does the expressiveness of a full lexicalised parser outweigh the computational disadvantages in comparison to the use of faster, less expressive syntactic models, such as POS tagging?

In the final technical chapter, we examine the potential for the application of exact, high-order Hidden Markov Models (HMMs) to segmenting the source side of a Finnish-English SMT system. For morphologically rich source languages, decomposing the original text into smaller units, based either on morphologically or statistically coherent units, can reduce sparsity, push down OOV rates, and improve performance. In this chapter, we present a novel method, allowing for the use of exact, high order latent language models, to the application of the source-side decomposition task. In particular, we ask:

RQ 4. Can high-order HMMs be used to recover a partition of the source side of a parallel corpora that reduces sparsity and leads to improved translation quality?

a. HMMs which have a large latent vocabulary, or use a high amount of context, are often intractable. Can we create high-order HMMs that are tractable given large latent vocabulary spaces, or use large amounts of context, and are still exact?

b. Does the inclusion of the additional context, contained in the higher order models, lead to improved translation quality when segmenting Finnish? To what extent do the application of the different models leads to a reduction in sparsity problems and model learning?

c. Can improvements be made that exploit numerous different segmentation of the same input? What methods can we use do this?

1.4 Main Contributions

The following summarises the main contributions of this dissertation, which adds both theoretical and practical insights to the body of existing knowledge in the field.

- **Language modelling for microblogging platforms** We empirically demonstrate the challenges of a language identification system for microblog posts, and develop a method for improving recognition accuracy.

- **Mining microblog platforms for phrase translation acquisition** We propose three alternative models for finding new translation candidates on a popular microblogging platforms. We demonstrate the utility of our approach on an end-to-end translation task, and empirically compare our approach to existing methods developed for the news domain.

- **Rerankers based on novel feature set** We develop a novel syntactic n-best list reranker based on rectifying simple reordering mistakes, leveraging shallow Part-of-Speech sequences built from context-less taggers.

- **Comparison of syntactic rerankers** We provide a comprehensive analysis and evaluation of differing syntactic models, demonstrating why our simple technique outperforms more complex models.

- **Exact inferences for large HMMs** We provide an algorithm detailing how to do exact inference from HMMs, where the application of dynamic programming techniques in an otherwise standard setting is not feasible due to high-order or large-latent vocabularies.

In addition, we make available various data sets and resources to facilitate future work and comparisons with the work in this thesis. These include five thousand language identified microblog posts for the purpose of language identification experiments, and a collection of 1500 microblog posts translated from English into Dutch.

1.5 Overview of the Thesis

Chapter 2 serves as an introduction to the field of SMT, detailing the core models used in translating natural language text into another language. Many of the contributions of this thesis are distributed across the technical Chapters 4 through to 7.

- **Chapter 2 - Background** In this chapter we give an overview of an SMT system; including detailing core models used, parameter learning and the decoding framework for mapping a foreign sentence into its translation.

- **Chapter 3 - Methodology** In this chapter we describe the common experimental set-up for running and evaluating a machine translation system. Specifically, we describe the automatic evaluation metrics we use for reporting translation results, the significance tests we run, the toolkits we use, and the test collections used.

- **Chapter 4 - Microblog Language Identification** In this chapter, we examine the problem of language identifying microblog posts from a popular multilingual microblogging platform. This chapter serves as a necessary prelude to the extraction of new translations. We resolve the problem of short, idiomatic and unedited text by exploiting priors that give additional information which our classifier can exploit to improve classification accuracy.

- **Chapter 5 - Novel Term Translation Extraction** In this chapter, we tackle the problem of mining a popular microblogging platform for new, additional translations. Such translations have value in themselves in aiding to reduce source-side OOV rates on informal text domains, including SMS, blogs, microblogs, and online discussion forums. We compare existing methods originally built for the news service domain, demonstrating the problems they encounter and the benefit of exploiting the redundancy and natural features found in microblogging platforms to finding new translations.

- **Chapter 6 - Syntactic Reranking** In this chapter, we turn our focus to the exploitation of syntactic features that go beyond surface features to improve translation quality. First we demonstrate that syntactic parsers, built and optimised for a different task, are able to differentiate probabilistically between human and machine produced text. We then proceed to conduct a thorough investigation of different feature types and their impact on final translation quality.

- **Chapter 7 - Source-side Morphological Analysis** In this chapter we move beyond the exploitation of global features in a reranking phase and examine the potential for large HMMs to reduce source language sparsity. First we present a novel method for the doing exact inference given high order spaces, and demonstrate the applicability of such methods on two example tasks. We then proceed to apply the algorithm to decomposing Finnish language texts. We present results on BLEU, demonstrating the impact that different learning strategies and HMM orders have on final translation quality.

Most of the chapters should be accessible to anyone with some background in natural language processing (NLP). Chapter 7 is an exception, in that it is more technical in nature than the other chapters and expects the reader to have some familiarity in statistical inference. Chapters 2 and 3 provide background material for the remainder of the thesis. Chapters 4 and 5 constitute Part I, devoted to exploration, and Chapters 6 and 7 constitute Part II, devoted to exploitation. Parts I and II can be read independently.

1.6 Origins

The thesis is based on the following publications that have arisen as part of the thesis work. Early versions of the work presented in Part I were published as

- "Semi-Supervised Priors for Microblog Language Identification" [34] (Chapter 4)
- "Twitter hashtags: Joint Translation and Clustering" [33] (Chapter 4)
- "Microblog language identification: overcoming the limitations of short, unedited and idiomatic text" [36] (Chapter 4)

Part II builds on work presented in

- "Parsing Statistical Machine Translation Output" [29] (Chapter 6)
- "Discriminative Syntactic Reranking for Statistical Machine Translation" [30] (Chapter 6)
- "Syntactic Discriminative Language Model Rerankers for Statistical Machine Translation" [31] (Chapter 6)
- "Exact Sampling and Decoding in High-Order Hidden Markov Models" [35] (Chapter 7)

Finally, other publication sources for this thesis include [32, 204].

Chapter 2

Background

This chapter serves as an introduction to SMT, giving an overview of an SMT system, as well as describing evaluation metrics, optimisation methods and significance testing. Related work specific to the various chapters will be introduced in the respective chapters.

We begin in Section 2.1 by giving an overview of an SMT system, and an in-depth description of the core models used in modern, state of the art systems in Section 2.2. We then finish the chapter with a description of the decoding algorithm used for translating a foreign sentence in Section 2.3, and optimising the weights for the different models described in Section 2.2 in Section 2.4.

2.1 Statistical Machine Translation - System Overview

The aim of SMT is to take a foreign sentence, f, and translate it into a target sentence, e, using statistical models generated using machine learning techniques. Using a corpus of foreign and target sentences that we know to be translations of one another, the problem becomes one of constructing accurate probability distributions, or models, that can be used to translate a collection of foreign sentences, unseen in the training data, into a target language.

Intuitively, given a foreign sentence f, the problem of statistical machine translation can be formulated as picking the target sentence e with the highest probability according to the model $p(e|f)$. Using Bayes' theorem, we can rewrite this to:

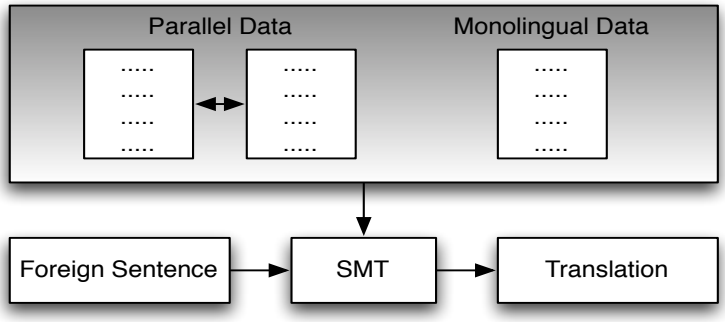


Figure 2.1: The basic architecture of noisy channel based translation systems.

$$p(e|f) = \frac{p(f|e)p(e)}{p(f)}. \tag{2.1}$$

Given that we seek to find the target sentence $\hat{e}$ for which the probability arrived at through Equation 2.1 is greatest, and considering that the denominator $p(f)$ is a constant independent of e, we can reformulate Equation 2.1 to:

$$\hat{e} = \arg\max_{e} p(f|e)p(e). \tag{2.2}$$

This approach is often referred to as the *noisy channel approach* [26]. Here, $p(f|e)$ is the translation model, or the likelihood of generating the foreign sentence given the target sentence, and $p(e)$ is the language model, which tells us the likelihood of a given target sentence. Together these form the core of all SMT systems, and thus Equation 2.2 is described as the Fundamental Equation of SMT [26].

Figure 2.1 demonstrates the translation process in a system using the noisy channel approach. The model $p(f|e)$ is trained on a corpus of translations between foreign and target sentences, and $p(e)$ is trained purely on monolingual target language data.

2.2 Standard models

Phrase-based SMT, as described in [138, 139], extends the noisy channel approach by using weighted log linear combinations of a set of k feature functions, f, to score possible translations:

$$\hat{e} = \arg\max_{e,a} \sum_{i=1}^{k} \lambda_i f_i(f, a, e), \tag{2.3}$$

where a is a phrasal alignment, and $p(f|e)$ and $p(e)$ become a subset of the k different feature functions, each with their own weights λ_i. A typical system (such as Moses [107]), would use, at the minimum, the following features:

- Translation Model
- Language Model
- Reordering Model
- Word and Phrase penalties.

We now discuss these models/features in turn.

2.2.1 Translation Models

The translation models are typically made up of two distinct models used in both directions. We now discuss these in turn:

Phrase Translation Probabilities $p_\tau(f|e)$ and $p_\tau(e|f)$. The probability distributions of a phrase given another in both directions. These are typically learnt from raw frequency counts, such that $p_\tau(f|e) = \frac{c(f,e)}{c(e)}$ and $p_\tau(e|f) = \frac{c(e,f)}{c(f)}$. Here, $c(\cdot)$ is the count of the n-gram or n-gram pair in the training data.

Lexical Translation Probabilities $p_l(f|e,a)$ and $p_l(e|f,a)$. The intuition behind using the lexical translation distributions along with the previous phrase translation probabilities is that, because they are calculated at the word level, they are based on more reliable statistics. This is particularly true where the phrases under consideration get long in length; in such situations the normalisation counts are usually small, leading to over-optimistic probabilities. Here the lexical weights can give a more realistic estimate of one phrase being a translation of another. It is calculated as:

$$p_l(e|f,a) = \prod_{i=1}^{length(e)} \frac{1}{|\{j|(i,j) \in a\}|} \sum_{\forall(i,j)\in a} w(e_i|f_i),$$

where $w(e_i|f_i)$ is the word translation probability, and a is the set of word alignments. phrase translation and lexical translation probabilities are often stored in a phrase table, a file containing phrase-level translations along with the corresponding probabilities.

The question now is how the phrase and lexical translation probabilities are learnt from a sentence aligned parallel corpus. This is governed by two steps; the first is word alignment, the second phrase extraction. We proceed to explain these steps.

Word alignment. Given a corpus of parallel sentences, also called a bi-text, the first goal is to find the hidden, latent alignments (first proposed by Brown et al. [24]) between words on different sides of the bi-text, specifying which words translate to which foreign words given in each parallel sentence. As the bi-text does not include any labels specifying which words are aligned, this task is an unsupervised one. Once the bi-text has been word aligned, phrasal translations can be extracted.

The most common methods for word alignment are based on the IBM models 1–5 [26]. Multiple models are used in a bootstrapping fashion, where the output of one model is used as input to the next, because the higher-order models are more complex. In this way the simpler models are used to initiate the parameters of the more complex models. The entire search is governed by the Expectation Maximisation (EM) algorithm [56]. Because the word alignment algorithms define a distribution over all possible alignments, when the algorithms are finished, the viterbi-best alignments are used for each sentence pair.

Further, because the IBM models only allow for single-to-many alignments, where one word can be aligned to many, but not the other way around, these models are run in both directions and combined to get many-to-many alignments. There are a number of ways of merging the alignments from both directions, including taking the intersection, taking the union, or starting with the intersection, and expanding symmetrised alignment points if they appear in the union (*grow-diag*), and finally adding unaligned words that also appear in the union (*grow-diag-final*).

Phrase extraction. The word translations probabilities $w(e_i|f_i)$ and $w(f_i|e_i)$ used to estimate $p_l(f|e), p_l(e|f)$ can be estimated directly from the symmetric word alignments, however we still need to extract phrases before we can estimate the phrase translation probabilities:

- all words in the extracted phrase must align to each other,
- phrase pairs can include unaligned words between aligned words, or at the boundaries of the source and target phrase, and
- a phrase pair must include at least one alignment point.

Once phrase pairs have been extracted, phrase translation probabilities can then be computed.

2.2.2 Language Models

Language models are an essential component in SMT. Their most basic function is to measure the likelihood of a sequence of words being uttered by a fluent English speaker. Formally, LMs are a probability distribution $p(s)$ over a set of strings S that attempt to reflect how common s is [42]. Conceptually they are very easy to understand; $p(\text{the black car}) > p(\text{the car black})$. One can see that when making judgments about the fluency of an output translation, the language model is aiding the translation system in making difficult word order decisions.

Traditionally, the most widely used language models are n-gram markov models. The probability of a sentence is decomposed into the product of the probabilities of each word given its history, or context, of $n-1$ words:

$$p(s) \approx \prod_{i=1}^{l+1} p(w_i | w_{i-n+1}^{i-1}). \tag{2.4}$$

We say that w_0 is BOS, a beginning of sentence tag, which gives us the probability of a sequence of words being at the beginning of the sentence. Thus the probability of a full stop "." immediately preceding the BOS will be low. We take the product of probabilities of words up $l+1$ as we also add an end of sentence tag EOS. This serves a similar function to the BOS tag, where the probability of EOS preceding a full stop should be high, but also insures that the probability distribution sums to unity, $\sum p(s) = 1$ [41].

To estimate the probability of $p(w_i | w_{i-n+1}^{i-1})$, the probability that a word w_i follows the sequence w_{i-n+1}^{i-1}, we take the counts of $w_{i-n+1}^{i-1} w_i$ from the training data and normalise:

$$p(w_i | w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i-1}, w_i)}{\sum_{w_i} c(w_{i-n+1}^{i-1}, w_i)}. \tag{2.5}$$

By using context we are thus able to build a fluent sentence. For example, the counts in the data may tell us that "home" is more frequent than "house," $c(\text{home}) > c(\text{house})$, yet the frequency of $c(\text{the home}) < c(\text{the house})$. In this way language models are able, as well as in aiding word order choices, to play a role in deciding between possible translations of the same foreign word/phrase. So if we are are translating the German phrase "Das Haus", whilst "home" may be the most likely translation of "Haus", given

the context of what has been translated so far it may be more appropriate to translate it as "house" instead.

Thus, where there are multiple translations for a given foreign word or phrase, the language model, even without any knowledge of the foreign sentence or language, is able to aid in choosing the most appropriate translation. Without the language model, the translation system would simply pick the highest scoring translation from the probabilistic phrase table for each foreign word/phrase pair. The output is nonsense.

The problem with applying Equation 2.5 directly in an SMT system is that if a word in the translation hypothesis is unseen in the training data for the language model, we will get $p(w_i|w_{i-n+1}^{i-1}) = 0$. Increasing the size of the training set can in no way guarantee that we will not come across an unseen word, giving rise to a sparsity problem. To deal with this we can apply smoothing, a class of techniques that aim to smooth a distribution by making it more uniform, including assigning some weight to otherwise zero count events.

One simple smoothing technique, called additive or laplace smoothing, is to simply add a constant to all counts. In this way unseen tokens now have a small but non-zero probability. Other, better performing techniques, include Kneser-Ney smoothing [41, 96], and stupid-backoff [22], which as its name suggests is rather simple, but is easier to train on large data-sets and provides comparable results to Kneser-Ney when doing so. For a full discussion of the plethora of different smoothing techniques, see [41]. Throughout the thesis we use Kneser-Ney smoothing unless otherwise stated.

2.2.3 Reordering Models

There are two main reordering models, often used together, which give a score (distance) and probability (lexicalised) of translating a certain phrase in a source sentence, given the previous phrase translated. The distance score ignores the content of the phrase itself, and relies purely on the distance between the previous phrase translated in the source sentence, and the current one. Larger distances, or jumps, accrue a large penalty.

The lexicalised probability, as discussed in [185, 188], is composed of three different distributions: (1) monotone, (2) swap and (3) discontinuous. Depending on the orientation of the current phrase application, one of the three is used. Given that we apply the distributions in both directions, and along with the distance metric function, we have seven functions for scoring the reordering used in total.

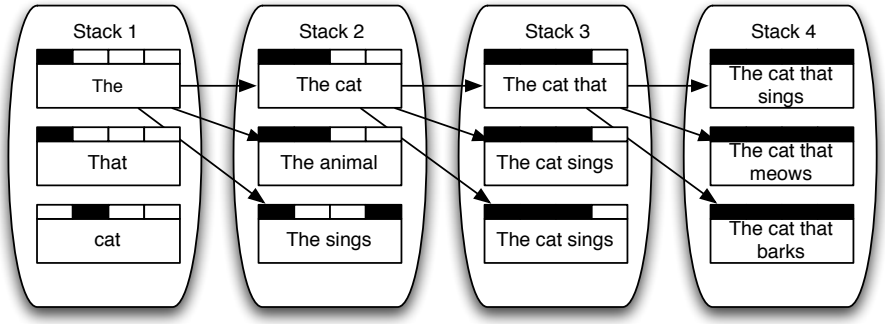


Figure 2.2: A trivial example of the stack based decoding framework used in this thesis, given the French sentence to translate "Le chat qui chant." We display bit vectors above the different states in the translation stacks.

2.2.4 Phrase and Word Penalty

Word penalty. To control translation length, the word penalty feature adds a factor w for each word in the translation. If $w < 1$ shorter outputs are preferred; if $w > 1$, then longer translations are preferred. The word penalty can be used to alter translation length towards very different texts, from the short tweets examined in Chapters 4 and 5, or the longer newswire and parliamentary sentences tested on in Chapters 6 and 7. The word penalty is also used to negate the fact that the language model (discussed in Section 2.2.2) tends to prefer shorter sentences.

Phrase penalty. To recap, there are multiple phrase segmentations possible during translation, meaning there are still, even ignoring reordering and translation options, multiple ways of translating the same source sentence. To control the number of segmentations used, where more segmentations means shorter phrases are translated, a uniform phrase penalty p is applied. As with the word penalty, $p < 1$ favours more phrase applications, $p > 1$ favours fewer.

2.3 Decoding

So far we have introduced the most common models used during the translation process, but not how they are used to create the best translation for a given foreign sentence. The search for the translation maximising Equation 2.2 is called *decoding*, and

an algorithm that implements a particular method for conducting this search is called a *decoder*. Because we have multiple ways of segmenting the foreign sentence, multiple translations for each phrase, and multiple ways of reordering the phrases in the translation, the consideration of all possible translations during the decoding process, and therefore the maximisation over all possible translations in Equation 2.2, is often computationally infeasible. Precisely for these reasons, SMT has been shown to be NP complete [97].

There are a number of different approaches to conduct the search for a translation given a foreign sentence. Approaches can be categorised into four distinct categories; string-to-string, string-to-tree, tree-to-string and tree-to-tree. String-to-string approaches include two of the most common decoding types: stack based decoding using beam search and hierarchical decoding using chart parsers based on the CYK algorithm. The hierarchical (Hiero) approach [44, 45] uses synchronous chart parsing to build up a non-linguistic parse tree, as the non-terminal vocabulary in the context free grammar is made up of arbitrary symbols X. The string-to-string approach of Hiero can be expanded into one of either tree-to-string, string-to-tree, or tree-to-tree approaches. One common approach, syntactically aware machine translation (SAMT) [212], builds upon the Hiero approach by using a syntactified grammar on the target side, so that the parse forest resembles that of a typical PCFG parser. Instead of using a synchronous CKY algorithm which works in a top-down manner, the stack based approach, which in its basic form uses no syntax at all, creates translations in a left-to-right manner [106].

A comparison of different decoders can be found in [213]; though depending on the language task, different decoders perform differently on different language tasks. In this thesis, we use the stack-based approach primarily because of its consistent use in the SMT community, along with the availability of systems such as Pharoah [98] and Moses [107] that implement it. We now proceed to explain how the stack-based algorithm works.

The search begins with an empty hypothesis or state, which we expand into other hypotheses by picking a part of the source sentence that has yet to be translated, and seeing if we have a phrase translation for that sequence of source of words. If we do, we create a new state, translate the source words and mark them off as being translated by switching the relevant bit to one in a bit vector, and finally we append the translation to the existing partial translation in a left to right manner. The resulting states are then stored in stacks according to the number of foreign words translated. As is often the case, there will be more than one translation for a given source string within the source sentence, and so those different translations are applied, creating different unique states. The scores from the various feature functions that make up our SMT system, that is those described in Sections 2.2, are applied and states are ranked accordingly. The best translation is the best scoring state in the last stack, which cor-

responds to the whole source sentence being translated. Using back-pointers stored in the state, we can then re-construct the best translation, or output an n-best list for reranking and optimisation purposes if need be.

During the decoding process, as well as applying the model scores to each hypothesis, we need to take into account the fact that some phrase segmentations may be easier to translate then others. Because partial translation candidates are stored in stacks based purely on the number of source words translated, certain translation candidates may be unfairly penalised due to having translated more difficult parts of the source in comparison to others. To counteract this, for each hypothesis we take into account the 'future cost' when ranking states.

The future cost is an estimate of the score of the remaining phrase segmentations to be translated, and takes into account a simplified, optimistic, estimate of the phrase translation, reordering and language model scores. The future cost is precomputed for each source sentence using a dynamic programming algorithm. Then, depending on the bit vector of the current hypothesis under consideration, a single look-up can be made to retrieve the cost. By taking into account the future cost estimate when ranking hypotheses, better pruning decisions (discussed below) can be made.

To illustrate the stack-based translation decoder, we provide a trivial example in Figure 2.2 of the translation process given the foreign sentence "Le chat qui chant." To begin, we start with a dummy start state, not shown. Then a source phrase to translate is chosen, a hypothesis is created, and scored according to the current model costs along with the future cost estimate and stored in the appropriate stack. Once all translations of all source phrases are examined, we move to the first stack, and for each state, we repeat this process. We continue in this manner moving from stack-n to stack-$n+1$.

So far we have discussed the stack-based decoder, but have not mentioned how the search space is managed. With no limits, a stack-based decoder would still run into computational challenges. To restrict the search space, we apply four heuristics. The first, the beam width, is based on the the score of the translation relative to a specific threshold from the highest scoring state in the same stack. The second is an arbitrary stack limit; any state falling outside of the stack-limit is dropped and forgotten. Third, we limit the reordering distance allowed between the current and next phrase segmentations. Finally, we limit the number of translation candidates available to each source phrase.

As well as the heuristics just discussed, we can take into account the observation that there are different ways of reaching the same translation, or the same partial translation. Utilising this fact, we can apply a lossless method, hypothesis recombination, to reduce the search space further.

Two states are candidates for hypothesis recombination when they share the same bit vector, the same $n-1$ words as their history, and the current sentence position, as

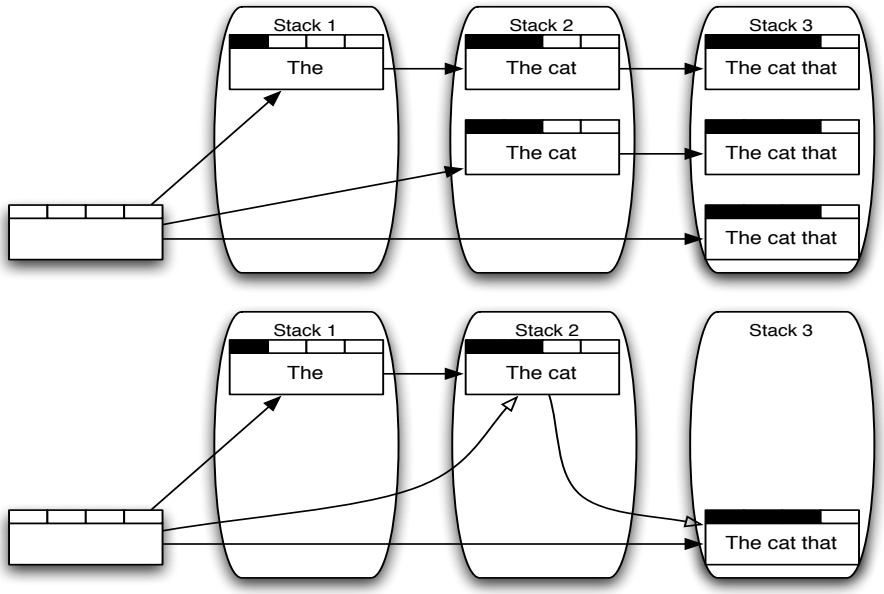


Figure 2.3: A recombination example. In the top row, we show different search paths when recombination has not been used; in the bottom the states that arise out of recombination.

well as the position of the last translated phrase. This is because, while the phrase table only bases its score on the current phrase-application, the language model uses the $n-1$ words to compute its score, and the reordering models require the position of the last foreign sentence translated to compute their scores. If two candidates are a match for recombination, the state with the higher score is kept, and the other is dropped from the stack. It is important to note that the dropped hypothesis can never be part of the best translation path.

In Figure 2.3 we provide an example of the recombination of states. On the top row, there are 3 different paths that lead to the same translation. In the bottom row we show the same space with states recombined. When states are dropped, we move the back-pointer from the worse state to the new one. This is important when we want to consider creating n-best list and retrieving not just the best but the top-n best translations.

2.4 System Optimisation

As seen in Equation 2.3, a translation system is composed of different features, or models, that each have different opinions (expressed in terms of raw scores or probabilities) about the translation space. The phrase table probabilities may prefer one translation over another for a given source segment, however the language model may prefer another translation given the context. We need to find the set of weights that gives the best translation quality for unseen data.

To do this, we need a development set. The development, or tuning set, is a corpus of sentences, similar to the bi-text used for training our phrase tables, for which we have one or more human translations. This development set is usually between 500-2000 sentences long.

Given this set, there are a number of methods one can use to find the optimal weights, ranging from those based on maximum entropy [138], or learning-to-rank approaches [81], to those based on minimising the error with respect to some pre-defined metric [136]. Because methods based on minimum error rate training (MERT) are the most widely used in the SMT literature, and because they are available with popular open source translation toolkits such as Moses [107], we use MERT throughout the thesis for optimising our translation system.

MERT has some well know issues; in particular it is unable to handle feature numbers that go much beyond the twenties. In such a setting, the learning-to-rank approach has been shown to perform better [81]. However, in this thesis, we do not examine scenarios that would necessitate such a method, and as much of the work presented in this thesis predates the learning-to-rank approach and its acceptance as a viable alternative to MERT in the research community, we use MERT throughout.

2.5 Summary

In this chapter, we have provided a detailed introduction to the background of SMT, including the overarching structure of an SMT system, the models used, the decoding process and parameter learning. In the next chapter we discuss the experimental methodology that we follow in the thesis as well as the common experimental environment used in the later technical chapters. Then, in Part I, we address our exploration theme, concerning the acquisition of more (relevant) training data. In Part II of the thesis we address our second research theme, on better leveraging existing resources for machine translation purposes.

Chapter 3

Experimental Methodology

In the previous chapter we described the general architecture of an SMT system and the underlying models that underpin the work in this thesis. In this chapter, we introduce the experimental methodology generally employed in SMT and also used in this thesis. We begin by discussing the notion of automatic translation evaluation, detailing the standard metrics used in Section 3.1, then proceed to discuss the significance tests we use in Section 3.2, and finish with a discussion of the data sets used in the technical portions of this thesis.

3.1 Evaluation

Conducting research into novel methods and algorithms for improving on existing SMT systems entails the need for a method for evaluating an individual system's performance in some consistent standardised manner, against which multiple systems can then be compared. Humans, bilingual in the language pairs of the translation systems under examination, are and have been used for precisely this purpose. They can be either professionals, or appropriately qualified researchers, who are typically asked to judge the translations output by a system on a numeric scale according to both adequacy and fluency [105, 134]. In this way the human judges are being asked to score not just the quality of the output translation, but to what extent it captures the semantics, the meaning, of the original foreign sentence.

While arguably the best way to evaluate and compare systems is through the use of human judges, they are financially expensive, and introduce a delay into the ex-

perimental process. Thus, the use of automatic evaluation metrics, which can tell the researcher quickly what score a system has, and if any changes had a positive or negative impact on the quality, are widely used. In the next section, we describe the many metrics available to the researcher in the SMT field.

3.1.1 Automatic Metrics

There are various automatic metrics which assess translation quality by assigning a certain score that can then be used for evaluation and comparison. The underlying mechanism behind all the presented metrics is that they compare the translations of the machine translator to a reference set. We now proceed to describe the main evaluation metrics used in the research community for SMT.

- **WER** Word Error Rate [166] is computed as the Levenshtein distance between a translation and reference sentence, divided by the length of the reference translation. WER can not take into account multiple reference translations, and does not take into account word and phrase reordering.
- **TER** Translation Error Rate [181] calculates a metric based on the number of edits a human would have to make to convert the system translation into a reference translation, and expands on WER by allowing for block shifting of words, where the edit cost is the same as that of editing, inserting or deleting a word. TER-Plus [181] has been proposed as an extension to TER, with optimisable edit-costs and features including paraphrases, stemming and and synonyms.
- **PER** Position-independent Error Rate [190] is an even simpler version of WER, where the position of words is not taken into account when counting the number of matching words.
- **METEOR** Standing for Metric for Evaluation of Translation with Explicit Ordering [13], METEOR combines precision and recall with a penalty for disfluency based on the fragmentation of n-grams that match the reference translation. Unfortunately, METEOR requires optimisation to best correlate with human judgements. Continued research into METEOR has taken place with a number of publications, including [4, 58, 59, 113, 114].
- **BLEU** Proposed in 2002 [144], BLEU is the main automated evaluation metric used within the Statistical Machine Translation field; for instance, it is the official metric of the NIST OpenMT evaluation campaign [135]. It is based on counting the number of n-gram matches between the translation and one or more multiple reference sentences.

As BLEU is widely used in the research community and in evaluation campaigns [135], and has been shown to correlate well with human judgement [61, 115, 155], we use this metric throughout the thesis when reporting on translation quality. We now describe BLEU in detail.

3.1.2 BLEU

Reliant on a corpus of human translations, the BLEU metric measures the similarity between MT output and the reference translations through the number of n-grams that occur in both sets. As there can be many different acceptable translations for a given sentence, BLEU can take multiple reference translations into account. Often data sets will come with somewhere between 2 to 4 reference translations, though BLEU still works when there is only one single reference translation per source sentence available. BLEU is represented as;

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right),$$

where the brevity penalty, BP, is defined as;

$$BP = \left\{ \begin{array}{ll} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{array} \right\}, \tag{3.1}$$

and where r is the effective reference length for the entire test corpus, computed by summing the best match lengths for each candidate sentence in the corpus, and c is the total length of the candidate translation corpus. The brevity-penalty is required to penalise translations that are too short and would otherwise have too high a score according to the precision metric p_n. Given $Count_{clip}(\cdot)$, which either returns the number of times an n-gram occurred in the candidate, of if this happens to be more then any of the occurrences in the reference translations, then the highest count from the reference translations, the metric p_n itself is defined as:

$$p_n = \frac{\sum_{C \in (Candidates)} \sum_{n-gram \in (C)} Count_{clip}(n\text{-}gram)}{\sum_{C' \in (Candidates)} \sum_{n-gram' \in (C')} Count(n\text{-}gram')}. \tag{3.2}$$

BLEU is not an infallible metric; Callison-Burch et al. [27] show that the measure does not necessarily correlate with human judgments, and that an improved BLEU score does not always mean an improvement in translation quality.

3.2 Statistical Significance Testing

Given BLEU, we have a metric to evaluate the translation performance of any given translation system. However, the question we wish to ask is, given two different systems, if one is better than another. Statistical significance testing can help us answer the question in a robust and secure way; that is, which system is truly better, and not better due to random chance. To answer this question, significance tests are composed of two hypothesis:

- **Null hypothesis** In this hypothesis both systems are the same, and any difference in evaluation metric scores is due to pure random chance.
- **Alternative hypothesis** The difference in system scores is reflective of a true underlying difference in the translation quality of each system.

During significance testing, it is the second hypothesis we wish to prove; i.e., one system is better or worse than another. Significance tests cannot give a binary answer in the form of a certain yes or no. They can however give us the degree to which they believe the alternative hypothesis to be true. This degree is often represented as a p value, where a $p < 0.05$ indicates that there is less than five per cent chance that the null hypothesis is true and the output from both systems comes from the same underlying distribution. P values are pre-set; in this thesis we use values of $p < 0.05$ and $p < 0.01$.

We use the paired bootstrap resampling method in this thesis [99]. We begin by creating new virtual test sets by sampling with replacement from the original test sets. We then compare the BLEU results for each of the virtual test sets, and see how many times one system beats the other. If one system beats the other 99% of the time, then we say that it is better with 99% statistical significance.

3.3 Toolkits

For training our language models, we used the SRILM toolkit [182], and build models using the Kneser-Ney and interpolate options, unless otherwise stated. We build language models of size 5, i.e., the probability of each word is based on the previous four words.

We used the GIZA++ toolkit to extract word alignment over any given bi-text [137]. To extract phrases from resulting word alignments and estimate lexical and phrasal translation probabilities, we used scripts that come with the Moses decoder [107]. We also used Moses as our main decoder for translating foreign test collections.

3.4 Test Collections

In this section we describe the data sets that are common to multiple technical chapters. In the first two technical chapters of this thesis (Chapters 4 and 5), we had to acquire and manually annotate our own collections for the purposing of evaluating the quality of the proposed language identifiers and in-domain SMT systems. Thus, we describe the test collection and the methodology for acquiring it in the respective chapters themselves.

3.4.1 Europarl Collection

In Chapters 5 and 7 we use parts of the Europarl corpus [100] for training our translation model and language models. The data is composed of a crawl of the Proceedings of the European Parliament. The corpus has undergone a number of revisions since it was first released, growing from the original 11 languages to, at the time of writing, 21 languages. This is a collection that is routinely used for training and evaluating SMT systems; see for example the workshops on machine translation(`http://www.statmt.org/`). In this thesis, we kept with commonly used splits of the data, that is, reserving the fourth quarter of the year 2000 for parameter optimisation and testing purposes.

For building the LMs we use the English side of the Europarl training portion. For building the phrase tables using this corpus, we first have to run the document and sentence alignment scripts, which are based on the work of Gale and Church [70], prior to having a parallel corpus on which we can extract word alignments with GIZA++.

3.4.2 NIST MT-Eval Sets

This collection comprises test data provided by the NIST open Machine Translation Evaluation, which has run in the years 2002 through to 2009 (skipping 2007). Language pairs examined include Arabic-English, Chinese-English, and Urdu-English. We refer to the sets henceforth as MT02, MT03, and so on.

3.5 Summary

In this chapter we have introduced our experimental environment, including detailing the automatic evaluation metric, BLEU, which we use throughout, the toolkits we use to conduct experiments, and the data sets that are common to more than one technical chapter.

Part I
Exploration

Mining the plethora of documents available online for learning new translations is not a new task. Yet, increasingly, more content on the web comes from informal, unedited, self published sources, such as forum comments, blog posts, and more recently, microblog posts. However, the majority of previous research approaching the task of mining new translations from web based documents have focused on structured, formal, edited texts, predominately emanating from news and broadcast organisations. Little is known about the behaviour of such algorithms when applied to content that displays properties that are completely opposite of such newswire articles in terms of style, grammaticality and degree of colloquial term usage. In the first part of the thesis, we examine and tackle the challenges specific to working with such informal, unedited text, with the aim of exploring large amounts of microblog posts for learning new, domain-specific translations.

We begin in Chapter 4 by studying the task of classifying microblog posts, extracted from Twitter, a popular platform, into distinct languages. We then examine in Chapter 5 a number of different approaches that use or mine these tweets for additional, in-domain, tweet specific, translations. Our contributions in Part I include: an analysis of the problems of identifying the language of tweets; the proposal of a number of models that utilise various additional sources to improve accuracy; an analysis of language use on a large-scale corpus of microblog posts; and a comparison of two distinct approaches to using the language defined corpora for improving an out-of-domain SMT system for translating tweets.

Chapter 4

Language Identification

Microblogging platforms such as Twitter have become important real-time information resources [73], with a broad range of uses and applications, including event detection [167, 198], media analysis [7], mining consumer and political opinions [84, 194], and predicting movie ratings [142]. Microbloggers participate from all around the world contributing content, usually, in their own native language. Language plurality can potentially affect the outcomes of content analysis and retrieval of microblog posts [123], and we therefore aim for a monolingual content set for analysis. To facilitate this, language identification becomes an important and integrated part of content analysis. We address the task of language identification of microblog posts in this chapter.

Document length has been shown to significantly affect language identification, shorter documents being much harder to identify successfully [11]. To show that microblog language is a challenge in itself, we perform an initial experiment on short formal texts versus short microblog texts. In particular, for each language, we use documents from the EuroParl corpus [100] and from those we select sentences less than 140 characters long. We randomly sample 1,000 sentences per language, from which 500 are used for training and 500 are used for testing. Table 4.1 shows the performance of our baseline model (detailed in Section 4.2) on the formal (EuroParl) language documents and the microblog posts. Results clearly indicate that language identification on the idiomatic microblog language is more challenging than on formal texts of equal length, with the two systems significantly different according to the p-test (see Section 4.4.1 for details on the dataset and significance test).

In this chapter we present novel language identifiers that use additional sources of information, aside from the text in the tweet itself, to aid in classifying the language of the post. The models we introduce use five *semi-supervised priors*. We explore the effects on language identification accuracy of (i) a blogger prior, using previous microblog posts by the same blogger,[1] (ii) a link prior, using content from the web page hyperlinks within the post, (iii) a mention prior, using the blogger prior of the blogger mentioned in this post, (iv) a tag prior, using content of posts tagged with the same tag, and (v) a conversation prior, using content from the previous post in the conversation.

	Dutch	English	French	German	Spanish	Overall
Formal	99.6%	98.6%	99.4%	99.4%	99.8%	99.4%
Microblog	90.2%	94.8%	90.0%	95.8%	91.2%	92.4%

Table 4.1: Accuracy for language identification on formal language (EuroParl) and microblog language.

Besides exploring the effects of the individual priors on language identification performance, we also explore different ways of combining priors: we look at post-independent and post-dependent combination models. For the post-dependent combination models, we introduce two ways to measure the confidence of a prior. The confidence of a prior can then be used in a linear combination model. We compare these post-dependent combination models to two post-independent models, (i) a linear combination model with fixed weights, and (ii) a voting model.

The final aim of this chapter is to demonstrate the application of our novel language identifiers in a real world large-scale setting, where we classify over 1.1 million microblog posts that were posted in a single day in March 2011. We demonstrate that explicit geo-location and language metadata fields are too infrequently used to be relied upon as additional features, and that the majority of posts are non-English, corroborating our initial motivation that a large of amount of content needs to be translated if we wish to know what is being communicated.

To summarise, we aim at answering the following main research question in this chapter:

RQ 1. Can we successfully identify the language of a microblog post, given the short, idiomatic and unedited nature exhibited?

[1] We use the term blogger, instead of Twitterer, as Twitter is an instance of a microblog platform.

This general research question gives rise to the following five specific research questions.

RQ 1a. What is the performance of a strong language identification method for microblogs posts?

RQ 1b. Does domain-specific training of language models help improve identification accuracy?

RQ 1c. What is the effect on accuracy of using priors extracted from microblog characteristics?

RQ 1d. Can we successfully combine semi-supervised priors in a post-independent way?

RQ 1e. How can we determine confidence of individual priors, and can we use confidence to combine priors in a post-dependent way?

In answering these research specific questions, we present the following contributions in this chapter:

- We explore the performance of a strong language identification method on microblog posts.
- We propose a variety of novel algorithms to help identification accuracy in sparse and noisy data, and provide the research community with a manually annotated dataset on which results can be reported and compared.
- We introduce confidence metrics that can be used to weight the additional content sources.
- We performs an in-depth analysis of identification results in a real-world, large-scale, setting.

The remainder of the chapter is organised as follows: in Section 4.1 we explore previous work in this area. In Section 4.2 we introduce our baseline model, and the semi-supervised priors. Section 4.3 talks about combining priors, and introduces our confidence metrics. We test our models using the setup detailed in Section 4.4.1, and in Section 4.4.2 we present our results. We analyse and discuss the results in Sections 4.4.3 and 4.4.4. Finally, we present an analysis of the classification of 1 million tweets published in a single day in March in Section 4.5, and conclude in Section 4.6.

4.1 Related Work

Language identification can be seen as a subproblem in text categorisation. Cavnar and Trenkle [37] propose a character n-gram-based approach to solving text categorisation in general, and test it on language identification. Their approach compares a document "profile" to category profiles, and assigns to the document the category with the smallest distance. Profiles are constructed by ranking n-grams in the training set (or the document) based on their frequency. These ranked lists are then compared using a rank-order statistic, resulting in a 'out-of-place' (OOP) distance measure between document and category. Tested on a set of Usenet documents, it achieves an accuracy of 99.8% for language identification.

Other approaches to the n-gram OOP method have been examined in [11, 14, 208]. This chapter differs in that we examine the utility of microblog priors, as opposed to comparing different classification algorithms. Note that the priors presented in this work could easily be integrated into other models (e.g., Naive Bayes, SVM).

Accuracy is often high when looking at structured and well-written documents (above 98% [37]), however research has been done examining different types of text. Language identification on web pages already seems more difficult: Martins and Silva [122] test an n-gram-based approach with web-related enhancement, and show that accuracy is between 80% and 99%, depending on the language. Another interesting study by Baldwin and Lui [11] also explores the impact of document length on language identification. They test language identification on Wikipedia pages, and show that performance on this task improves with growing document length: Accuracy for longer documents reaches 90%, whereas this is only 60–70% for shorter documents. Finally, interesting work examining the language identification of query like short text is done by Gottron and Lipka [75]. The authors explore performance of language identification approaches on "queries" (news headlines), which are, on average, 45.1 characters long. They achieve high accuracy results of 99.4% using 5-grams, but focus on short newswire text, without the idiomatic limitation imposed by the social media domain (the impact of which is demonstrated in Table 4.1), as examined in this work.

4.2 Language Identification Components

Based on previous work, we opt for using an n-gram approach to language identification. More precisely, we use the TextCat[2] implementation of the approach described in [37]. This model has shown good and robust performance on language identifica-

[2] `http://www.let.rug.nl/~vannoord/TextCat/`

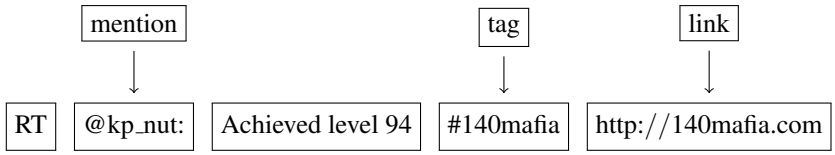


Figure 4.1: An example tweet with the three surface features used in our model highlighted.

tion. In the previous section we explained how TextCat works to identify a document's language. We use the TextCat algorithm for language identification on our microblog post set and study the effect on accuracy of language models trained on different data sets. We consider two types of language models: (i) *out-of-the-box*, which uses the training data supplied by TextCat, and we set this as our baseline, and (ii) *microblog*, for which we use a training set of posts from our target platform to re-train TextCat.

More formally, let z be the total number of languages for which we have trained language models and $i \in \{1, \ldots, z\}$ denote the corresponding model for a language. For each post p we define a language vector

$$\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}} = \langle \lambda_p^1, \lambda_p^2, \ldots, \lambda_p^z \rangle, \tag{4.1}$$

where λ_p^i is a score denoting the distance between post p and language i: the smaller the distance, the more likely it is that post p is written in language i. In the remainder of the chapter, we refer to vectors constructed from the microblog post itself as *content-based identification* vectors, written as ${}_{\boldsymbol{C}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}}$.

4.2.1 Semi-Supervised Priors

On top of the language identification on the content of the actual post, as described above, we use five semi-supervised priors to overcome problems due to sparseness or noise (see the introduction to this chapter, in particular Table 4.1), and help improve the accuracy of our baseline classifiers. Our priors are (i) semi-supervised, because they exploit classifications of the supervised language identifier on unlabeled data, for which we do not know beforehand the true language, and (ii) priors, because they allow us to identify the language of a post without content-based identification. Given the setting of microblogs, we are offered several natural priors. The example tweet in Figure 4.1 shows three surface features we plan to exploit as priors. Besides these three surface features, we also use priors based on the conversation and blogger history.

Link prior: posts in microblogs often contain links, referring to content elsewhere on the web. This content is often of longer text length that the post itself. We identify the language of the linked web page, and use this as link prior for the post that contains the link. Let $L = \{l_1, \ldots, l_k\}$ be a set of links found in post p. For each web page $l_i \in L$ we apply the *out-of-the-box* model to its content, and construct a link prior vector from the average of content-based identification vectors of web pages found in p:

$$_{\boldsymbol{L}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}} = \frac{1}{k}\sum_{i=1}^{k} {}_{\boldsymbol{C}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{l_i}}. \tag{4.2}$$

Blogger prior: behind each post is a blogger who wrote it, and probably the current post is not her first; there is a post history for each blogger the content of which can be beneficial for our purposes. By identifying (or guessing) the language for previous posts by the same blogger, we construct a blogger prior for the current post. Let $P = \{p_1, \ldots, p_k\}$ be a set of posts predating p from blogger u. For each $p_i \in P$, we use the *microblog* language models, and construct $\hat{\boldsymbol{\lambda}}_{\boldsymbol{p_i}}$, as explained before. We then derive a blogger prior from the average of content-based identification vectors of previous posts:

$$_{\boldsymbol{B}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}} = \frac{1}{k}\sum_{i=1}^{k} {}_{\boldsymbol{C}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p_i}}. \tag{4.3}$$

Mention prior: as a social medium, microblogs are used to communicate directly between people. Post in microblogs are often directed to one or several specific persons, indicated by a special token. We can identify the language for these users that are addressed, and use this information as mention prior. Let $U = \{u_1, \ldots, u_k\}$ be a set of bloggers mentioned in post p. For each $u_i \in U$, we build a blogger prior ${}_{\boldsymbol{B}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{u_i}}$ as in Eq. 4.3. We derive the mention from the average of blogger priors:

$$_{\boldsymbol{M}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}} = \frac{1}{k}\sum_{i=1}^{k} {}_{\boldsymbol{B}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{u_i}}. \tag{4.4}$$

Conversation prior: certain posts form part of a specific conversation between individuals, as opposed to being part of a more general conversation between numerous bloggers. When this is the case, it is safe to assume that this conversation is

taking part in a single language common to both bloggers. Posts that are part of a conversation are not recognizable as such from the content, but this information is stored in the post's metadata. Let p_{i-1} be the previous post in the same conversation as post p. We use the *microblog* language model to construct ${}_{\boldsymbol{C}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p_{i-1}}}$, as explained before, and use this as the conversation prior ${}_{\boldsymbol{V}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}}$.

Tag prior: bloggers often contribute to a corpus of microblog posts on a specific topic, where the topic is represented by a tag. This corpus of posts, i.e., posts that share the same tag, can be beneficial for our purposes. We derive a tag prior based on the average over microblog posts that share the same tag. Let $T = \{t_1, \ldots, t_k\}$ be a set of posts predating p in the corpus of tag T. For each $t_i \in T$, we use the *microblog* language models, and construct ${}_{\boldsymbol{C}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p_i}}$, as explained before.

$$ {}_{\boldsymbol{T}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}} = \frac{1}{k} \sum_{i=1}^{k} {}_{\boldsymbol{C}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{t_i}}. \tag{4.5} $$

Since scores generated by TextCat are not normalised by default, for all priors that require averaging, that is all those except the conversation prior, we normalise the raw scores using z-scores. Our language identification approach leaves us with a content-based identification vector and five semi-supervised priors. For ease of reference, in the rest of the chapter, priors will refer to these five priors and the content-based identification vector, unless clearly stated otherwise. The next section details how we combine these vectors into one, and obtain our final estimate of a tweet's language.

4.3 Combining Priors

The combination of priors and the content-based identification is a form of "evidence combination" and we have two obvious ways of going about it: (i) treat all posts equally, and use post-independent combination models, or (ii) observe each post individually, and use a post-dependent model to combine evidence. For the first combination approach we need training data, and for the second approach we need a way to determine which priors are most reliable for a given post. In this section we explore both aspects: Section 4.3.1 introduces the post-independent combination models and Section 4.3.2 discusses the post-dependent combination, with a focus on the *confidence metrics* that can be used. After discussing our models and metrics here, we introduce our dataset in the next section and discuss how we train our models.

4.3.1 Post-Independent Combination

In this section we present two different ways for post-independent prior combination. The first approach uses post-independent weight optimisation for linear interpolation and the second is based on voting, a technique for combining multiple classifiers.

Linear interpolation with post-independent weight optimisation

To create a linear model, we first construct vectors for the content, and each of the priors, with scores for each language, and combine these vectors using a weighted linear combination. More formally, we identify the most probable language for post p as follows:

$$lang(p) = \arg\max \sum_{q} w_q \cdot {}_{\boldsymbol{q}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}}, \tag{4.6}$$

where $q = \{L, B, M, C, V, T\}$. This model has two important components: first, to make ${}_{\boldsymbol{q}}\hat{\boldsymbol{\lambda}}_{\boldsymbol{p}}$ suitable for linear interpolation, we need to normalize the values. Scores are normalised using z-scores. The second component is w_q, the actual weight of the prior q. To find the optimal weights for each prior, we perform a sweep over the parameter space in an interpolated model over all priors. We optimize for overall accuracy (accuracy over all five languages) on our development set (see Section 4.4.1). The post-independent weight optimisation approach does not take post-specific features into account and requires training data for the weights.

Majority voting

As well as trying sweeps for the optimal linear interpolation parameters, we explore the use of voting for classifying a post. Majority voting is a principled way to combine classifications from multiple classifiers [60]. Majority voting applies each classifier to the input, in this case a post, takes the classifications, and selects the label that was assigned most. As long as each individual classifier performs better than chance, it has been shown that this approach can lead to a better performance than relying on any single classifier [60].

The main issue with majority voting is how to deal with ties: the case where multiple labels receive an equal number of votes. In our case, we use the normalised scores for solving ties. When a tie occurs, we select the label (language) that has the highest normalised score over all priors. Although more ways of solving ties are possible, experiments on the development set show this approach is successful. The advantage of the majority voting approach is that it is quite insensitive to fluctuations in scores, since it only relies on votes. On the other hand, ignoring scores also means the loss of (potentially valuable) information on the strength of priors.

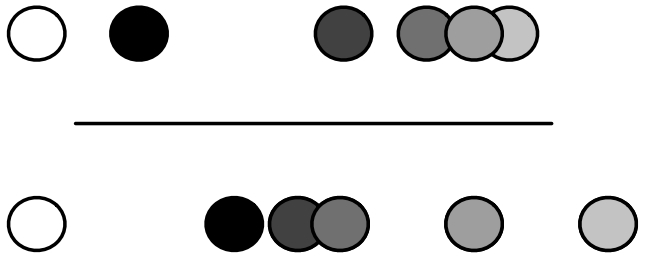


Figure 4.2: Two graphical representations of confidence, with a confident prior (top) and uncertain prior (bottom). The white dot represents the post profile and the shaded dots represent the profiles of different languages.

4.3.2 Post-Dependent Combination

The aim of a post-dependent model is to vary the weights of the priors that give optimal classification results for that specific post. Here, we propose to use a post-dependent linear combination model. This model is similar to the one introduced in Eq. 4.6, where each prior is weighted. Unlike the post-independent linear interpolation, however, we cannot learn these weights, since we only have one instance from each post. In this section, we introduce two ways of estimating the *confidence* of each prior, which can be used in our linear combination.

To explain the notion of confidence, observe the two situations in Figure 4.2. The top half shows a situation where the prior is very confident: one language (the black dot) is close to the post (white dot), and the other languages (shaded dots) are quite far away. This prior is confident that this post is written in a certain language. The bottom example shows a different situation, in which several languages (shaded dots) are close to the post: the prior is uncertain as to which language is the right one. We aim to exploit the observations from Figure 4.2, and propose the following two confidence metrics: (i) the beam confidence, and (ii) the lead confidence.

Beam confidence

The beam confidence builds on the observation that when multiple languages are close to the most likely language, the prior is less confident. To concretize this observation, we use the following reasoning: given a beam b (e.g., 5%), we calculate a limit distance based on the (raw) distance of the most likely language. Languages are ordered by their raw scores, from lowest to highest. The language first in this list is the most likely. This limit distance is defined as $limit(p) = d(\lambda^1) + b$, the raw distance of the most likely language increased by the beam (in percentages). We then move on to the next most

likely language, and see if this language is closer to the post profile than the limit. If this is the case, we add this language to the list of languages "within the beam," $LIB(p)$, and repeat with the next most likely language. If not we stop. Eqs. 4.7 and 4.8 show how we calculate the $LIB(p)$ for post p over all languages λ.

$$LIB(p) = \sum_{i=2}^{k} inBeam(\lambda^i) \tag{4.7}$$

$$inBeam(\lambda^i) = \begin{cases} 1 & \text{if } d(\lambda^i) < d(\lambda^{i-1}) + b \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

where $d(\lambda^i)$ is the raw distance between the post profile and the language profile.

We now have the number of languages that falls within the beam; from this we can calculate a weight for the prior. We use both a linear and exponential function to obtain the final weights. The linear function is defined as follows:

$$weight(p) = \frac{|\lambda| - LIB(p)}{|\lambda| - 1} \tag{4.9}$$

The exponential function uses an exponentially increasing punishment for more language in the beam:

$$weight(p) = e^{-LIB(p)+1} \tag{4.10}$$

Lead confidence

The second confidence metric we introduce is the lead confidence. This metric tries to capture the lead the most likely language has over its closest "rival" language. The further away a language is from its nearest rival, the more confident a prior apparently is about this language. We use a fairly straightforward approach to measure the lead confidence: we take the difference between the first $d(\lambda^1)$ and second $d(\lambda^2)$ ranked languages normalised scores. We take this difference as the weight of the prior:

$$weight(p) = d(\lambda^1) - d(\lambda^2) \tag{4.11}$$

4.4 Experiments

We begin by detailing the experimental setup we use in Section 4.4.1, and presenting experimental results in Section 4.4.2.

4.4.1 Experimental Setup

For testing our models we need a collection of microblog posts. We collect these posts from one particular microblog platform, Twitter. We test our models on a set of five languages, Dutch, English, French, German, and Spanish, and gather an initial set of *tweets* by selecting tweets based on their location. From this initial sample, we manually select 1,000 tweets in the appropriate language: tweets that contain only non-word characters (e.g. URLs, emoticons) are ignored. For multilingual tweets, we assign the language that is most "content-bearing" for that post. To determine this, we would remove text belonging to one of the languages, and ask ourselves if the meaning of the tweet can still be ascertained. If it was, then the text of that language was determined to be non content-bearing. In certain situations, the second language did carry some meaning, but was minor in comparison to the main language; e.g., the apology in the first tweet in Table 4.6.

For training purposes, we split each set in a training set of 400 tweets (for TextCat training), a development set of 100 tweets (for weight optimisation), and a test set of 500 tweets.[3] For the blogger prior, we extract as many tweets as possible from the poster's history, which on average is 154.8 posts per user. For the mention prior, of the 2,483 unique users mentioned in tweets, the average number of tweets extracted from the posters history was 129.3. For the hashtag prior, we extract the 200 most recent posts that contain the hashtag posts. We placed no time restrictions on the extraction of such data. Table 4.2 lists several characteristics of the tweets in our training set.

	Number of tweets with			Total number of		
Language	Links	Tags	Replies	Links	Tags	Replies
Dutch	59	77	213	60	94	251
English	123	54	174	123	78	201
French	140	71	183	143	105	217
German	182	107	108	183	219	119
Spanish	103	42	190	103	55	226

Table 4.2: Number of tweets in the training set (400 tweets per language) with at least one link, tag, or reply and the total number of these items per language.

TextCat allows us to select the number of n-grams we want to use for profiling our language and documents. Preliminary experimentation with this parameter revealed

[3] The training data and the trained models are available at `http://ilps.science.uva.nl/resources/Twitterlid`.

that the standard value (top 400 n-grams) works best, and we use this value for the remainder of the experiments. We report on accuracy (the percentage of tweets for which the language is identified correctly) for each language, and overall.

The number of languages examined will impact on the absolute accuracy results reported, both for the baseline system and for the more elaborate methods proposed here. However, our goal in answering the five research questions is to demonstrate a significant increase in performance over the baseline using the methods proposed in this work. For computing significant differences between two models, we use the p-test [207] on the overall accuracy:

$$Z = \frac{p_a - p_b}{\sqrt{2p(1-p)/n}},$$

where $p = \frac{p_a + p_b}{2}$, p_a and p_b are accuracy results of the two systems being compared, and n is the number of tweets classified by both models. Significance levels of 90%, 95% and 99% are referred with $^{!}$, †, and ‡ respectively.

To answer the our research questions, we will conduct the following four runs: (i) in-domain vs out-of-domain models (compare the use of TextCat on our Twitter data with out-of-the-box and microblog trained models) (ii) individual priors (we incorporate all our priors individually with the original language identification based on the content of the tweet), (iii) post-independent prior combination (linear interpolation vs majority vote) , and finally (iiii) post-dependent runs (we compare our beam confidence and lead confidence approaches to combining all the priors).

4.4.2 Results

We design and conduct four experiments to answer our five research questions. Below, we detail each of the four experiments and present the results.

Language identification on microblog posts The first experiment aims at answering the first two research questions, namely, what is the performance of a strong language identification method on microblog posts, and whether domain-specific training can help improve accuracy. Results in Table 4.3 show that language identification on short posts in microblogs is not as straightforward as it is on formal short pieces of text (see Table 4.1, where accuracy on formal text is much higher). The use of the microblog model improves performance by 3% on average, but accuracy is still limited, with Dutch showing no improvement at all.

	Dutch	English	French	German	Spanish	Overall
Out-of-the-box	**90.2%**	88.4%	86.2%	94.6%	88.0%	89.5%
Microblog	**90.2%**	**94.8%**	**90.0%**	**95.8%**	**91.2%**	**92.4%**$^{!}$

Table 4.3: Results for baseline content-based identification runs using the out-of-the-box and the microblog language models.

Individual priors In our second experiment we target our third research question and we study the effect on accuracy of our set of individual semi-supervised priors which we derived from microblog characteristics. We learn the weights of the prior versus the content-based identification on our development set using weight sweeps as explained in Section 4.3.1, limiting the sum of weights to 1, and report on the best performing prior weights in Table 4.4. The results show that incorporating the semi-supervised priors leads to an increase in accuracy for all languages over content-based identification using the microblog model. In particular, among all priors, the blogger and mention priors are found to perform the best, as they encode the language in which the blogger usually posts, and the language of the blogger's social network.

Run		Dutch	English	French	German	Spanish	Overall
Microblog		90.2%	94.8%	90.0%	95.8%	91.2%	92.4%
Blogger	(0.4)	**95.2%**	**98.6%**	**95.4%**	**98.6%**	**96.0%**	**96.8%**‡
Link	(0.2)	90.2%	95.4%	90.6%	96.2%	91.8%	92.8%
Mention	(0.3)	91.6%	96.0%	90.8%	96.6%	93.0%	93.6%
Tag	(0.2)	90.4%	95.2%	90.4%	96.0%	91.4%	92.7%
Conv.	(0.3)	90.8%	95.4%	90.6%	96.2%	92.2%	93.0%

Table 4.4: Results for content-based identification and five individual semi-supervised priors using the microblog language model. The weights assigned to each prior are shown in brackets, and learnt on the development set. We test for significant differences against the baseline microblog model.

Post-independent In our third experiment we tackle research question four. Here, we look at the effect on performance after we combine individual priors in a post-

Run	Dutch	English	French	German	Spanish	Overall
Blogger (0.4)	95.2%	98.6%	**95.4%**	98.6%	96.0%	96.8%
Linear int.	96.0%	99.0%	**95.4%**	**98.8%**	**96.8%**	97.2% ‡
Majority vote	94.4%	96.4%	94.2%	97.2%	**96.8%**	95.8% †
Beam conf.	**97.6%**	**99.4%**	95.2%	98.6%	96.2%	**97.4%**‡
Lead conf.	96.0%	99.2%	90.6%	97.8%	94.4%	95.6% †

Table 4.5: Results for content-based identification runs using *post-independent* (§4.3.1; lines 3 and 4) and *post-dependent* (§4.3.2; lines 5 and 6) combination of the priors and the microblog language model. We test for significant differences against the microblog + blogger model.

independent way. We learn the weights as explained before and find that the content-based identification vector (0.4), blogger prior (0.3), link prior (0.1), and mention prior (0.2) contribute to the best performing setting. Table 4.5 (top) shows that combining the priors results in better accuracy than using them individually. In particular, performance peaks when we make use of fixed weights in the linear interpolation. Inspection of the results reveals that most errors in the voting method are due to ties, which, according to the results, are not always handled appropriately by our method.

Post-dependent In our last experiment, we turn to our last research question, namely, the effect of post-dependent combination of priors and the use of different confidence scores of priors. Before testing, we explore the beam function and width for the beam confidence. Experiments on the development set show a clear preference for the exponential function (95.4% vs. 91.0% accuracy using a 10% beam). As to the beam width b, we look at values of 1%, 5%, 10%, and 15% using the exponential function. Here, the difference is not as big, but we find that 5% is most favorable (97.8% vs. 97.6% for 1% beam and 95.4% for 10% beam). Results in Table 4.5 (bottom) show that post-dependent combination outperforms the use of individual priors and is marginally better than post-independent combinations.

Turning to accuracy for individual languages, we see that language identification works best for English and German, followed by Dutch, French and Spanish with performance hovering at the same levels. In the next section we briefly touch on this with some examples of errors made in the identification process.

Language		Content of microblog post
Assessed	Classified	
Fluent multilingual posts		
French	English	RT @msolveig: Sinusite de printemps, pause pour le moment... V.I.P. reporté, qqs jours de repos et je serai sur pieds. Sorry... Good luck!!!
Spanish	English	RT @FlamencoExport: Espana no solo es flamenco. Tambien es jamon! RT @Plateofjamon Nice article about Iberian ham: http://nyti.ms/6QVF9I ...
Posts containing named entities		
French	English	Vous insultez Ashley de pouf ,de pétasse et autre ... mais vous vous êtes vu bande de connasse ? #JeMenerve
Spanish	English	Pues yo slo quiero que venga Panic! At The Disco. Con eso me conformo.
Prior effects		
French	English	EPISODE N^o 2 : DANS LA LAGUNE...: http://bit.ly/bhi4FG #buzz
Spanish	English	@mariaam1004 *-* Graciaaas! Mi tweet #4777 va para tí (:
Language ambiguous posts		
French	English	#emploi #technicien TECHNICIEN(NE) BE ELECTRIQUE http://is.gd/bnx8A
Dutch	English	@Chenny83 Ja :D

Table 4.6: Examples of misclassified tweets, along with the languages assigned, broken down by error type.

4.4.3 Error Analysis

In analysing the posts misclassified by our final classifier using all priors, we group them into four distinct categories: fluent multilingual posts, those containing named entities, prior effects, and language ambiguous. We give examples in Table 4.6, and explain each type of error in turn.

Fluent multilingual posts: these are posts which are a grammatical sentence with words written in two or more languages. Usually these take the form of a sentence split into two, with both halves in different languages.

Named entity errors: these posts are misclassified because they contain a reference to a foreign language named entity, such as a company or product name, song title, etc. The named entities contained in the post outweigh the correct language tokens in the post in scoring, leading to the misclassification.

Prior effects: the use of priors can sometimes have a negative effect. For example, if the user mentioned a post in a different language to their own post, or when a tag is used mostly from a different language group. E.g., some tweets contain links which point to a webpage in a different language to that used in the post.

Language ambiguous: these posts are misclassified because they only contain a few tokens which could belong to a number of different languages.

Finally, we demonstrate in Table 4.7 for each true language the number of tweets that were incorrectly assigned another language for the post-dependent beam microblog model. In the final row we show the total counts for each misclassified language. English is the most incorrectly assigned label by far, with 54 out of 65, or 83%, of misclassified tweets being assigned an English label. French, as demonstrated in Table 4.5, has the most misclassified posts.

	Dutch	English	French	German	Spanish	Total
Spanish	1	17	0	1	-	19
German	0	7	0	-	0	7
French	1	21	-	0	2	24
English	1	-	0	0	2	3
Dutch	-	9	1	2	0	12
Total	3	54	1	3	4	65

Table 4.7: Misclassification breakdown by language. The leftmost column represents the correct language, and numbers indicate the number of posts classified as another language. Finally in the rightmost column we show the total number of misclassified posts per language.

4.4.4 Discussion

We discuss how the weights of individual priors affect performance, the robustness of our methods when domain-specific training is unavailable, and finally candidate priors unexplored in this chapter for methodological reasons.

Individual Prior Weights

In order to better understand the effects of individual priors when combined with the content-based identifier, we vary their weights from not using the prior at all (0), to using almost only the prior and not the content (0.9). Figure 4.3 shows that for prior weights around 0.4 priors are most helpful. Blogger, mention, and conversation priors are robust to the weights, whilst link and tag show a drop in performance when they are weighted more than 0.4.

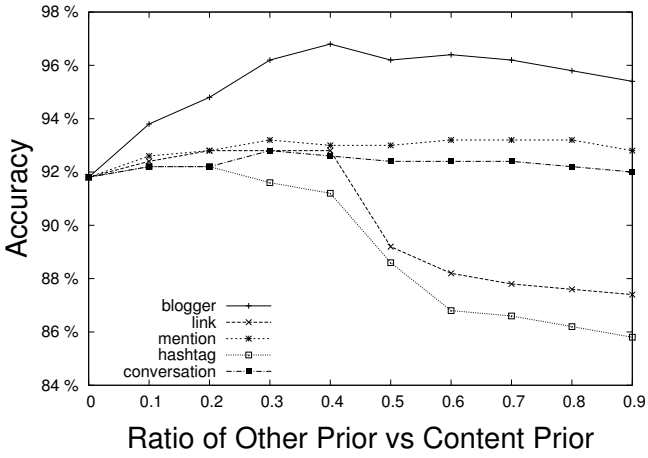


Figure 4.3: Accuracy while changing the ratio of individual priors and content-based prior.

Domain Nonspecific Training

As shown earlier in Table 4.3, training on microblog posts clearly outperforms the use of out-of-the-box models supplied with TextCat. However it may not always be possible to acquire the microblog posts for training, especially if applying the language identifier to many languages. To examine the improvements possible when using out-of-the-box models (or data from domains other than microblogs), we show in Table 4.8 results using priors trained on these models.

The best results using a single prior are achieved using the blogger prior, giving 5% improvement in overall classification accuracy over a domain generic baseline. Again, the combinations of priors show best overall accuracy, with the linear interpolation (post-independent) and the beam confidence (post-dependent) resulting in a 6.5% in-

crease. Interestingly, the best reported accuracies using out-of-the-box models are only about 1.5% lower than best reported microblog models, indicating that, if it is not possible to acquire microblog posts for training, using normal text with the priors defined in this chapter can still lead to high classification results.

Run	Dutch	English	French	German	Spanish	Overall
Out-of-the-box	90.2%	88.4%	86.2%	94.6%	88.0%	89.5%
Blogger (0.4)	95.6%	95.8%	91.4%	98.6%	92.0%	94.7% ‡
Link (0.2)	90.0%	88.8%	86.4%	95.0%	87.4%	89.5%
Mention (0.3)	92.0%	90.6%	87.0%	95.0%	89.8%	90.9%
Tag (0.2)	90.2%	89.0%	85.6%	95.0%	87.8%	89.5%
Conv. (0.3)	91.4%	89.0%	86.6%	95.0%	89.2%	90.2%
Linear int.	96.4%	96.6%	**91.8%**	**98.8%**	93.2%	95.4% ‡
Majority vote	95.0%	**98.0%**	89.2%	97.4%	93.2%	94.6% ‡
Beam conf.	**97.0%**	97.8%	**91.8%**	98.2%	**94.8%**	**95.9%** ‡
Lead conf.	94.0%	97.8%	86.0%	96.6%	90.8%	93.0% †

Table 4.8: Results and significance levels for content-based identification, five individual semi-supervised priors, and their combinations using the TextCat language model: *blogger*, *link*, *mention*, *tag*, *conversation*.

4.5 Online Twitter Analysis

Usage of Twitter is not just limited to the English-speaking world. Other countries, like Indonesia, Brazil, Germany, and the Netherlands actively participate on Twitter, and contribute to a large degree to what is discussed in the microblogosphere. However the distributional profiles of language use on Twitter remains unknown, and thus, alongside the work published in [80, 153, 174] we provide the one of the first analyses of language use on Twitter.

4.5.1 Twitter Language Distribution

We apply our language identification method to a corpus of 1.1 million tweets, collected during the period of one day (2nd of March 2011). These tweets are collected from the GardenHose Streaming API service provided by Twitter, which, at the time, represented a random sample of 10% of the public posts on Twitter. For the languages that fall outside of our original five languages, we use the language models distributed

with TextCat. In Table 4.9 we provide the feature statistics of this corpus over all languages.

	Link	Tag	Reply
Number of tweets	204,127	141,457	621,122
Total number	205,624	191,625	819,553

Table 4.9: Number of tweets with at least one link, tag, or reply and the total number of these items in the set of 1.1 million tweets.

In Figure 4.4, we present the ranked distribution of post languages with counts over 1,000. English ranks highest, with Japanese and Spanish following in second and third. Together, they make up approximately 63% of corpus. The top five languages make up 82% of all tweets in our corpus, and the top 10 languages make up 92%.

The presence of Esperanto and Latin posts is surprising. A manual evaluation confirms these can be accounted for due to classification error.[4] The approximately 1,000 tweets classified as Latin and Esperanto represent only a small portion of the entire corpus (0.009%). The findings published in other work [80, 153, 174] independently confirm the validity of the reported results in this work with respect to the top languages used on the Twitter microblogging platform.

Having a large corpus of labeled microblog posts, we now turn our attention to answering the following analysis questions:

I Does language use alter with time of day?

II To what extent do the classified languages correlate with the geo-location and language metadata fields that come with the Twitter stream?

III How does usage of Twitter features (used as priors in this work) change with language?

4.5.2 Time Series Analysis

Examining the corpus of 1.1 million tweets, we do not know the true underlying distribution. A manual evaluation of all 69 languages classified in the corpus is not possible by the author. However, we believe it would be interesting to examine the language use of bloggers with time. In particular, we expect to see differences in language use of the

[4]Note we do not claim that our language identification classification system achieves 100% accuracy, and thus the inclusion or absence of certain languages could be a result of incorrect labeling.

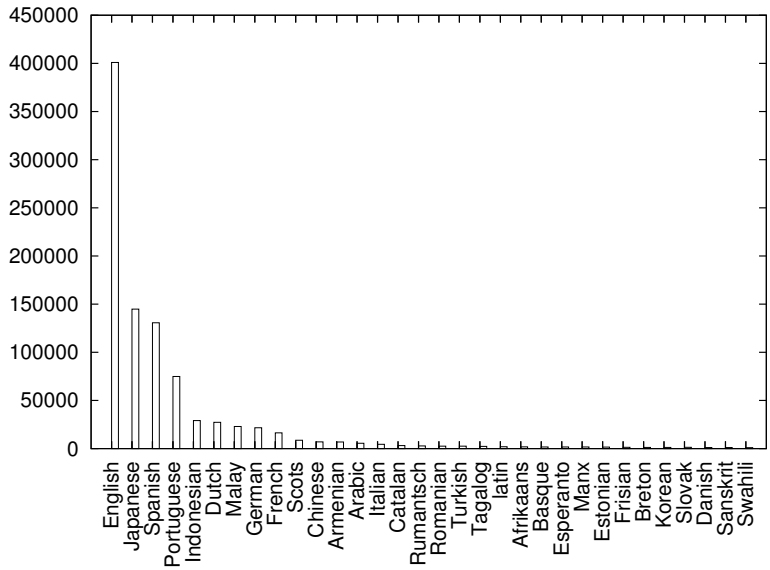


Figure 4.4: Number of tweets per language (published during one day, showing languages with $>$ 1,000 posts).

top five languages classified according to different time zones. Using the 'created_at' time field within the metadata, we bucket each post by their publication hour. Hours are based on GMT +0000. We present the results in Figure 4.5.

We can clearly see two groups of languages according to their distribution over time: (i) English, Spanish, and Portuguese and (ii) Japanese and Indonesian. The former group of languages has its largest speaking population in the Americas, including the USA (English), Brazil (Portuguese), and the other South American countries (Spanish). The latter group is mainly focused around Japan and Indonesia. The differences in time zones between the countries in the two groups explain the differences in peak and dip times: The Asian languages peak around 1pm GMT and reach their lowest dips around 8pm GTM. For the other group of languages we find the peaks between 11pm and 3am, and their dips are found at 7–9 am GMT.

Converting the GMT times to the actual times of the main contributing countries

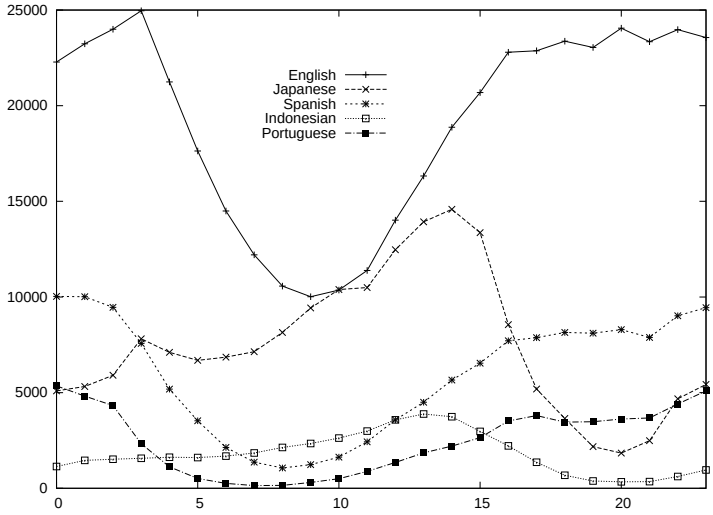


Figure 4.5: Number of tweets in each language published on Twitter in hourly buckets (hours 0−23), for the top five languages classified.

for each language group, we find that for both group the peaks appear between 10pm and midnight and the dips are in the early morning (4–5 am).

4.5.3 Metadata

Obvious priors to explore when creating a language identifier for microblog platforms are the metadata fields that could hint towards the language used by the blogger. Twitter offers two such fields, geo-location and interface language. The geo-location prior was left unexplored in Section 4.4.2 for methodological reasons: in order to collect tweets for a language for our experiments, we used the location to filter tweets for that language. Using location as a prior would bias the results. We also ignored the interface language field, as it is limited to seven languages (English, French, German, Italian, Japanese, Korean, Spanish). Having classified a large corpus of tweets, it is interesting, though, to see to what extent these metadata fields correlate with the languages

assigned.

Interface language field. In Table 4.10 we present the distribution of languages according to the interface language metadata field, along with the number of tweets assigned to each of the seven languages according to our own classifier. Interestingly, of the seven language options offered by Twitter, our classifier and the language metadata field only agree on Japanese, with a difference of 5,000 tweets. For English, we find that almost two times more tweets come from the English interface than our classifier assigns to English (840,000 vs 460,000). We observe similar patterns for German and Korean, while the effect is reversed for French, Spanish, and (less so) for Italian. These figures, along with the fact that there are many more languages used in microblog posts than are offered as interface language options (including Portuguese and Indonesian), indicate the poor suitability of the interface language field in itself as a predictor of microblog post language.

	English	French	German	Italian	Japanese	Korean	Spanish
Metadata	839,856	8,150	6,450	3,348	185,360	6,657	101,728
Classified	459,318	42,706	4,890	4,890	180,140	1,077	142,401

Table 4.10: Tweets per language according to the language metadata field and our classifier.

Geo-location field. We now turn our attention to the analysis of the geo-location information. In particular, Twitter automatically encodes the longitude and latitude points into country information. In total, only 17,737 of the 1.1 million tweets, or 1.6%, contain geo-location information, with 34 countries presented in total. The top countries according to this field are Brazil (6,527 tweets), USA (4,616), Indonesia (2,080), the UK (1,164), and the Netherlands (500). Due to the sparsity in use of the geo-location information, we posit there is a limited utility for the geo-location field for language identification within our framework.

4.5.4 Twitter Feature Usage

We are interested in the way people use Twitter in different languages, and would like to see if there are obvious differences between languages in the usage of Twitter features. For this, we look at three Twitter specific features, *hashtags*, *links* and *mentions*, and explore their usage in the top five languages classified.

In Table 4.11 we report on the percentage of tweets that contain a link for each language, the percentage of tweets having at least one hashtag, the average number of

	English	Japanese	Spanish	Portuguese	Indonesian
Mentioned tweets	54.8%	48.2%	61.6%	44.8%	76.6%
Avg. mentions per tweet	1.3	1.2	1.3	1.2	1.8
Tagged tweets	16.6%	4.17%	13.8%	10.8%	9.8%
Avg. tags per tweet	1.4	1.2	1.2	1.5	1.1
Linked tweets	26.1%	10.7%	14.4%	10.0%	12.4%

Table 4.11: Twitter feature usage per language, for the top five languages

hashtags per *tagged* tweet, the percentage of tweets that contain at least one mention and finally the average number of mentions in tweets that have mention.

We see that Indonesian and Spanish show high mention usage, with over three quarters of Indonesian tweets containing at least one mention. On average, they contain 1.8 mentions, indicating the popularity of this feature for Indonesian microbloggers to interact with multiple other microbloggers.

The proportion of tweets containing a tag or link is far lower across all languages than those containing mentions. English and Spanish have the highest percentage of tweets containing a hashtag. Though only 10.8% of Portuguese tweets contain a hashtag, when they do, they have the highest average tags per tagged tweet rate, indicating that when they do use tags, they tend to use multiple. Finally, English displays the highest proportional use of links, with just over 25% containing a link, 10% more than Spanish posts at 14.4%.

4.6 Conclusion

In this chapter we studied language identification on microblog posts. We demonstrated that, given the short nature of the posts, the rather idiomatic language in these (due to abbreviations, spelling variants, etc.), and mixed language usage, language identification is a difficult task.

Our approach is based on a character n-gram distance metric. To tackle the challenges in microblogs, we identified five microblog characteristics that can help in language identification: the language profile of the blogger (blogger), the content of an attached hyperlink (link), the language profile of other users mentioned (mention) in the post, the language profile of a tag (tag), and the language of the original post (conversation), if the post we examine is a reply. Further, we looked at methods on how to combine these priors in a *post-dependent* and *post-independent* way.

Results show that the use of language models trained on microblog posts increase

accuracy by 3%. Individual priors add to performance, with the blogger prior adding another 5%. The combination of priors is found to outperform their individual use, with post-dependent combination leading to the best performance, close to that of formal short texts. A manual analysis reveals four main categories of errors: fluent multilingual posts, prior effects, named entity errors, and language ambiguity.

We also conducted a large scale study of language distribution on a popular, global, microblogging platform. We demonstrated that the language and country metadata fields that come with the microblog posts make poor signals for language identification, with the language field greatly over-or-underestimating the true underlying language distribution, and the geo-location field being too sparsely used to be relied upon for language identification. Finally, we have demonstrated the differing use of Twitter specific features per language.

We now revisit the research questions formulated earlier in the chapter.

RQ 1. Can we successfully identify the language of a microblog post, given the short, idiomatic and unedited nature exhibited?

Using extensive experiments with a plethora of models, we found we were able to achieve significant improvements in classification accuracy over the baseline. In particular, we found that the blogger prior was the single most important source of additional content in aiding classification accuracy, though greater improvements could be made using all available additional information sources. In our restricted experimental setting where we had microblog posts in five distinct language to classify, we achieved the highest average classification rates of 97.4% over all languages. While this is a restricted experimental setting with only five languages, we argue that the relative, significant differences between our model and the baseline would hold in a framework with more languages, thought the actual raw scores themselves would decrease.

We now turn our attention to the five specific subquestions.

RQ 1a. What is the performance of a strong language identification method for microblogs posts?

RQ 1b. Does domain-specific training of language models help improve identification accuracy?

We presented experiments demonstrating the fact that the idiomatic nature of microblog posts led to decreased classification results of 7% overall in comparison to more formal texts of the same size. This indicated that the nature of microblog post language was in itself a problem. By training our classifiers on the microblog data, we saw improvements of 2.9% accuracy overall, indicating that domain-specific training does help improve classification accuracy. However, even with domain-specific training data, an

overall classification accuracy of 92.4% indicates that just under 1 out of every 10 microblog posts will be misclassified, and considering the sheer volume of posts, this is an un-acceptably high number.

Our intention in turning to the next subquestions was to propose algorithms that specifically increased the overall classification accuracy giving a method with acceptable classification accuracy in a real world setting.

RQ 1c. What is the effect on accuracy of using priors extracted from microblog characteristics?

RQ 1d. Can we successfully combine semi-supervised priors in a post-independent way?

RQ 1e. How can we determine confidence of individual priors, and can we use confidence to combine priors in a post-dependent way?

We found that different priors give varying amounts of improvements above the baseline, with the blogger prior giving the best single improvements, though the use of all priors lead to the best improvements. Experiments demonstrated that assigning confidence scores to each of the priors, and then using these to determine a post-dependent prior weighting scheme gave the best results, though this result is not significantly better than the post-independent method where weights are optimised on a development set.

In the next chapter, we evaluate different strategies for adapting an out-of-domain translation system for the task of translating microblog posts. We examine the use of retrieval based methods for adapting an existing phrase table, and compare this to the simpler self-learning approach. Having then examined how we can explore microblog posts for additional training material, we turn to the second research theme of this thesis and examine how we can exploit existing data sets for improved translation quality.

Chapter 5

Exploring Twitter for Microblog Post Translation

Microblogging platforms serve as a way of disseminating information quickly [179], and as a by-product of microblogging platforms, internationally. There is a natural desire to understand what is being posted in different languages, whether it be posts on news-events, opinions about world leaders, or product reviews. However, many users post in foreign languages (in Section 4.5, we demonstrated that in our sample of 1.1 million tweets posted in a day, 700,000, or approximately 63%, of these were classified as being non-English[1]), so we need to use machine translation to understand this content. Unfortunately existing systems, trained on freely available bi-texts such as the Europarl corpus (see Section 3.4), perform badly when applied to texts of a different domain and or genre [76]. There are no publicly available, microblog specific, bitexts that can be used as in-domain parallel corpora to train our SMT system on. So in this chapter, we examine the task of in-domain bi-text acquisition. Specifically, we ask in this chapter how we can automatically mine multilingual tweets for new translations, so we can better translate microblog posts. The main research question we ask is:

RQ 2. Given a set of language identified microblog posts, how can we learn new phrase translations from the data?

This general research question gives rise to the following specific subquestions:

[1] Also see [174].

RQ 2a. Do information retrieval based approaches outperform simpler methods based on self learning?

The automated translation of microblog posts faces two main challenges. First, there are no publicly available microblog corpora usable for evaluation and optimisation purposes. Without this data, the scientific examination and quantifiable analysis of any proposed methods is not possible. Second, the idiomatic, unedited and short nature of microblog posts makes the adaptation of existing, more formal parallel corpora a challenge.

We propose to exploit the fact that posts in different languages will often be on the same topics, e.g, related to the same national or world-wide event; cultural, natural, sports or political related. The central idea is to indiscriminately crawl posts from the same period, and examine the use of either self-learning or retrieval techniques for learning new translations. In examining the use of both approaches, there are many smaller questions that need to be answered. These include:

RQ 2b. Which ways of combining the in-domain data with the baseline out-of-domain data work best? How much data from the microblog post corpus can we exploit? How important is the common filtering step for the information retrieval method?

To answer these questions, we need an in-domain evaluation corpus, for which there are none available. So, for optimisation and evaluation purposes, we use a hand-selected group of bilingual, non-professional translators to translate English tweets into their native language, Dutch. Thus, as well as answering the previous questions, the creation of a corpus of translated tweets also forms part of the contributions of this chapter.

This chapter is structured as follows. In Section 5.1 we briefly discuss the related literature. In Section 5.2 we describe the two approaches for mining parallel corpora for domain-adaptation examined in this chapter. In Section 5.3 we describe the initial out-of-domain we use, the in-domain data data we have at our disposal, and how we set about an evaluation set we could use for development and testing purposes. We then present our experimental results in Section 5.4, and draw conclusions in Section 5.5.

5.1 Related Work

The seminal work on extracting parallel sentences from comparable data, especially newswire documents, was conducted in [129]. The work assumes that the nature of the newswire source enforces a level of parallelism at the document level between the two languages; the task is in retrieving parallel documents for a given source document automatically, and then doing parallel sentence selection over the set of all potential

aligned sentence pairs. To find the set of parallel documents, the source document is translated and fed to an index containing documents in the target language. The returned documents are restricted to appear within a time window of the source document, and the target documents ranked using standard Information Retrieval (IR) retrieval metrics, such as TF-IDF [88, 161, 162], and are finally then heuristically pruned such that only the top-n are kept. Sentences from the documents are then extracted, filtered according to a word-overlap metric, and then scored with a maximum entropy classifier with length, word overlap, and alignment based features.

The first step in the approach of Munteanu and Marcu [129] is often considered a prerequisite to reduce the space of potential aligned sentence pairs. However, recently an extension has been proposed to ignore any explicit document level information, and to do a search for translation candidates directly on a large set of sentences [186, 187] (tens or hundreds of thousands of sentences, instead of the hundreds and thousands as in [129]). Tillman [186] use a beam search algorithm, similar to the beam search used in translation systems as described in Section 2.3, to find likely candidates. Given the exhaustive nature of the search, the algorithm takes four times as long to run in comparison to the approach presented in [129], at the benefit of improved BLEU scores. Note that they limit the space off all sentences over which they search to a seven day window, which still restricts the size of the candidate sentence set.

Other extensions to [129] include applying simple heuristics to deal with the case where their is additional information in one sentence, but not in the other [1], or by extracting just phrases from non-parallel data [130], or by explicitly modelling the document level alignment in the wikipedia domain [180].

An alternative, simpler approach, is to directly use the translations output of the SMT system as translation candidates, instead of using them as queries to find candidate document translations. This approach is often referred to as the self-learning approach [2, 83, 171, 172]. This approach has the added benefit that you can get improved performance without the need for bilingual corpora, however this approach is not able to learn new translations; the set of all translation candidates come from the translation dictionary or phrase table used to create them. Improvements come from learning different word-ordering, and being able to learn translations for longer source phrases. An extension that we do not examine reverses the standard self-learning approach by translating the target tweets into the source language [111] . The intuition is that any mistakes in the reordering of the source-side translations are acceptable as they will never match with any source test sentences.

In this work, we build off the framework of [129]. Our work is similar in that we examine the use of a cross-lingual TF-IDF retrieval based method for finding parallel documents. However they work with a source in which the production of documents guarantees a specific (though unquantifiable) level of parallelism between documents.

In examining microblog posts, we operate over a non-parallel document space. In such cases, the application of phrase-based methods as in [130] would prove an appropriate extension of the work presented in this chapter. This is something we leave to future work.

The only other work that we are aware of that examines the problem of translating tweets is that of [85]. They work in the Arabic-to-English translation setting. They focus on tweets which are on or related to the topic of the "Arab Spring," and so create their monolingual corpora on which they train, develop and test their system on by retrieving tweets using a keyword-based crawling. The keywords they use are all human picked proper-nouns related to the Arabic spring event. The work presented in this chapter differs in that we make no topic based constraint. That is, we use a non-topic specific, non-keyword crawl of Twitter which involves pulling tweets from the public Twitter API.[2] While both approaches are equally valid and do not violate the methodological principle of keeping the development and test sets unseen at the training stage, the approach of Jehl et al. [85] is implicitly, at the topic level, a transductive approach, because while they do not actually look at the development and test tweets, they will share a topical similarity with the training data. Our approach differs in that is, possibly detrimentally, topic independent. Further, though not tweets, the task of translating noisy data has also been in examined in the Haitian-Creole emergency SMS message translation task, part of the 2011 Workshop on Statistical Machine Translation [28]. SMS messages give rise to many similar problems to tweets; they are restricted in length, and so contain short, abbreviated, and noisy language.

In this work, we also assume that we work in an online setting where we do not know the test data to be translated at training time. There has been work however that has examined the use of transductive semi-supervised methods for adapting an SMT system to that of either a specific test sentence or test-corpus. The aim of such transductive methods is to use the test input, though not the test reference translations, for creating a better system [195]. A different approach to dealing with mismatches in training and test set domains is to separate or classify the training data into different domains/genres, and then apply the appropriate, in-domain models, with an incoming test sentence [12].

To acquire our manually annotated development and test Twitter corpus, we used on-site bilingual human annotators. However, where locally resourced bilingual annotators are not available for a specific language pair, using crowd-sourcing is a popular alternative. Previous literature has demonstrated the methodology in using online crowd-sourcing resources for doing this in such a way as to generate good quality translations at a reasonable cost far below that of professional translators [85, 209, 210].

[2] `https://dev/Twitter.com/docs/streaming-apis`

Alternative to paid crowd-sourcing, an approach includes expanding on the annotating as a game framework [199] and casting the annotation problem within a language learning setting `http://duolingo.com/`.

5.2 Methods

We now present the approaches we use to adapt our baseline data (which will be explained in Section 5.3) for translating tweets. There are two main approaches we examine in this work; the first based on retrieval methods, described in Section 5.2.1, and the other based on self-learning, which we describe in more detail in Section 5.2.2.

5.2.1 Retrieval-based Methods

In the first approach we examine, we look at the use of information retrieval (IR) methods for finding suitable translation candidates of foreign tweets. Here, instead of using the translations output by an out-of-domain SMT system directly, we use these translations as queries that we send to an index containing the target tweets. An index is an efficient structure where documents are stored in an inverted table, so that given a query made up of terms, documents containing these terms can be returned quickly.

How documents are ranked is important for finding potential translation candidates. Ideally, the more query terms that appear in returned documents, or tweet, the higher it should appear in the ranked list. In this chapter, we use a common ranking method; the vector space model [168].

In the vector space model, documents and queries are represented by vectors, where each element or dimension corresponds to a separate term in the vocabulary space V. The value of each element can for example be the raw count of the term in that document. However a common scheme used is term frequency-inverse document frequency (TF–IDF) approach. TF–IDF defines the weight for each term t in a document d as:

$$w_{t,d} = tf_{t,d} \cdot log \frac{|D|}{|\{d' \in D | t \in d'\}|}, \tag{5.1}$$

where $tf_{t,d}$ is the term frequency of t in d, and the second factor is the inverse document frequency (IDF), where $|D|$ is the total number of documents, and $|\{d^{'} \in D | t \in d^{'}\}|$ is the number of documents containing the term t. IDF used along with the raw term frequency counts, to negate the impact of terms that appear not just frequently in the document but are also frequent in the entire document corpus. The score then for a document d given a query q is then the cosine similarity between the document and query vectors;

$$score(d, q) = \frac{\Sigma_{i=1}^{V} w_{i,d} w_{i,q}}{\sqrt{\Sigma_{i=1}^{V} w_{i,d}^2} \sqrt{\Sigma_{i=1}^{V} w_{i,q}^2}}. \quad (5.2)$$

We used PyLucence,[3] a python wrapper over the core Java based Lucene index.[4] For speed of retrieval, PyLucene implements a boolean search first, where documents returned must contain at least one matching term with the query. Over this set, the cosine similarity metric using TF–IDF based term-vectors is then applied to this subset to attain a ranked list of target tweets.

5.2.2 Self-Learning

In the self-learning approach, the abundance of the source-side resources, parts of which may overlap with the tweets to be translated, are exploited for domain adaptation purposes. Specifically, the tweets for which we have no human translations, are translated by an SMT system built on out-of-domain data. The output translations, along with the original source tweets, are then used as new additional pseudo-parallel data.

In the related work, we mentioned that self-learning, or lightly-supervised, approaches could not learn new translations for unseen, or OOV tokens. In our setting this is not accurate. As we apply the self-learning approach, we leave unseen tokens in the source sentence to be translated. This means that when we rerun word alignment (we describe exactly how we do this in Section 5.2.3), translations for OOV tokens, can be learnt if the OOV token is aligned with a word in the translation.

In Figure 5.1, we show the overview of both approaches to mining parallel corpora for additional bitext, placing them both in a common framework. In the self-learning approach, the translations are directly used as pseudo-parallel corpora; in the retrieval method, the translations are fed to an index containing target tweets, and output from the index is used as the the target side of the pseudo-parallel corpus.

In the self-learning setting, we use the single best translation output by the SMT system. We could use the top-n output, but outputting an n-best list considerably slows the translation process. In the IR setting, we use the top 3 returned documents, which are ranked according to Equation 5.2.

[3] `http://lucene.apache.org/pylucene/`
[4] `http://lucene.apache.org/core/`

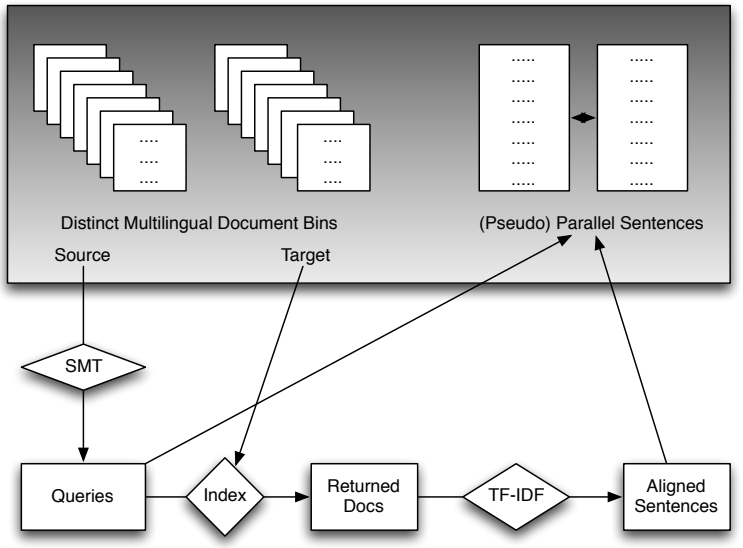


Figure 5.1: The generic framework in which the self-learning and the IR approaches are outlined.

5.2.3 Building In-Domain Phrase Tables

There are different ways to use the additional parallel-sentences from either the self-learning or retrieval based methods. At the word alignment step, we can bootstrap from the original alignments, or completely start word alignment again from the beginning. We could keep the original out-of-domain, but highly parallel data, and the new in-domain, but noisy, pseudo parallel data, separate, giving rise to two individual phrase tables. Otherwise we can concatenate the files into one big file, rerun word alignment from the beginning, and use the resulting single phrase table file. In this work, we examine two different approaches for using the in-domain data with the original out-of-domain data. The first based on concatenation, which we will refer to in the experimental results as (CO-), and the second, a backoff approach (BO-), where we will only use the in-domain data set for unigrams which are unseen in the out-of-domain parallel corpora.

For the information retrieval based approaches, we can further limit the number of returned target tweets for a given source tweet, and also place a minimum limit on the number of alignments between the source tweet and the returned tweet, either

Language:	English	Spanish	Dutch	French	Total
# tweets:	85,642,254	23,105,008	6,167,870	2,938,305	117,853,437

Table 5.1: Distribution of languages in the tweet corpus used throughout this chapter.

at the word or phrase level. Such alignment based filters have previously been used in [85, 129]. We examine a number of different settings, explained fully in Section 5.4.

5.3 Data

We now describe the data used in this chapter, including the out-of-domain baseline data in Section 5.3.1, the gathering of the in-domain tweets in Section 5.3.2, and the acquiring of a manual translated development and test set in Section 5.3.3.

5.3.1 Out-of-Domain Data

The out-of-domain data we begin with is composed of the English-Dutch portions of the Europarl corpus [100]. After document and sentence alignment, we end up with a parallel corpus consisting of 1,910,597 sentences.

5.3.2 Twitter Crawl

The tweets making up our dataset were extracted using the Twitter API.[5] In the previous chapter, we explicitly evaluated our language identification approach on a single days worth of data. Here, we used all tweets we had gathered in period between the 1st of January 2011 and the 30th of August 2011, giving a total of 117 million tweets. In order to separate the tweets into different languages, we applied the microblog models we learnt in the previous chapter to the 117 million tweets. Table 5.1 shows the language distribution of our microblog collection for the top five languages, along with the number of tweets classified as Dutch. For this study, we kept all the tweets classified as being either English or Dutch, and discarded the rest.

5.3.3 Creating a Human Annotated Twitter Corpus

In order to be able to evaluate the methods proposed in Section 5.2, we need to have a corpus of human translated tweets. To do this, we first split the bilingual tweet corpus

[5] `https://dev/Twitter.com/docs/streaming-apis`

Ann.	BLEU	1	2	3	4
1	00.00	41.38	08.49	03.12	00.00
2	10.79	39.13	13.33	07.37	03.53
3	05.90	43.64	08.00	03.33	01.25
4	04.76	37.38	06.19	02.30	01.30
5	**11.00**	**45.61**	**12.50**	**07.45**	**03.57**
6	07.26	42.98	10.58	05.32	01.19
7	00.00	37.72	06.73	02.13	00.00
8	06.72	43.75	09.80	04.35	01.22
9	05.69	42.48	06.80	03.23	01.20
10	05.65	39.39	08.99	03.80	01.45
11	06.12	40.74	10.20	03.41	01.28
12	06.52	39.09	10.00	04.44	01.25
13	06.28	36.84	08.65	04.26	01.19
14	00.00	33.02	08.33	01.16	00.00

Table 5.2: BLEU scores of the annotator translations of the ten Europarl sentences, per annotator. We show the individual n-gram precision results from 1 to 4 per annotator. The average BLEU score, taking into account the annotators with a zero BLEU score because they did not have a single 4-gram match, is 05.48

into two sets; the first containing tweets between January and the end of July 2011; the second set containing the rest, that is those microblog posts posted in August 2011.

From the tweets posted in August, we randomly selected a portion of tweets to be annotated. Because re-tweets often had a portion of the tweet which was being reposted cut off, due to the 140 character limit, we removed any tweet from the set to be annotated that was a re-tweet. We left mentions, urls and hashtags in the tweets to be translated because they often form part of the natural sequence of the tweet, and the location of their placement in the translation is important. Further, when possible, annotators were advised to translate hashtags into Dutch when a translation existed.

These tweets were then translated by a hand selected group of fourteen bilingual researchers in the authors research institute. Because, for the majority of the annotators, their first language was Dutch, they were asked to translate English tweets into their native tongue. Further, as a check against the quality of their translations, we presented them with ten English Europarl sentences, for which we already have translations.

In Table 5.2 we present the BLEU scores for the translations of the ten Europarl sentences presented to the annotators. The results are low, ranging from BLEU scores

Europarl Source	is that so bad
Europarl Reference	wat is daar zo verkeerd aan
Annotator Translations	
1-2, 6, 8, 10-14	is dat zo erg
3, 9	is dat zo erg ?
4	is dat zo slecht
5	is dat zo moeilijk
7	is dat nou zo erg

Table 5.3: The translations of a single Europarl sentence by the annotators, demonstrating the quality of the translations in comparison to the Europarl provided translation.

of 0 to the highest score of only 11 BLEU points. These scores at first sight look bad. An examination of the individual precision scores for n-grams of between 1 and 4 show that while the unigram precision levels are in the range of 33 to 45.6, the bigram and above precision scores drop considerably.

Because of the low BLEU scores in Table 5.2, we conducted a manual analysis of the translations for these Europarl sentences. We present the different annotations for each annotator for a given sentence in Table 5.3. As can be seen in the table, the translations make sense, more so then the Europarl reference translation, which seems to take into account context not in the source sentence itself, i.e information not available to the annotators.

The example shown in Table 5.3 is representative, with respect to annotator agreement and translation quality, of the other nine Europarl sentences and their annotations. Given the manual, qualitative analysis of the quality control translations, we posit the low BLUE scores between the annotators and the Europarl reference translations are due to two factors. The first is the fact that the translations only have a single reference translation in the Europarl corpus. This means BLEU is unable to take into account the use of synonyms, paraphrases and the general variance existent in creating translations. Second, we believe the translations differ partly due to the additional context that seems to have been available to the Europarl translators, context not available to our annotators.

In Table 5.4, we present the inter-annotator agreement scores, measured in BLEU. To compute these per annotator inter-annotator agreement scores, for each annotator, we create a unique reference set made up of the translations from all the other annotators, and then compute BLEU, as explained in Section 2.4. The lowest inter-annotator

Ann.	BLEU	1	2	3	4	5	6	7	8	9
1	75.29	93.97	83.02	70.83	58.14	47.37	40.30	32.76	22.00	14.29
2	68.96	90.43	75.24	64.21	51.76	41.33	33.33	24.56	18.37	12.20
3	86.84	98.18	93.00	84.44	73.75	58.57	43.33	33.33	20.93	11.43
4	**91.15**	**98.13**	**95.88**	**89.66**	**81.82**	**77.61**	**70.69**	**62.00**	**61.90**	**60.00**
5	86.97	96.49	88.46	84.04	79.76	72.97	63.08	51.79	43.75	37.50
6	87.56	96.49	91.35	86.17	77.38	70.27	64.62	60.71	56.25	52.50
7	73.65	92.11	78.85	68.09	59.52	51.35	43.75	36.36	27.66	17.95
8	77.15	94.64	84.31	72.83	60.98	52.78	41.27	29.63	21.74	15.79
9	83.64	95.58	89.32	80.65	71.08	58.90	49.21	42.59	36.96	34.21
10	72.54	92.93	82.02	69.62	52.17	38.98	28.00	19.05	11.76	07.69
11	71.52	91.67	78.57	65.91	55.13	42.65	33.90	27.45	20.93	14.29
12	77.77	96.36	85.00	74.44	60.00	51.43	42.62	30.19	22.22	16.22
13	72.58	93.86	78.85	67.02	55.95	45.95	35.38	26.79	14.58	07.50
14	64.01	87.74	72.92	56.98	46.05	37.88	31.58	22.92	12.50	09.37

Table 5.4: Inter-annotator agreement rates, broken down per annotator (Ann.), over the ten Europarl sentences. Agreement rates were computed for each user by comparing their translations to those of the other annotators using BLEU. We also display the individual n-gram precisions from 1 to 9 per annotator. The average inter-annotate agreement rate is 77.83.

agreement rate is 64, the highest is 91. The average inter-annotator rate is 77.83. The high agreement rates are encouraging, though are maybe inflated due to the formal nature of the Europarl sentences.

Finally, though we are not able to compute inter-annotate agreement rates on tweet translations, we present in Table 5.5 five source tweets and their translations from different annotators. Again, the translations are encouraging, though differences in word-choice, word-position, and use of grammar does differ between annotators.

5.4 Experiments

We begin by detailing the experimental methodology we use in Section 5.4.1, and presenting experimental results in Section 5.4.2.

Source:	Tired but I'm feeln good today !
Ref 1:	Moe maar ik voel me goed vandaag!
Ref 2:	Moe maar ik voel me vandaag goed!
Source:	Jose's annoying like uhh, I don't think sollll
Ref 1:	Jose is vervelend als uhh, I denk het niet!!!!
Ref 2:	Jose is irritant, met uhm, ik denk het niet
Source:	@EarthToLauur I agree, haha. I wanna see green lantern partly just because he is in it.
Ref 1:	@EarthToLauur ik ben het met je eens, haha. Ik wil de groene lantaarn onder andere zien omdat hij er in zit .
Ref 2:	@EarthToLauur mee eens, ha ha. Ik wil Green Latern mede zien omdat hij erin zit.
Ref 3:	@EarthToLauur Mee eens, haha. Ik wil green lantern zien deels alleen omdat hij erin speelt..
Source:	Thank da lord 4 allowin me 2 see anotha day #TrulyBlessed
Ref 1:	Dank de heer voor het schenken van weer een dag aan mij #TrulyBlessed
Ref 2:	Dank zij god dat hij mij toestaat om nog een dag te zien #TrulyBlessed
Source:	@JimHarris @SharonLabchuk In the name of CEO added sitemap to Green Party PEI http://greenparty.pe.ca/sitemap.xml
Ref 1:	@JimHarris @SharonLabchuk Uit naam van de directeur een sitemap toegevoegd aan Green Party PEI http://greenparty.pe.ca/sitemap.xml
Ref 2:	@JimHarris @SharonLabchuk Namens de CEO een sitemap toegevoegd aan Groene Partij PEI http://greenparty.pe.ca/sitemap.xml

Table 5.5: Example tweets that were translated by two or more annotators.

5.4.1 Methodology

From the entire set of tweets annotated as being English, we randomly picked 1 million of these between the dates of 1st January and the 31st of July, to use as our in-domain, unlabelled, source tweets. On the target side, we indexed the entire side of the Dutch tweets. In the cross lingual approach, we examine two different ways of refining the tweets; the first with no date restriction; the second where returned tweets must have

Method	bitext size
Baseline	1,910,597
Self-Train	2,910,335
TND-TOP1-MIN0	2,902,902
TND-TOP3-MIN0	4,058,942
TND-TOP1-MIN10	2,494,248
TND-TOP3-MIN10	3,661,498
TWD-TOP1-MIN0	2,828,298
TWD-TOP3-MIN0	4,038,188
TWD-TOP1-MIN10	2,497,448
TWD-TOP3-MIN10	3,669,436

Table 5.6: Number of different sentences in the parallel data used for each method. Note that the concatenate and backoff approaches use the same parallel data, so are not mentioned here.

been posted with 12 hours either way of the source tweet. The motivation behind applying a date restriction is that the tweets posted in the same period are more likely to be on the same topic. On the other hand, not having a date restriction increases the chance that returned tweets have a higher number of matching terms.

We split the human annotated tweets into a 500 tweet development set and 1000 tweet test set. For preprocessing the manually annotated tweets, we use two different strategies; one where the hashtags, urls and mentions are left in, referred to as the noisy set, and another where they are removed, referred to the clean set. We do this because, while hashtags, urls and mentions can simply be copied into the translation as OOV items, their placement in the translation is still important for reasons of fluency. However, leaving them in will inflate the BLEU scores; specifically, we expect unigram precision scores will receive a boost.

Our baseline consists of a phrase table built from exclusively out-of-domain Europarl data, as described in Section 5.3.1. For the language model, we use the Dutch side of the Europarl bi-text, and concatenate the in-domain Dutch tweets on to this. We build a 5-gram language model using Kneser-Ney smoothing. All MERT runs for all approaches use the 500 tweet development set, and use the same language model as previously described.

Model	BLEU	BLEU-NBP	1	2	3	4
Baseline - no additional pseudo-parallel data						
B	24.91	24.91	57.63	31.18	18.73	11.45
Self-Learning approaches						
S-CO1	**27.35**‡	**27.35**	**59.88**	**33.84**	**21.00**	**13.21**
S-BO1	24.69	24.79	57.58	31.18	18.65	11.30
Information Retrieval approaches						
TND-CO1-PA0	23.31	23.65	57.06	30.03	17.55	10.41
TND-CO1-PA10	24.47	24.90	58.67	31.35	18.52	11.30
TND-CO3-PA0	23.53	23.87	57.54	30.38	17.71	10.50
TND-CO3-PA10	23.78	23.98	57.57	30.47	17.71	10.66
TND-BO1-PA0	22.51	22.83	56.04	28.81	16.80	10.02
TWD-CO1-PA0	21.96	22.39	56.07	28.68	16.46	9.51
TWD-CO1-PA10	22.91	23.25	56.76	29.59	17.16	10.15
TWD-CO3-PA0	22.78	23.05	56.54	29.45	16.92	10.03
TWD-CO3-PA10	23.16	23.47	57.00	29.68	17.32	10.37
TWD-BO1-PA0	23.52	23.81	57.47	30.16	17.58	10.56

Table 5.7: Results on the clean set, where urls, hashtags and mentions were discarded. Numbers in bold indicate the highest scores for that particular column.

5.4.2 Results

We present experimental results in Tables 5.7 and 5.8, for the noisy and clean test sets respectively. To remind the reader, the clean test set has urls, hashtags and mentions removed from the source and reference translations; the noisy set does not. (-CO) means the additional parallel text was concatenated onto the out-of-domain data, and word alignment was rerun from the beginning. (-BO) indicates we kept the in-domain pseudo-parallel bi-text separate from the out-of-domain bi-text, and ran word alignment just on this new in-domain data, and then use the resulting phrase table in a backoff framework. That is the out-of-domain phrase table is used to translate the test set, and we only use, or backoff, to the in-domain phrase table, for unseen unigrams. The numbers after the affixes (-BO) and (-CO) indicate the number of target sentences a source sentence is paired up with prior to running word alignment. A number of 3

indicates that we used the top-3 returned target microblog posts, and for each source post, created three new pseudo parallel sentences. Finally, for the information retrieval approaches, we examine the impact that limiting the number of phrase-alignments between the source and returned tweet, according to the out-of-domain phrase table. These limits are referred to as (-PA), where the number indicates the threshold. In this work, we examine thresholds of 0, indicating no threshold is used, and 10, meaning there must be 10 unique phrase matches between any n-gram in the source tweet and any n-gram in the returned tweet. As the various different models use different amounts of pseudo parallel data given different settings, we show in Table 5.6 the number of parallel sentences, parts of which are pseudo-parallel, for each of the different models.

As can be seen, the baseline (B) has a score of 24.91, or 37.08 on the noisy set. By using the top translation of the million tweets as additional parallel, or pseudo-parallel data (S-CO1), we see significant improvements of 2.44 and 2.19 BLEU points on the clean and noisy sets. These improvements are significant at the $p < 0.001$. These improvements do not hold if we use the same data in backoff setting (S-BO1), where results are now below the baseline (24.91 vs 24.69 and 37.08 vs 36.90).

Interestingly, S-CO1 was the only model to outperform the baseline. The information retrieval based methods (TND) and (TWD), even though they are able to learn from in-domain target microblog posts, all performed worse then the baseline, decreasing translation quality on both sets; between 2.87–1.38 BLEU points on the clean set, and 3.47–2.15 BLEU points on the noisy set.

5.4.3 Error Analysis

We now examine in greater detail exactly why the self-learning approach using concatenated bitexts not only significantly outperformed the baseline, but all other approaches.

OOV tokens. We examine the OOV rates of the four different approaches. We present the OOV percentages for both the clean and noisy sets in Table 5.9. Instead of just presenting the unigram out-of-vocabulary, we also show the bigram, trigram and 4-gram OOV percentages against the respective test sets. The self-learning approaches (*S-CO1* and *S-BO1*) have the lowest OOV rates.

Even though the best performing method (*S-CO1*), according to BLEU, displays the lowest OOV rates, this metric can not explain the difference in results between the different models; the *S-BO1* approach displays similar OOV rates, but performs significantly worse then *S-CO1*. Given the gap in results across data-sets of these two approaches, even though they use the exact same data sets, though in different ways,

Model	BLEU	BLEU-NBP	1	2	3	4
Baseline - no additional pseudo-parallel data						
B	37.08	37.08	65.10	42.18	30.13	22.85
Self-Learning approaches						
S-CO1	**39.27**‡	**39.27**	**66.97**	**44.49**	**32.32**	**24.71**
S-BO	36.90	37.02	65.01	42.09	30.05	22.85
Information Retrieval approaches						
TND-CO1-PA0	34.89	35.47	64.05	40.71	28.49	21.32
TND-CO1-PA10	35.90	36.59	65.55	41.81	29.41	22.25
TND-CO3-PA0	34.93	35.56	64.33	40.75	28.49	21.40
TND-CO3-PA10	35.13	35.59	64.29	40.79	28.52	21.46
TND-BO1-PA0	34.40	34.81	63.43	39.65	27.81	21.03
TWD-CO1-PA0	33.61	34.38	63.34	39.49	27.39	20.40
TWD-CO1-PA10	34.85	35.43	64.41	40.60	28.30	21.30
TWD-CO3-PA0	34.18	34.69	63.46	39.86	27.63	20.73
TWD-CO3-PA10	34.36	34.85	63.62	39.91	27.83	20.89
TWD-BO1-PA0	34.41	34.62	62.87	39.65	27.67	20.84

Table 5.8: Results on the noisy set, where urls, hashtags and mentions were kept. Numbers in bold indicate the highest scores for that particular column.

we now proceed to examine the impact that splitting the in-domain and out-of-domain data, versus concatenation into a single bi-text, has on word alignment.

Word alignments To remind the reader, in the backoff approach, word alignment for the additional pseudo-parallel bitext is run independently from the out-of-domain data. In the concatenation approach, the additional in-domain parallel data is appended to the out-of-domain data. Because the backoff approach does not have the clean, though out-of-domain, bitext to help in the search for the viterbi best alignments, we hypothesise that splitting the in-domain and out-of-domain bitexts could lead to the learning of noiser word alignments.

To take an example, we show in Figure 5.2 the symmetric (*grow-diag-final*) alignments of the split approach (left) and concatenation approach (right) for the source

Set	Clean				Noisy			
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**
B	30%	56%	84%	96%	46%	60%	85%	97%
S-CO1	**5%**	**21%**	**52%**	**80%**	**22%**	**27%**	**56%**	**81%**
S-BO1	**5%**	22%	53%	**80%**	**22%**	28%	57%	82%
TND-CO1-PA0	16%	54%	84%	96%	33%	58%	85%	97%
TND-CO1-PA10	13%	41%	75%	93%	31%	47%	77%	94%
TND-CO3-PA0	11%	36%	71%	91%	29%	42%	73%	92%
TND-CO3-PA10	11%	37%	71%	91%	29%	43%	74%	92%
TWD-CO1-PA0	15%	56%	85%	97%	32%	60%	86%	97%
TWD-CO1-PA10	14%	42%	76%	94%	31%	47%	778%	94%
TWD-CO3-PA0	11%	37%	71%	92%	29%	43%	74%	92%
TWD-CO3-PA10	11%	38%	72%	92%	29%	44%	74%	93%

Table 5.9: OOV rates between the different methods on the clean and noisy test set. We show OOV rates for unigrams through to 4-grams.

tweet "**new orleans ... come hold my hand .. .**", and its translation: "**new orleans ... kom mijn hand nemen .. .**". On the left, we see that "hand" is incorrectly aligned with the target word "nemen", whereas in the alignment on the right, hand is only aligned with "hand". By concatenating the out-of-domain with the in-domain data, the cleaner Europarl data helps guide the learning of the word alignments, giving rise to models with less noise.

Backoff vs Interpolation Further, the reduced results seen in the self-learning approach with the application of the in-domain phrase table via the backoff mechanism *S-BO1*, where only unseen unigrams are translated with the new in-domain phrase table, corroborates previous work [171] in indicating that the main benefit of the self-learning approach is the of learning new in-domain-specific phrase-orderings for seen source phrases. In this backoff context, these new phrase-orderings for source phrases that exist in the out-of-domain data can not be used, and thus accounts for the differences between the BLEU scores of *S-CO1* and *S-BO1*.

Self Training versus Information Retrieval approaches Surprisingly, the retrieval methods performed worse then both the baseline and self-learning approaches. This

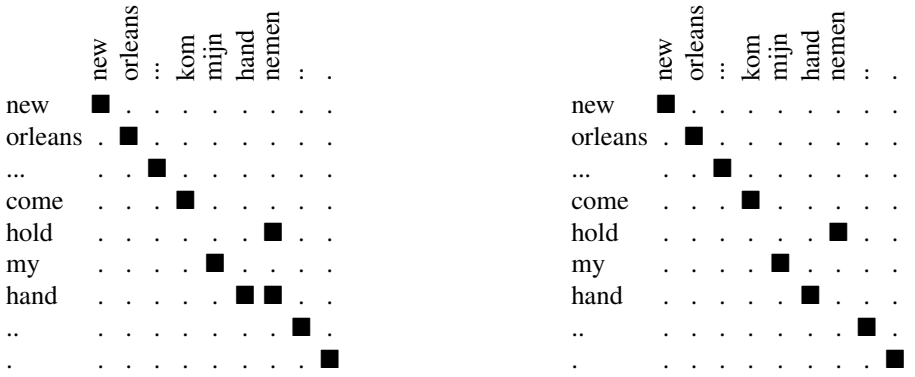


Figure 5.2: On the left we show the symmetric alignment points given the split approach, and on the right we show the alignments for the concatenation approach. Both show alignments between the source tweet: "**new orleans ... come hold my hand .. .**", and it's translation: "**new orleans ... kom mijn hand nemen .. .**".

observation holds, regardless of whether we used the concatenation or backoff strategy, or if we used the top 1 or top 3 returned target tweets in the pseudo-parallel bi-text, or even if we impose a minimum limit on the number of phrase alignments. This also runs contrary to previous work which has indicated further improvements over not just the baseline but also the self-learning approach [1, 85].

If we examine the differences in results given different settings for the phrase-alignments heuristic, we see that having a limit, albeit very small at 10, does lead to a general trend of improved results over the same models which do not use any limits; i.e, on the clean set, the model TND-CO1-PA10 has a higher score of 24.47 then TND-CO1-PA0 of 23.31, both with respect to BLEU. This trend holds for all TND and TWD models on both the clean and noisy sets. Thus, we posit that there is a general trend to learning better translations of unseen terms when some threshold at the phrase-alignment level between the source and IR retrieved tweets is applied. Self-learning approaches are constrained, by the nature of the approach, to learning new phrase-reordering without explicitly attacking the OOV problem, and thus avoid introducing noise. It is possible that a large limit would lead to more improvements.

To validate this, we show in Table 5.10 the translations for a single, clean, source tweet in the test set. Both the baseline (B) and IR approaches (specifically, this is the TND-CO1-PA10 approach) incorrectly translate "cold" into its nominal form, that is "koude," and not as an adjective, "koud." Interestingly, we see that the self-learning

Source:	cold beer , sun &
Reference:	koud bier , zon &
Translations:	
B	**koude** bier , zon &
S	koud bier , zon &
IR	**koude** bier , zon &

Table 5.10: Translations output by the different models, along with the reference translations, for a single example source tweet (clean version). We highlight in bold the incorrect terms in the Baseline (B) and (IR) approaches.

approach does translate "cold" correctly. If we look at the phrase table entries of the different models, as shown in Table 5.11, we see that according to the baseline and IR translation models, "koude" is the translation with the higher weight. The LM disagrees, giving the translation with "koud" higher weight then the alternative, but during translation in the baseline and IR approaches, is outweighed by the higher translation model scores.

So one can see from the phrase tables entries why the baseline and IR approaches output "koude" instead of "koud", but what is more interesting analytically, is why the self-learning approach was able to learn translations for the phrase "cold beer", when the IR approach could not. Indeed; they both had access to the same corpus of 1 million English tweets. Looking through those tweets, we found 15 that contained the phrase "cold beer". In Table 5.12, we show a couple of these tweets, along with the tweets retrieved using the IR approach, and the translations using the baseline translation models, which are used in the self-learning approach. These are representative of the full list of tweets, which we present in Table A.1.

For the IR retrieved tweets, only one of the 15 returned tweets (the first example in Table 5.12) contained an appropriate phrase for "cold beer", that is "koud biertje". In contrast, "koud bier" was seen 4 times, enough to be learnt as a phrase. The other occurrences were made up of the phrases "koude bier", "koud biertje", and "koude biertje", all arguably an improvement over the phrases in the IR retrieved tweets.

Thus, going back to the phrase table entries in 5.11, we can see that while the correct translation of cold beer has a marginally lower weight than the incorrect translation, the difference is not so small as to outweigh the LM. To summarise this focused analysis on the examination of a single phrase-application, we see that the quality of returned tweets using the IR approach in our setting is, even with the minimum phrase-

Source	Translation	Weights			
Baseline:					
cold	koud	0.708333	0.266375	0.0731707	0.0580952
cold	koude	0.943888	0.874576	0.675753	0.491429
Self learning:					
cold	koud	0.831111	0.758755	0.316199	0.30593
cold	koude	0.961376	0.9559	0.538722	0.503295
cold beer	koud bier	1	0.651003	0.235294	0.223663
cold beer	koud biertje	1	0.745673	0.235294	0.0146538
cold beer	koude bier	0.9	0.820151	0.529412	0.367955
IR:					
cold	koud	0.489362	0.173599	0.0961338	0.122137
cold	koude	0.936027	0.719372	0.580982	0.218511

Table 5.11: Translation entries in the phrase tables of the different models for the source phrase "cold," and "cold beer" where it exists.

alignment heuristic, still rather poor.[6]

Finally, on a different note, another potential explanation for the difference between the negative impact we report the IR approach to have, and the positive results published in the literature [1, 85], is due to the metric we use. We used a raw TF-IDF scheme, as opposed to an altered retrieval metric that directly incorporate phrase table scores in the ranking score [205]. Nonetheless, the reasons behind the negative results we report using the IR approach, and why this differs from the trend reported in the literature, remain an open question.

5.5 Conclusions

We examined two different approaches to learning microblog specific translations for adapting an existing out-of-domain SMT system. The self-learning approach uses the translations of a large corpus of in-domain source sentences directly as additional, pseudo-parallel data. The information retrieval approach uses these translations as a

[6] Actually, we only presented the returned tweets from a single IR approach, but the others do not return any higher quality, or more relevant, tweets.

original tweet
been there . hope you are home right now with a cold beer .
ir retrieved tweet
zo , trainen van vanavond zit er weer op. nu thuis op de bank met een overheerlijk koud biertje
self translation
@loriming er geweest . hoop dat je huis nu met een koud biertje .
original tweet
phew finishing work , looking forward to a cold beer !!!
ir retrieved tweet
wat een storm , phew !
self translation
phew laatste werk , hopen op een koude bier !!!

Table 5.12: In this table we list some of the tweets in the corpus of English tweets that contained the phrase "cold beer," and either the IR retrieved sentence, or the translation using the out-of-domain phrase table, the very same translations that are then used by the self-learning approaching.

query to an index containing target sentences, or in this chapter, target tweets. The returned tweets are then used with the original sentences as additional parallel sentences.

Even though the self-learning approach can not learn translations for originally unseen or out-of-vocabulary terms, the self-learning approach improves on the baseline by two BLEU points. The IR approach performs worse then the baseline; reducing the BLEU score by circa 2 BLEU points. The self-learning approach, with the in-domain pseudo-parallel data appended to the out-of-domain bitext prior to word alignment, achieved the best improvements over the baseline. In comparing the results with those output by the self-learning approach using the in-domain data in a backoff setting, our results concur with previous research [85, 171] in demonstrating that the main benefit of self-learning approach is learning new, in-domain, phrase-orderings.

We demonstrated the importance of filtering to the information retrieval approaches. Without restricting the returned target translations to have a minimum number of alignment points, we learn nonsense translations, so OOV rates reduce, but so do BLEU

scores. Our results indicate a correlation with previously published work in indicating, though not conclusively, that IR approaches need to filter the tweets returned by the index with an alignment heuristic. Runs with no filter in this work performed the worst. It is possible that just rerunning the IR approach with higher minimum alignment thresholds would raise the IR approach results above the baseline.

A potential, though unexplored, explanation for the lack of improvements using the retrieval based approached over the self-learning approaches is that we did not limit our monolingual corpus of English and Dutch tweets to be on a specific topic. Previous work [85] used a regional event, the "Arab Spring," to create the monolingual sets which were used as a basis for phrase translation mining. Given the global ramifications of the event and it's widespread reporting in both the formal news-outlets, and more informal news-resources such as Twitter, this made sense. In this work, working in the English to Dutch setting, we made no such event or topic restriction. An examination in future work of explicit topic or event filtering would lead to learning appropriate translations instead of noise. We also leave to future work an examination of alternative retrieval approaches, including those that directly integrate the out-of-domain phrase table into the ranking metric [205, 206], and approaches moving beyond the surface form to exploit latent semantics.

Part II
Exploitation

In Part I of the thesis, we examined the challenges involved in exploring and mining the large amount of content available on a popular microblogging platform, with the final aim of adapting an out-of-domain SMT system for translating microblog posts. In Part II of the thesis, we turn our attention towards better utilising existing resources. That is, instead of aiming to acquire more relevant data, we examine different approaches that dig deeper into existing resources. Specifically, we examine in detail different approaches to improving the quality of translation in the pre-processing and post-processing steps.

In Chapter 6, we examine the potential of syntactic features for a reranking n-best lists. Given a thorough analysis of a variety of features, we propose a naive, content-unaware POS tagger. We then turn to the pre-processing step in Chapter 7, and propose a novel algorithm for inference of high-order HMMs, and apply it to the task of source-side decomposition of Finnish, a morphologically rich language that is highly agglutinative. Contributions of Part II of the thesis include the examination of a large number of different syntactic features, extractable from different toolkits, for reranking translation output, and the proposal of an algorithm for the joint inference tasks of decoding and sampling from high-order HMMs.

Chapter 6

Discriminative Syntactic Reranking

Having looked at improving the phrase table via domain adaptation techniques using comparable corpora in Part I, we turn our attention to language models. As explained in Chapter 2, LMs, alongside translation models, form the core of modern SMT systems, whether they be phrase-based [106] or hierarchical systems [44, 45]. A language model's primary job in an SMT system is to check and enforce the fluency of a translation candidate without any knowledge of the source sentence being translated. In so doing, the language model impacts on word order decisions and translation selection decisions.

The most commonly used language models are word n-gram models, which make the markovian assumption and base the probability of a word following a prefix string on a limited context of the previous $n-1$ words. Since n-gram language models utilise a limited lexical context when deciding on the overall fluency of a sentence, the use of syntax, which contains deeper, and consequently long-range dependency information, is intuitively appealing. This brings us to our third main research question:

RQ 3. Can a generative syntactic parser discriminate between human produced and machine output language? Are they useful in an SMT setting for reranking output, where global information may prove beneficial to translation quality?

We saw in Chapter 5 the improvements that could be made by learning new phrases with different word orderings, giving more options during decoding from which the

language model can choose more fluent translations. Here we seek to understand if actually digging deeper, that is going beyond the surface of the text to the underlying syntax, can lead to better language models, again to help in picking translation hypotheses with better word-orderings representing more fluent translations. However, the application of syntax to machine translation is not straightforward (see Section 6.1 for a full discussion on previous work attempting to integrate syntax into the components of an SMT system). A problem arising from the use of syntactic toolkits for improved machine translation is that such models often expect the input to be well-formed, human produced, text. Yet, in the context where we wish to utilise syntax, and are this working with machine produced sentences, these assumptions do not hold. This leads to the first research subquestion of this chapter:

RQ 3a. Given that syntactic models are trained only on examples of good text, and are optimised towards parse retrieval as opposed to LM probability, to what extent can they be used for differentiating between human and machine text, and more importantly, to what extent can they differentiate between different machine produced texts?

In examining the use of syntax to rerank translation hypothesis, there are many toolkits from which we can extract a variety of different features. Parsers provide a full parse-tree for a given input sentence, but to do so, have a $O(n^3)$ complexity in relation to sentence length. POS taggers, on the other hand, have a linear complexity in relationship to the input sentence length, but only output a shallow POS sequence. It is interesting to see if the additional information contained within a full parse allows for the extraction of features that give bigger improvements in translation accuracy beyond those extracted from a POS tagger. Specifically, the next research subquestion we seek to answer is:

RQ 3b. As input to our machine learning algorithms for reranking, we extract and compute many different feature types from differing syntactic models. What features help most, and from which syntactic models are they best extracted? Does the expressiveness of a full lexicalised parser outweigh the computational disadvantages in comparison to the use of faster, less expressive syntactic models, such as POS tagging?

To summarise, by answering these research questions, we hope to gain a better understanding of the extent to which different syntactic analyses can handle noisy input text, and of the potential for a variety of different syntactic features for reranking translation output. In addition to answering the aforementioned research questions, the contributions of this chapter include:

- Demonstrating that the use of full parse tree features do not provide the improvements expected.
- Nonetheless, empirically showing that improvements can be gained by using features extracted from a novel, context-insensitive Part-of-Speech (POS) tagger.
- Finally, demonstrating that these features lead to larger gains than using a state of the art conditional random field (CRF) POS tagger on two of the three test sets.

The remainder of this chapter is structured as follows; we discuss related work in Section 6.1. We then examine the ability of a parser to discriminate between sentences of varying degrees of fluency in Section 6.2. We introduce discriminative language models in Section 6.3, present the perceptron model used in Section 6.3.1, and describe the syntactic features in Section 6.4. Experimental results are reported and discussed in Section 6.5, and we conclude in Section 6.6.

6.1 Related Work

One manner to overcome the data sparseness problem of n-gram language models has been the generalisation of existing data to encapsulate greater information that goes beyond the surface form. Class-based models [25] generalise from the word form to an abstract representation based on word clusters. These models are often used during decoding via interpolation with a standard lexical model. The class-based models are further generalised via factored language models (FLMS) [16, 17, 103]. Here, each word is represented by as many different forms required, from morphological information to supertags and partial parses [17]. Both of these move beyond the surface word form, but stay within the n-gram framework.

A study in the use of a range of generative syntactic models was undertaken by Och et al. [140, 141], who did n-best reranking as we have done. Syntactic features were not successful. The approach of [154] integrates a parser as language model into the decoding framework, however they are unable to outperform the baseline. Unlike the two previous approaches which attempted to utilise full parse trees for improving SMT output, the utility of shallow parse information has been demonstrated by Hasan et al. [79] for the translation of speech corpora. Supper-tagging, lightweight dependency analysis, a link grammar parser and a chunk parser are used to rerank n-best lists within a log-linear framework.

Shen et al. [175] are the first to use a perceptron-like algorithm in a small-scale application of reranking SMT n-best lists. They used the algorithm to optimise weights

for a small number of features (tens instead of millions). The use of perceptron type algorithms with millions of features for SMT has been explored by Arun and Koehn [8]. They examine the use of online algorithms for the discriminative training of a phrase based SMT system. In this chapter we focus on the use of perceptrons for reranking using only target side syntactic information. Other researchers have attempted to examine the training of specific features types; from the training of reordering features [38, 189], translation model features [20], independent target and source side features [47], and both translation and language model features combined [46, 117, 175, 202].

The work most closely related to ours is the discriminative syntactic LM proposed in [52]. The work presented in that paper differs from ours in two important aspects. First, we focus on the use of syntactic features in SMT. Second, we propose the use of a simple POS tagger, which gives us significant improvements over a 1-best baseline and competing state of the art reranker. Li and Khudanpur [116] apply the framework of [160] to create the first large scale discriminative language model to SMT for reranking. Using a standard n-gram feature set, they outperformed the 1-best output of their baseline SMT system. They focus on the application of n-gram only models to SMT and the use of data filtering thresholds.

This chapter extends our previous work in [29, 30] in three ways; first by expanding upon the original experiments in [29] with two new additional test sets, and three higher order LMs, demontrating the validity of the results and conclusions drawn. Second, we motivate the work first presented in [30] by giving concrete examples of where parsers go wrong, motivating the best performing S-POS reranker. Finally, we present new deep, syntactic reranking, models, and provide a more detailed analysis of both deep and shallow syntactic features for reranking SMT output.

6.2 Parsing Ungrammatical English

In this section, we examine the ability of parsers to differentiate between fluent and disfluent English. Specifically, we look at two tasks. The first, in Section 6.2.3, is differentiating between SMT output and human produced English. If a state of the art parser is unable to do this task as well as a standard LM, then it is not suitable for use within an SMT system, as suggested by [141]. The second, harder, task, reported in Section 6.2.4, is to differentiate between varying degrees of fluent sentences produced by a SMT system. This second set of experiments is representative of a standard reranking task.

6.2.1 Parser

We used an implementation of Collins Model 2 (CM2) by [15] to provide parses and parse probabilities. Collins Model 2 uses four probability distributions for assigning probabilities to (i) head labels, (ii) sibling labels and head tags, (iii) sibling head words, and (iv) sub-categorization frames. We chose Model 2 above Model 1 because of the higher reported Labeled Recall and Precision scores [49]. The parser was trained on sections 02–21 of the Wall Street Journal portion of the Penn Tree Bank (PTB) [120], about 40,000 sentences, and tested on section 23, about 2,500 sentences.

We compare the parser against a large 4-gram language model built using the AFP and Xinhua portions of the English Gigaword corpus (LDC2003T05) and the English side of the bitext.

6.2.2 Experimental Setup

In this section, we provide details of the experimental set-up for our experiments analysing the ability of parsers to differentiate fluent and disfluent English.

Moses was used as a state of the art baseline SMT system for reporting experimental results [107]. The parameters used for the experiments discussed here are as follows: stack size of 100, distortion limit of 6, and phrase table limit of 20. We utilise lexicalised reordering along with standard models described in Section 2.2.

To build the phrase table and LM, we used five corpora distributed by the Linguistic Data Consortium (LDC), totalling 300 thousand sentence pairs. The parallel text includes Arabic news LDC2004T18, automatically extracted parallel text LDC2007T08, eTIRR news LDC2004E72 and translated Arabic treebank data LDC2005E46. Alignments were extracted using the GIZA++ toolkit [137]. Minimum Error Rate Training (MERT) [136] was used for optimising the parameters of the Moses baseline SMT system.

For Arabic-to-English translation, performance is evaluated using NIST's MT-Eval benchmarking sets from 2002 through to 2006. Statistics for each set (#source sentences/#refs): MT02 (1043/4), MT03 (663/4), MT04 (1353/5), MT05 (1056/5), MT06 (1796/4). Sets MT02 and MT03 were used for development.

6.2.3 Discriminating between SMT and Human Translations

Formally, our experiment is composed of a set of SMT output and their respective reference translations, which in our case is 4 each. We say a sentence set is a specific SMT output sentence with its respective references. Unlike in earlier work published in [29], no thresholding based on SMT output sentence length is applied. We define the

	No Normalisation			Normalised		
Model	MT04	MT05	MT06	MT04	MT05	MT06
LM4	13.6	28.1	15.28	23.08	30.78	30.68
CM2	**34.09**	**49.1**	**33.6**	**61.09**	**54.85**	**63.37**

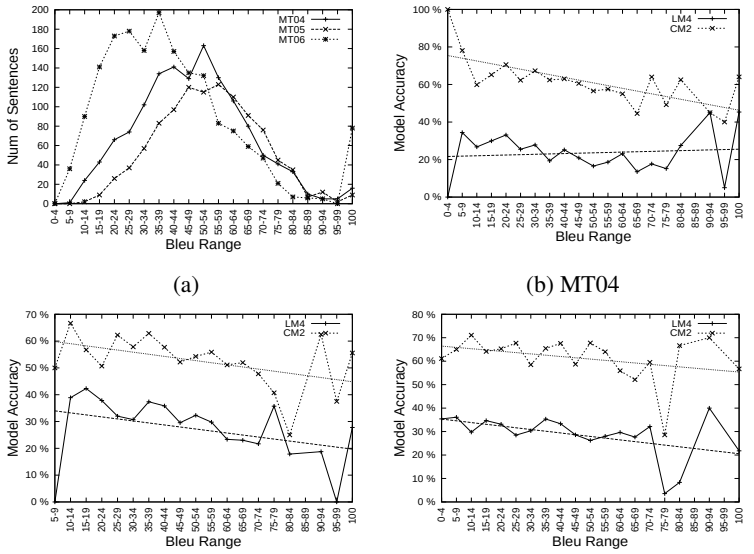
Table 6.1: Percentage of references with higher probability than respective SMT output for the test sets MT04, MT05 and MT06. We present results when using no normalisation and length normalisation.

classification accuracy of a model to be the percentage of references which are assigned an equal or higher probability than the MT output.

We show in Table 6.1 the percentage of reference sentences that were assigned a higher or equal probability to the respective SMT output by each of the different models on the three different test sets. All results are reported using simple length normalization, where the log probability is divided by the sentence length, as we found this to have a positive impact. The CM2 parser outperforms the n-gram language model. Further, when using length normalisation, the parser performs better than chance, while the language model continues to perform worse than 50%.

To see to what extent these findings hold for SMT sentences of different degrees of fluency, where fluency is approximated by BLEU, the standard evaluation metric for SMT systems [144], we bucketed the reference sentences by the BLEU score of the SMT output. We would expect that SMT output that has a higher BLEU score is harder to differentiate from the human produced references. Results are displayed in Figures 6.1b, 6.1c and 6.1d. The correlation between BLEU range and model accuracy is measured by applying linear regression. Unsurprisingly, as shown by the linear regression lines, classification accuracy appears to decrease as BLEU increases for all three test sets; the better the SMT output, the harder it is for both models to differentiate between good and bad English.

To take a closer look into which sentences the models were having problems with, we bucketed the SMT translations by their length, and examined the classification accuracy for each bucket for each model. Results are displayed in Figures 6.2b, 6.2c and 6.2d. Even for small reference sentences in the range of 1-4 words, the parser still outperforms the n-gram LM. Moreover, as the regression lines show, the parser performs better as the sentence length increases, in contrast to the 4-gram language model, whose performance decreases with longer translations. The parser is able to exploit deep syntactic structures that capture long range information within the sentence to assign it a parse whose probability better reflects the fluency of the translation,



(a) (b) MT04

(c) MT05 (d) MT06

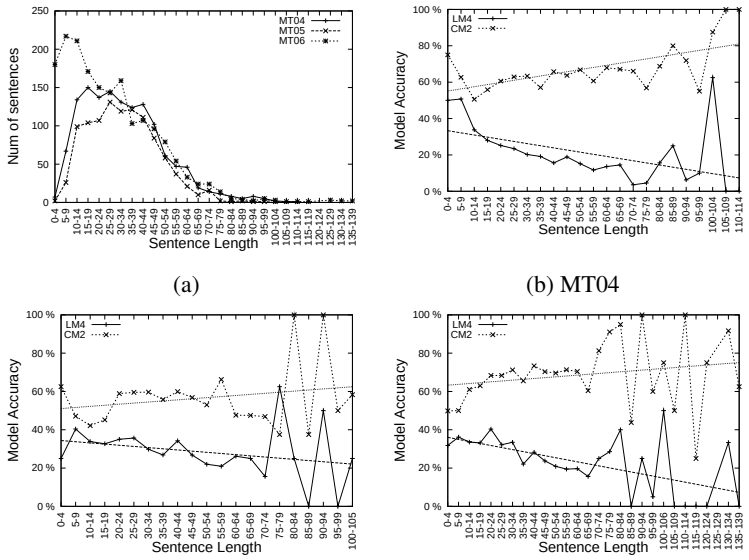
Figure 6.1: Breakdown of model performance in discriminating between SMT output and human produced references using length normalised probabilities of a 4-gram LM and state of the art parser. Figure 6.1a shows the number of sentences per BLEU bin for each of the three test sets. Figures 6.1b–6.1d give a breakdown of classification accuracy by BLEU score of SMT sentence for MT04, MT05 and MT06 test sets. Figures 6.1b through to 6.1d have linear regression lines drawn for both models showing the underlying trends.

whereas the n-gram LM, limited to a fixed history, is unable to utilise this information.

These results are interesting, as parsers perform worse on standard precision and recall measures as sentence length increases [125]. This demonstrates that these measures, the evaluation metrics that parsers are traditionally developed and optimised towards, are not necessarily indicative of a parsers ability to differentiate between SMT output and human produced translations.

6.2.4 Correlating Parser Scores with Translation Quality

Considering a parser's ability to better discriminate between SMT output and human translations, we would like to use parsers for reranking. As a first step we examine the



(a)

(b) MT04

(c) MT05

(d) MT06

Figure 6.2: Breakdown of model performance in discriminating between SMT output and human produced references using length normalised probabilities of a 4-gram LM and state of the art parser. Figure 6.2a shows the number of SMT sentences in each length bin. Figures 6.2b–6.2d show the classification accuracy of the length normalised models by length of the SMT sentences for the three test sets. Figures 6.2b through 6.2d have linear regression lines drawn for both models showing the underlying trends.

correlation between the probabilities assigned to a parser in an n-best list and smoothed BLEU [118]. We use the same SMT system and models as before. For each source sentence we output an n-best list of translation candidates. We score each sentence in the n-best list according to smoothed BLEU, and score each sentence with both the n-gram language model and parser. We convert these scores to ranks, tied where necessary, and compute the Pearson Correlation coefficient between the BLEU and model rankings. The correlation co-efficients are then averaged over the entire test set. Results are reported in Table 6.2. There is no correlation between the rankings assigned to the n-best lists by either n-gram or parser models and BLEU.

Model	MT04	MT05	MT06
LM4	-0.045	0.066	-0.027
CM2	-0.027	0.036	-0.046

Table 6.2: Average Pearsons moment correlation coefficient between the two different models and BLEU rankings for each of the MT04, MT05 and MT06 test sets.

6.2.5 Analysis

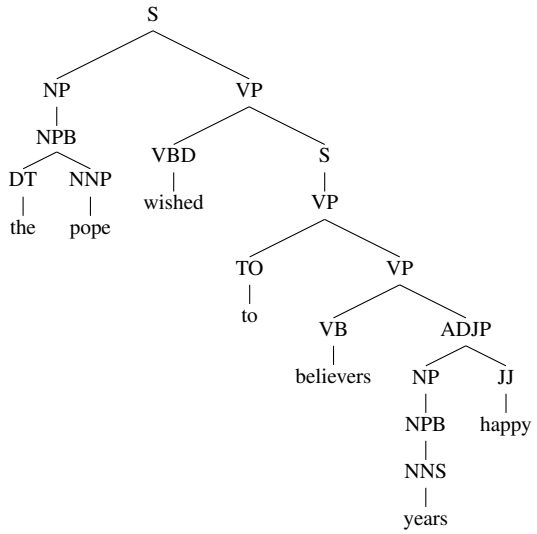
While the parser is able to discriminate between SMT output and human-produced sentences, the results reported in Section 6.2.4 highlight the difficulty in using the probabilities for discriminating between sentences of varying degrees of fluency and grammaticality. A look to the literature offers some potential explanations.

In analysing why the inclusion of scores from a parser, similar to the one used in this chapter, did not aid in improving BLEU scores during decoding [154], the authors argue that the use of a single probability to represent the quality of a single parse is too coarse a metric for the successful exploitation of parse tree information.

These results corroborate the earlier experiments of [141], which demonstrate the inutility of a generative parser for reranking. The results from our averaged correlation experiments in Section 6.2.4 support the literature in showing that the direct use of a single probability to rank an n-best list does not lead to BLEU improvements.

To give an example of why, we show in Figure 6.3 a parse by Collins Model 2 over a 1-best sentence output from the MT04 test set. Here, the length-normalised log probability assigned by the parser to this disfluent sentence was higher than those assigned to all four reference sentences. The parser incorrectly tags ‘believers’ as a verb instead of a noun. The parser does this to attain a good structure; the cost of correctly tagging ‘believers’ as a noun is too prohibitive. A quick glance at the WSJ training corpus hints at a reason why: out of 21 thousand instances of TO, its nearest right most sibling is a verb phrase (VP) 13,001 times, and a noun phrase (NP) 8,460 times, making the former more likely. In assuming the input is a well formed fluent English sentence, assigning the most likely parse tends to good high level structures that mask disfluency at the local level. In comparison, the n-gram language model, which makes only local decisions, correctly assigns the sentence a lower length normalised probability than its references.

Looking at the example parse, one can formulate a number of different features that could be used to distinguish between a fluent and disfluent sentence. A lexicalised parent-head based feature, similar to that used in [52], could determine that



Type	Sentence	LM4	CM2
SMT	the pope wished to believers years happy	**-26.61**	-5.5
REF 1	pope wishes worshipers happy new year	-22.75	**-7.13**
REF 2	pope wishes believers happy year .	-24.21	-6.01
REF 3	pope wishes the faithful a happy year	-25.86	-5.72
REF 4	pope wishes happy year for the faithful	-23.3	-5.92

Figure 6.3: Above we show the parse over an SMT translation of the MT04 source segment 139. In the table below we show the same translation, along with the four human produced reference translations. In the two final columns we print the length normalised log probability assigned to each sentence by the 4-gram LM (LM4) and Collins Model 2 (CM2) parser. The parser assigns the SMT output a higher normalised log probability than the four references translations, while the n-gram LM assigns the sentence the lowest normalised log probability. We highlight in bold the lowest scores output by both models.

a VP headed by 'believers' is unlikely. Further, a shallow POS n-gram sequence extracted from the parse could show that the trigram VB NNS JJ is improbable.

There are many features, deep or shallow, that could be extracted and learnt from the parse that lead to potential gains in a reranking setting. Our solution is to examine the potential benefits of multiple syntactic features exploited in a discriminative language model framework. Such a framework would allow us to conduct a thorough investigation of the different types of syntactic information extractable from a full parse tree in a computationally practical manner.

6.3 Syntactic Discriminative Language Models

Discriminative language models (DLM) consist of a function $\phi(\cdot)$ that maps a sentence onto a feature space and weight vector w [160]. For training, negative and positive examples are supplied to the DLM for learning the weight vector. The weight vector and feature function is then used to assign a non-probabilistic score to an unseen sentence.

A benefit of using discriminative techniques over generative models is that standard generative LMs are trained exclusively on well-formed English. Given the large feature space they operate in, the accurate assignment of probabilities to unseen events is a difficult problem, and has been a major field of research for the past sixty years (for a detailed overview on language modelling and smoothing techniques, see [42]). Discriminative models are trained with positive and negative examples, and therefore learn to assign negative weights to harmful features, without having to infer this from positive data only.

Different parameter estimation methods to estimate the weight vector w for a DLM have been previously examined in the Automatic Speech Recognition (ASR) domain [159, 160]. These include optimising the log-likelihood under a log-linear model, a batch algorithm which requires processing all the data before outputting a weight vector as an answer, and approximating a 0/1 loss through the perceptron update rule, and an online algorithm which examines and updates the parameter vector sequentially. The reader is referred to [65, 159] for a discussion on the benefits of the log-linear model and the perceptron. Given that this chapter examines the use of a syntactic feature space, which is larger than an already large n-gram feature space, and that perceptrons perform feature selection as a direct consequent of its learning procedure, we opt to use the perceptron algorithm.

Algorithm 1 The standard perceptron algorithm

Perceptron

1: $w \leftarrow 0$
2: **for** $t = 1$ to T **do**
3: **for** $i = 1$ to N **do**
4: $y^i \leftarrow ORACLE(x^i)$
5: $z^i \leftarrow argmax_{z \in GEN(x^i)} \phi(z) \cdot w$
6: **if** $z^i \neq y^i$ **then**
7: $w \leftarrow w + \phi(y^i) - \phi(z^i)$
8: **return** w

6.3.1 Perceptron

The perceptron, proposed by [163] is an online error minimisation learner that, assuming linearly separable data, can theoretically converge to a solution that perfectly classifies the data [69]. The perceptron has been successfully applied to parse reranking [51], document reranking for IR [43, 53, 64], ASR reranking [52, 159, 178], and finally to SMT translation reranking [116, 175], where Chinese-English translation systems have been significantly improved.

6.3.2 Algorithm

The standard perceptron algorithm is shown in Algorithm 1. The algorithm takes as input a set of n-best lists X, a function *GEN*(x^i) that enumerates over each sentence in a n-best list x^i, and an oracle function *ORACLE*(x^i) that determines the best translation (oracle best) for each of the n-best lists x^i according to the BLEU metric. As DLMs make comparisons on the sentence level, we use sentence level BLEU with additive smoothing [118]. There are discrepancies between sentence and corpus-level BLEU, however we find sentence-level BLEU sufficient for reranking SMT. T defines the number of iterations and N defines the size of the test set, which in our case is the number of n-best lists. The algorithm iterates over the n-best lists in a sequential manner (lines 2 and 3). If the selected hypothesis and oracle best sentence match, the algorithm continues to the next n-best list. Otherwise, the weight vector is updated (line 7). Finally, it returns a weight vector as its solution (line 11).

To use the weight vector returned by the perceptron algorithm, each sentence z in an n-best list is scored by:

$$S(z) = \beta\phi_0(z) + w \cdot \phi(z) \tag{6.1}$$

The SMT model score for each translation hypothesis $\phi_0(z)$ is weighted by β. Roark et al. [160] argue that, while it is possible to include $\phi_0(z)$ as a feature of the perceptron model, this may lead to under-training, so we adhere to the convention of using a fixed value for β.

To score an n-best list x^i we use the weight vector returned by the perceptron to assign a score to each sentence and select the best one:

$$z^* = \mathrm{argmax}_{\mathrm{z}\in\mathrm{GEN}(\mathrm{x}^\mathrm{i})} S(z) \tag{6.2}$$

6.3.3 Variants

A shortcoming of the perceptron is that it can be unstable if the training data is not linearly separable. A number of solutions have been proposed in the literature. One solution is to use an averaged perceptron [69], where the parameter vector w output by the algorithm is averaged over each instance $w_{avg} = \Sigma_{t=1}^{T}\Sigma_{i=1}^{N}\frac{w_t^i}{N\cdot T}$. Another solution is the pocket perceptron [51, 71], where the weight vector returned is the one that correctly classifies the most training instances in a row, keeping an optimal model in its 'pocket'. A third solution, called the committee or voting perceptron, keeps a cache of optimal models, sorted by their success counts [64, 158]. The cache sizes differentiate the voting and committee perceptron, with the voting perceptron using the best cached model, and the committee perceptron utilising the top-n cached models. As previous published work on using perceptrons for reranking SMT output utilised the average perceptron [116], we also use this model.

6.4 Features

In examining different syntactic features we distinguish between deep features, those extractable from a syntactic annotation layer that goes beyond pre-terminals, and shallow features which require only POS tags. In Section 6.4.1, we outline the different toolkits that are used to extract features. In Section 6.4.2, we detail the features used that can only be extracted from a full parse tree. In Section 6.4.3 we explain the features that can be extracted from a POS tagger. In Table 6.3, we list the features we examine this in chapter, and provide references for those feature types that have been explored, albeit for different tasks, in either a SMT or ASR setting.

6.4.1 Annotation Layers

In this section, we outline the different syntactic analyses we extract our syntactic features from. Some of the features examined can only be extracted from a full parse tree.

Feature Type	Field	Task	Citation	helpful
SEQ-B & SEQ-C	ASR	reranking	[52]	yes
CFG	ASR	reranking	[52]	yes
HEAD	ASR	reranking	[52]	yes
T-DEPTH	SMT	difficult-to-translate phrase localisation	[127]	yes
UNIT-P	-	-	-	-
NT-C	SMT	reranking/decoder features	[47, 140]	no/yes
NO-C	SMT	-	-	-
POS	ASR / SMT	reranking/FLM features	[17, 52]	yes
VERBAGR	-	-	-	-
POSNUM	-	-	-	-
NOPOS	SMT	reranking/decoder features	[47, 140]	no

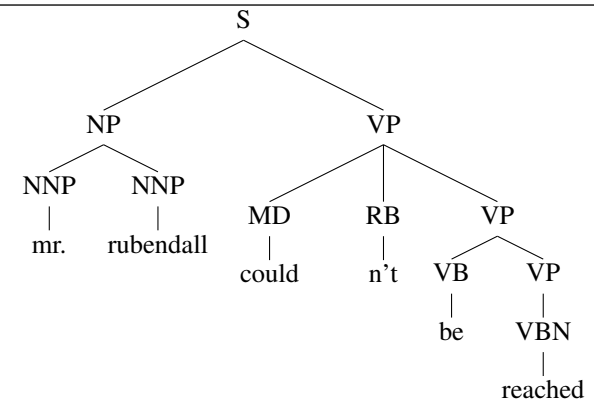
Table 6.3: The different syntactic features examined in this chapter. During experimentation different feature combinations are examined. Where a feature has been previously used, we list the field, task and citation, and whether or not it proved useful.

Others can be extracted from either parsers or taggers.

It is interesting to see if there is benefit of using features extracted from full parse trees as opposed to those extracted from a POS tagger or other shallow representation. Using parsers allows us to extract features that are global to the sentence and relay deep structural information. Unfortunately, they are slow and memory intensive, and may fail to return a parse for long sentences, as they have $O(n^3)$ complexity in relation to sentence length. On the other hand, POS taggers, while outputting no deep syntactic information, are more efficient and robust, as they always output a complete POS sequence for a given input sentence. As we continue to use the parser introduced in Section 6.2.1, we do not mention it again here.

CRF Tagger (CRF) We used Xuan-Hieu Phan's implementation of a Conditional Random Field tagger as our state of the art POS tagger.[1]

[1] Available at `http://crftagger.sourceforge.net`.



(a) Example parse

mr/NNP rubendall/NNP could/MD n't/RB be/VB reached/VBN

(b) POS

mr/NP^b rubendall/NP^c could/VP^b n't/VP^c be/VP^b reached/VP^b

(c) SEQ-B

mr/NNP-NP^b rubendall/NNP-NP^c could/MD-VP^b n't/RB-VP^c be/VB-VP^b reached/VBN-VP^b

(d) SEQ-C

Figure 6.4: In 6.4a we show an example parse tree, and in 6.4b, 6.4c and 6.4d we show the POS, SEQ-B and SEQ-C sequences extracted from 6.4a.

Simple Tagger (S-POS) We also use a simple maximum likelihood POS tagger, which assigns to each word the most likely tag according to a training set, regardless of any context. The simple model does not use any smoothing, meaning that out-of-vocabulary items are simply assigned <UNK> as their tag.

The use of a simple POS tagger is motivated by our analysis conducted in Section 6.2.5, where a manual evaluation of parse trees indicated that parsers provide good structures over disfluent English by incorrectly tagging words. Assigning words their most likely POS tags according to unigram estimates should allow a discriminative language model to better identify and penalise reordering mistakes.

6.4.2 Deep Features

Sequential Rules (POS, SEQ-B and SEQ-C) From the full parse tree, we extract three different layers. Figure 6.4 shows the three annotation layers we extract from the parse tree shown in Figure 6.4a. In 6.4b, the sequence is the POS tags for each word. 6.4c captures chunk-based sequences by associating with each word the first non-POS ancestor node. For each word, it is also indicated whether it starts or continues a shallow chunk (^{b} for the former and ^{c} for the latter). The sequence 6.4d is similar, but includes the POS tag of each word.

From the POS, SEQ-B and SEQ-C layers, as well as from the S-POS and CRF-POS output, we extract the following features, where w_i is a word at index i, and t is a tag specific to the layer we are extracting from:

$$(t_{i-2}t_{i-1}t_i), (t_{i-1}, t_i), (t_i), (t_i w_i)$$

Context Free Grammar Rules (CFG) Each feature is a basic context free grammar (CFG) rule used in the parse of a sentence. CFG's form the building blocks of most parsers, and except for rules from pre-terminals to leaf nodes, are entirely non-lexical. Thus these features capture information about categorical reordering decisions, such as Noun Phrase (NP) before Verb Phrase (VP), or vice versa. While simple, this feature type can capture long range reordering mistakes.

Syntactic Head Features (HEAD) We model head-parent relationships, such as NP headed by NN (syntactic), represented by NP/NN, or NP headed by car, NP/car, (lexical). These features are extracted for each Non Terminal (NT) in the tree. Heads denote the most important syntactic child of a phrase category. Heads are extracted using the hand crafted rules defined in Appendix A of [50].

As well as the more simple head-parent relationships, we also model more complex head-to-head dependencies within the parse tree, extracted to capture long range dependency relationships.

Given a parent NT P in a tree and its NT children (C ... C_k), we model the relationship between P, the head child of P C_h, and each sibling node of C_h. We denote the relative position between C_h and the sibling C_k with an integer, 1 if adjacent, 2 if not, positive if C_k is to the right, negative if to the left of C_h. Finally we note the lexical or POS head of C_h and C_k. The final feature is of the form: $P, HC, C_k, \{+, -\}, lex/POS, lex/POS$. Examples from Figure 6.4a include:

VP,MD,VP,2,could,be
VP,MD,VP,2,could,VBN
VP,MD,VP,2,MD,be
VP,MD,VP,2,MD,VBN

Tree Depth (T-DEPTH) This feature measures the maximum height of the tree. The intuition is that a deep complex structure is indicative of a particularly disfluent and ungrammatical sentence. Comparing all the baseline SMT translations with the reference sentences for MT04, we see that the SMT output has on average a far higher tree depth; the summed difference is between 20 to 56 levels higher for the SMT translations. We normalise this feature by sentence length.

Unit Productions (UNIT-P) Unit productions are rules in the grammar that have a single variable in both the left-hand side and the right-hand side. This feature takes the sum of all unit productions in the associated parse tree. Comparing all the baseline SMT translations with the reference sentences for MT04, we see that the SMT output has a summed difference of between 7.5 to 20 more unit productions. It appears that over generation of unit productions is indicative of a disfluent sentence, and thus include it as a feature.

NT Count (NT-C) This feature counts the number of non-terminal types in a sentence, normalised by sentence length. The aim is to capture the over-under production of certain feature types, a common problem in SMT output (e.g, a dropped verb).

Node Count (NO-C) This feature counts the number of nodes in a parse, normalised by sentence length. While simple as a feature, we note a general trend for parses of SMT sentences to contain more nodes than in comparison to human produced translations; the SMT output for MT04 has a summed difference of between 71 to 205 more nodes.

6.4.3 Shallow Features

POS n-grams We explore the use POS n-gram features, from unigram to trigram features.

Verb Agreement (VERBAGR) The verb agreement feature captures agreement between verb tenses that should match. We extract this feature by starting with each co-ordinating conjunction and comma in a sentence, and examine a window 5 words large on either side for verbs. If there are multiple verbs in this window, we return the nearest one either side. This feature is extracted only if we find a verb both to the left and right within the context length.

For example, given the sentence "George/NNP was/VBD shouting/VBG and/CC screaming/VBG," the verb agreement feature would be:

VBG CC VBG.

This feature can discriminate between the correct form "shouting and screaming " and the incorrect "shouting and screamed". Note this is not a trigram POS feature, as the verbs do not have to be adjacent to the comma or co-ordinating conjunction.

NT Length (POSNUM) It is also possible to extract features from the POS layers that capture frequency based information. In particular, we wish to model the frequency of POS types for a given translation length. Features are of the form:

$$length(x)/num(POS, x)$$

The length of a sentence is represented by $length(x)$, and the frequency a specific POS tag occurs in the hypothesis translation x is $num(POS, x)$. These features tie the number of POS tags to the length of a sentence. In this way we model the under-or-over production of certain POS types for specific sentence lengths. Here, we examine five such types: verbs, nouns, adverbs, adjectives and determiners.

POS Absence (NOPOS) A similar feature is one that models a lack of certain POS types, regardless of sentence length. Here again we model a lack of either verbs, nouns, adverbs, adjectives or determiners.

6.5 Syntax Experiments

In this section, we evaluate the impact of different syntactic features used by a discriminative language model on the MT04, MT05 and MT06 test sets. The SMT system and settings remain the same as those described in Section 6.2.2.

	MT04	MT05	MT06
Moses	48.97	53.92	38.40
+ DLM n-gram	49.57	54.42	39.08
Oracle	61.09	66.34	50.11

Table 6.4: Moses, n-gram reranked and oracle results on MT04, MT05 and MT06.

6.5.1 Parameter Optimisation

The SMT system is optimised on the MT02 and MT03 data sets. Since the parameters of the perceptron reranker also require optimisation, the development set was split into K folds. MERT was run on the union of the $K-1$ folds to optimise the parameters. The resulting setting was used to translate the remaining fold and to generate the n-best lists used for learning the parameter settings of the perceptron reranker. The n-best lists contain the top 1000 most likely and distinct translation candidates (different alignments can lead to sentences which are lexically identical but have different derivations). Untranslated source words were not removed from translations. Note, that the Moses baseline we compare against was still trained on all development data at once.

To optimise the β value in Equation 6.1, we performed a grid search, with increments of 0.1 examined between 0 and 1, and increments of 1 at 2^x thereafter, on the MT0203 set.

As we parse SMT output, all sentences were tokenised and lowercased in accordance with the output of the SMT system prior to training the parser. The simple unigram tagger was trained analogously. A sentence was assigned the <NOPARSE> feature if the parser failed to generate it a parse. The tagging accuracy of the parser and two POS taggers are as follows: CRF 97%, CM2 94.4% and S-POS 86.8%.

6.5.2 Results

Having detailed the different annotation layers and syntactic features we intend to explore, we now present experimental results. Table 6.4 presents Moses baseline 1-best results on MT04, MT05 and MT06 test sets. In addition to the Moses baseline, we present results using the averaged n-gram reranking model using unigram, bigram and trigram lexical features, as used by [116]. Finally, we also present oracle results in the last row of Table 6.4, demonstrating the large room left for improvement.

	MT0203	MT04	MT05	MT06
Moses	51.27	48.97	53.92	38.40
+ DLM n-gram	59.87	49.57	54.42	39.08
+ DLM n-gram + POS	59.70	49.47	**54.48**	39.07
+ DLM n-gram + SEQ-B	58.52	49.09	54.11	39.47
+ DLM n-gram + SEQ-C	60.37	49.46	54.19	39.07
+ DLM n-gram + CFG	59.89	49.53	54.44	39.58
+ DLM n-gram + HEAD	**61.53**	49.44	54.09	33.45
+ DLM n-gram + MAX-D	58.79	49.42	54.08	39.61
+ DLM n-gram + UNIT-P	59.51	**49.81**	54.39	**39.76**
+ DLM n-gram + NT-C	58.82	49.51	54.20	39.68
+ DLM n-gram + NO-C	53.52	47.14	52.68	36.92

Table 6.5: Results on MT development and test sets using syntactic features from full parse trees. We highlight in bold the largest scores.

Deep Features

In Table 6.5 we present the results of using our perceptron rerankers with features extracted from full parse trees. The use of features from full parse trees did not help at all for MT04, apart from the UNIT-P model, which gave improvements of 0.34 BLEU. For MT05, the CFG and POS feature sets show small improvements. Note for MT05 the UNIT-P model no longer gives improvements above the n-gram only model. For the MT06 test set, all syntactic models apart from HEAD achieve improvements. These improvements against the lexical only reranker do not hold for MT04 and MT05. Robustness is a problem; given unseen test data, we do not know if the inclusion of syntactic features from full parse trees will improve or harm the translation quality of the system.

POS Layers Compared

In comparing the impact of features extracted from different syntactic analyses, we conduct experiments with features extracted from the POS taggers. The results are displayed in Table 6.6.

	MT04	MT05	MT06
DLM n-gram	49.57	54.42	39.08
DLM n-gram + POS	49.47	54.48	39.07
Improvement	-0.10	0.06	-0.01
DLM n-gram + CRF	**49.74**	54.51	39.45
Improvement	**0.17**	0.09	0.37
DLM n-gram + S-POS	49.59	**54.60**	**39.48**
Improvement	0.02	**0.18**	**0.40**

Table 6.6: BLEU scores and improvements when using features from our two POS taggers and POS annotations from the full tree parser. POS features extracted from a simple unigram, maximum likelihood tagger give largest improvements on two of three sets.

The CRF DLM outperforms the n-gram only DLM model on all three test sets. The S-POS DLM yields gains over the DLM n-gram model on all three of the test sets also. Even though our S-POS tagger uses no backoff model or context, for two of the three test sets, it provides larger gains than the CRF tagger. Because the S-POS tagger results in higher scores than the CRF tagger for two of the three test sets, we only use the simple POS annotation layer for the following experiments.

Shallow Features using Simple POS Tagger

Table 6.7 summarizes the results of using the POSNUM, NOPOS and VERBAGR features. As for the POSNUM and NOPOS features, we look at specific POS categories (verbs, nouns, etc.), POS is replaced with the respective type V (verb), N (noun), D (determiner), RB (adverb), JJ (adjective) and ALL (all of the previous five types). For MT04, the best performing model is S-POS+noall, with a significant improvement at $p < 0.01$ over the DLM n-gram model of 0.13 corpus level BLEU.[2] For MT05, the best performing model is S-POS+V+DNUM with a significant improvement of 0.18 BLEU at $p < 0.01$. The S-POS+ALLNUM model gives the same absolute BLEU improvement for MT05, but is insignificant. For MT06, we have a larger improvement of 0.41 BLEU, again at $p < 0.01$, using S-POS+NOALL. The S-POS+V+DNUM model

[2]Statistical significance is calculated using the paired bootstrap resampling method [99].

	MT04	MT05	MT06
Moses	48.97	53.92	38.40
+ DLM n-gram	49.57	54.42	39.08
++ S-POS+V+DNUM	49.65†	**54.60**‡	39.67‡
++ S-POS+ALLNUM	49.65	**54.60**	39.67‡
++ S-POS+NOALL	**49.70**‡	54.46	**39.69**‡
++ S-POS+VERBAGRE	49.44	54.56	39.55‡

Table 6.7: Model results using POS tag frequency (vn, dn and allnum), lack of POS type (noall) and verb agreement (verbagr) features. We conduct significance tests between the syntactic models and the DLM n-gram model. Significance at $p < 0.01$: ‡. Significance at $p < 0.05$: †.

is not the best performing model on MT04 or MT06, but consistently gives significant improvements.

Deep + Shallow Feature Combinations

Finally, we investigate if a combination of deep and shallow features leads to improvements. As it is infeasible to try all feature combinations together, we pick the best performing deep feature type UNIT-P, and combine this with features extracted from our S-POS tagger. The combination of deep and shallow feature types do not lead to

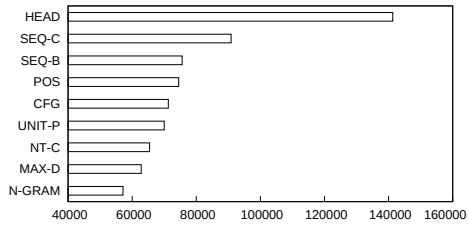


Figure 6.5: Number of active features (all features with non-zero weight) in each model.

Test Set	System	n-gram precision (%) 1	2	3	4
MT04	Moses	81.46	57.80	41.17	29.70
	+n-gram	**81.86**	58.36	41.72	30.28
	++syntax	81.76	**58.48**	**41.92**	**30.43**
improvement (%)		0.4/−0.1	1.2/0.2	1.8/0.5	2.5/0.5
MT05	Moses	83.18	62.34	46.66	34.93
	+n-gram	**83.31**	62.74	47.20	35.54
	++syntax	83.28	**62.96**	**47.43**	**35.74**
improvement (%)		0.1/−0.04	1/0.3	1.7/0.5	2.3/0.6
MT06	Moses	74.17	47.45	31.27	21.05
	+n-gram	**74.43**	47.84	31.75	21.50
	++syntax	74.31	**47.92**	**31.87**	**21.58**
improvement (%)		0.2/−0.2	1/0.2	1.9/0.4	2.5/0.4

Table 6.8: N-gram precisions and relative improvements on MT test sets above the Moses baseline and n-gram reranker. We highlight in bold the highest precision scores, and show percentage improvements of our syntactic model above the Moses baseline and lexical only reranker. For bigram, trigram and 4-gram precisions, syntactic rerankers achieve highest scores.

improvements over those presented in Table 6.3. We conclude that the deep features examined are redundant.

6.5.3 Discussion

An explanation for the under-performance of features from full parse tree is over-training. Examining the performance of the syntactic models on the MT0203 development set, SEQ-C and HEAD models perform considerably better than the n-gram only model and other syntactic models. This performance does not carry through to the test sets, indicating over-training. Looking at the number of active (non-zero weighted) features contained in the syntactic models in Figure 6.5, we see that adding syntactic features to our model increases the model size, and also that models SEQ-C and HEAD have the most features. We posit that these models are learning features that explain the development data well, but do not generalise to unseen data.

Overfitting is exacerbated by the lack of parses from which to extract features. This is because we do not apply any sentence level thresholding, meaning we are unable to parse every sentence in all the n-best lists, although every n-best list contained at least one parse. For the MT0203 training set, only 87.3% of the parses had a parse. This means that the large feature spaces demonstrated by Figure 6.5 was coming from a reduced example set. For the test sets, between 80.7% and 82.6% of the sentences had a parse, limiting the ability of the syntactic features to aid in improving the translation. The combination of a large feature space, over fewer training samples, leads to poor performance when using sparse features extracted from parsers.

Table 6.8 presents the individual n-gram precisions for our best syntactic models in comparison to the n-gram only DLM. There is a degrade in relative unigram precision on all three sets, but we see an increase in bigram, trigram and 4-gram precisions, indicating our syntactic features resolve some word reordering problems.

6.6 Conclusions

To conclude, the aim in this chapter was to gain a greater understanding for the potential of a variety of different syntactic features for reranking translation hypothesis. Turning to the main research question asked in this chapter:

RQ 3. Can a generative syntactic parser discriminate between human produced and machine output language? Are they useful in an SMT setting for reranking output, where global information may prove beneficial to translation quality?

We empirically demonstrated that a state of the art parser can differentiate between human produced and machine output language, better then standard n-gram LM. This ability improves for both a parser and n-gram LM when normalising the output scores by the respective sentence lengths. An extensive empirical analysis of different rerankers on three different test sets, found that the use of certain syntactic features within a perceptron reranker could lead to small but significant and consistent gains in translation quality. In particular, we found that the use of features extracted from a context-insensitive POS tagger lead outperformed a CRF tagger on two out of the three test sets.

This general research question gave rise to the following subquestions.

RQ 3a. Given that syntactic models are trained only on examples of good text, and are optimised towards parse retrieval as opposed to LM probability, to what extent can they be used for differentiating between human and machine text, and

more importantly, to what extent can they differentiate between different machine produced texts?

We compared a state of the art parser against a standard LM in two different settings. In the first, we compared the ability of the parser and language model to differentiate between machine output and human produced text. We have shown that, using the score output for a parser as metric of the fluency of the input sentence, a parser could differentiate between human and machine text between 54.85% and 63.37% of the time when the scores were normalised, in comparison to only 23.08% and 30.78% for the language model. However, when examining to what extent the scores output by both a standard language model operating over the surface tokens and full syntactic parser correlate with the smoothed BLEU scores, representing the varying degrees of fluency of machine output, we found no significant correlation between these rankings.

RQ 3b. As input to our machine learning algorithms for reranking, we extract and compute many different feature types from differing syntactic models. What features help most, and from which toolkit? Does the expressiveness of a full lexicalised parser outweigh the computational disadvantages, for example over a POS tagger?

We have shown that deep features, which are used with the intention of creating more generalisable models, do not help within the perceptron reranking setting as they overfit the training data, leading to problems with robustness of results over different test sets. Even when combined with shallow features, we were unable to see the significant improvements demonstrated in the ASR field. In addition to conducting a thorough analysis of both deep and shallow features, i.e features extracted from parsers and POS taggers, we proposed a simple, non-context aware POS tagger that overcomes problems encountered when using full parse tree analyses that are generated using standard syntactic toolkits. Extensive experiments of our syntactic models demonstrate significant BLEU improvements over non-reranked output and lexical only reranked models.

In the next chapter, we turn from looking at a post-processing reranking step to a pre-processing segmentation component. Instead of examining the use of syntactic analyses for improving the fluency of a translation, we look at the exploiting morphological analyses, via the use of novel, high-order HMMs, for segmenting words in the source-side of an SMT system to help reduce sparsity and improve the statistics used for computing the phrase translation probabilities.

Chapter 7

Source-side Morphological Analysis

In the previous chapter, we went beyond surface lexical features to use deeper syntax for reranking translation hypotheses for a given foreign source sentence. In this chapter, we examine the task of segmenting morphologically rich input sentences to achieve better translation quality. Morphologically rich languages pose a challenge for SMT systems; the extensive use of inflection, derivation and composition leads to a large vocabulary. This in turn leads to sparsity problems, and a lack of sufficient data for estimating translation models. We therefor examine the utility of segmenting the source input into smaller units of morphemes using high-order HMMs. Specifically, the research question we ask in this chapter is:

RQ 4. Can high-order HMMs be used to recover a partition of the source side of a parallel corpora that reduces sparsity and leads to improved translation quality?

The higher the order (in other words, the more context is used), or the larger the latent vocabulary space is, the more intractable the dynamic programes become. Intractable, in this setting, means that running standard inference algorithms take too long for computational reasons. Therefore, typically, approximate techniques are used. We begin this chapter by presenting a method that can do *exact* optimisation and sampling from high-order HMMs, which are generally handled by approximation techniques. This brings us to our first subquestion of this chapter:

RQ 4 a. HMMs that have a large latent vocabulary, or use a high amount of context, are often intractable. Can we create high-order HMMs that are tractable given large latent vocabulary spaces, or use large amounts of context, and are still exact?

Motivated by adaptive rejection sampling and heuristic search, we propose a strategy based on sequentially refining a lower-order language model that is an upper bound on the true model we wish to decode and sample from. This allows us to build tractable variable-order HMMs. To compute the scores we require at run-time efficiently, the ARPA format for language models is extended to enable an efficient use of the *max-backoff* quantities required to compute the upper bound. We evaluate our approach on two problems: a SMS retrieval task and a POS tagging experiment using 5-gram models. Results show that the same approach can be used for exact optimisation and sampling, while explicitly constructing only a fraction of the total implicit state-space.

Having defined and validated high-order HMMs, we then turn to the task of applying them to the task of segmenting Finnish. Finnish is a highly agglutinative language with a rich morphology, leading to well documented problems for numerous NLP tasks. Using labelled data provided by the Morpho Challenge project [110], we learn HMMs of differing orders that are then used for segmenting the source side of the Finnish-English data sets. By segmenting the words into smaller units composed of their stems and affixes, the hope is that words previously unseen in the training data are segmented into smaller units which are seen. These smaller segments can then be translated during the development and test stage, and will also hopefully lead to more reliable statistics at the initial word alignment phrase. Our next subquestion of this chapter is then:

RQ 4 b. Does the inclusion of the additional context, contained in the higher-order models, lead to improved translation quality when segmenting Finnish? To what extent do the application of the different models leads to a reduction in sparsity problems and model learning?

Finally, it has been demonstrated in the literature that while a single segmentation strategy in itself may not lead to improved translation quality, the combination of various different segmentations does [55]. We therefore examine the combination of different n-best lists, concatenated from the output of SMT systems fed either the original, unaltered source text, or fed segmented text given different HMM orders. This therefore brings us to our last research question of this chapter:

RQ 4 c. Can improvements be made that exploit numerous different segmentation of the same input? What methods can we use do this?

We examine the use of minimum bayes risk (MBR) for reranking the output of a single system and n-best lists which are concatenated from multiple different systems. We also compare our approach of using high-order HMMs with an unsupervised segmentor, Morfessor, developed specifically for the Finnish language [54].

To summarise, in answering these research questions, we present and experimentally validate a method for doing exact inference of high-order HMMs; and we examine the application of various segmentation approaches for segmenting the source-side of a Finnish-English bitext.

The contributions of this chapter include:

- The presentation of a method for doing exact decoding and sampling from high-order HMMs.
- An empirical demonstration of their validity on two different tasks, specifically chosen so we can present results when working with a high-vocabulary or high-order HMM.
- A comparison of different approaches for segmenting the source-side of Finnish-English SMT system.

The remainder of the chapter is structured as follows. In Section 7.1 we present related work that examines the problem of source-side segmentation, and other distinct approaches that have tacked exact inference in a wide variety of NLP settings. We describe our joint method for inference from high order HMMs in Section 7.2, and demonstrate its applicability on two HMMs tasks in Sections 7.3.1 and 7.3.2. Having demonstrated the validity of the approach on the two tasks, we turn to examining the application of high-order HMMs for source-side segmentation, specifically with the aim of reducing sparsity. We describe the experimental methodology in Section 7.4.2, and present results in Section 7.4.3. We then conclude and make suggestions for future work in Section 7.5.

7.1 Related Work

In this chapter we present a method for doing exact inference, that is exact decoding and sampling, from HMMs. Exact inference for a wide range of NLP tasks is becoming an increasingly attractive area of research. The problem in working with rich, complex models is that they can often lead to intractable, dynamic programmes. One paradigm that has become increasingly used in the NLP field is the coarse-to-fine approach [148]. Originally popular in the computer-vision field [68, 119], the idea is to

"construct simpler approximations thereof and use those to guide the learning or inference procedures" [148]. Often, this involves multiple passes, where each additional pass uses more complex representations only on regions of the search space where they are required. The underlying intuition is that, even with a coarse representation, certain regions are relatively unambiguous, and do not need a full, complex, representation for the purposes of inference. In NLP, the generic coarse-to-fine approach has been applied to syntactic parsing [149–151] and SMT [152]. A key difference between our work, and that of previous coarse-to-fine research, is that we propose an exact algorithm for the dual inference problems of optimisation and sampling.

Alternatively, a separate approach involves decomposing the original decoding problem using Lagrangian relaxation into subproblems that are easier to solve. Again, constraints are incrementally added until a solution, not necessarily exact, is found. Such approaches have been applied to SMT [39, 164], and again, parsing [108]. A problem with dual-decomposition is two fold; it can not be used for sampling, meaning it has a more limited application when used for learning, and second, unlike our algorithm, it is not guaranteed to find the true optimum solution.

Examining inference, in particular decoding, from HMMs with large latent vocabulary sets is not new. An algorithm similar to the one presented in this chapter has been proposed in an image processing context [90]. However, no connections to sampling were drawn. Felzenszwalb et al. [66] proposed a novel algorithm based on embedding hidden states in a grid space and than applying distance transform techniques. This method only works with certain distributions not often used in NLP. Alternatively, Mozes et al. [128] proposed a compression based approach that exploits occurrences of subsequences in the input for DNA sequencing. It is not clear to what extent this approach will be applicable to NLP tasks where repetitions are less common.

More recently, there is the work of Kaji et al. [89] and Huang et al. [82], who propose an algorithm based on staggered decoding, that limits the set of latent labels during optimisation. Differences between our algorithm and that of Kaji et al. [89] include the motivation of our work from the use of upper-bounds in rejection-sampling algorithm, instead of restricting the event space itself, and it is not clear if their approach can also be used for sampling.

Aside from the exact algorithms discussed, approximate decoding algorithms such as beam search have been investigated. Usually, approximate algorithms have the advantage of speed over exact algorithms, with the trade-off being unbounded error-rates and hyper-parameter optimisation. An easiest-first deterministic decoding algorithm was proposed in [193], and a parameter tying approach was presented for HMMs in [176], and in [48, 86] for CRFs.

Towards the end of the chapter we examine the combination of a segmented and non-segmented system. Reminiscent of bagging methods in machine learning [23],

recent research has demonstrated improvements in such approaches over any single method. For example, Zhang et al. [211] combine dictionary and CRF-based approaches for Chinese word segmentation, and Dyer et al. [63] use numerous different segmentations in a lattice translation based framework. This is opposite to the approach used in this chapter, where multiple segmentations are combined in a post-processing setting. Closer to the n-best system combination approach we use, de Gispert et al. [55] apply different morphological analyses to Arabic-English and Finnish-English systems, and demonstrate improvements when combining n-best lists using hypotheses from a system which, individually, performs worse than a non-analysed baseline. Similarly, Nakov et al. [132] demonstrate improvements using seven different segmented systems, and combining the output using machine-learning based reranking techniques.

Recently, Paul et al. [147] propose an unsupervised method, not reliant on any source-language specific morphological analysers, that bootstraps word alignment and segmentation steps for Chinese. A model that more tightly aligns the segmentation used with the final translation quality of the SMT system has been proposed in [133]. Interestingly, their method does not require any morphological information, though can integrate the use of language specific toolkits via the priors of the dirchlet distribution used (otherwise uniform, or uninformative priors, are used when such tools are not available). While their approach is approximate, in that it uses a Gibbs sampler to learn the best segmentation with respect to BLEU, our approach differs in that we do not learn, but rather decode the best segmentation according to high-order HMMs. Using our high-order, exact HMMs, to learn a segmentation against BLEU would be an interesting direction for future work.

Simpler approaches for unsupervised segmentation of text include the approach of Koehn and Knight [104], who examine the task of segmenting compounded nouns in German. Their approach assumes that the individual tokens that make up the words to be segmented have been seen individually. This is acceptable for the noun-decompounding task where this holds true, but inappropriate for the highly agglutinative languages such as Finnish, where it does not.

While source-side segmentation, either rooted in some form of linguistic analysis or generic machine learning algorithms, attempt to reduce OOV rates while at the same time not introducing additional ambiguity [147], other approaches to source-side prepossessing involve reordering words in the source-text to help alleviate the reordering problem for certain languages, e.g for languages that are or exhibit verb-final placements such as German or Dutch. For example, Khalilov and Fonollosa [92], Khalilov and Simaan [93], Khalilov et al. [94] all use syntax to reorder words in the source sentences, making the reordering aspect of the translation task easier.[1]

[1] Recall as described in Section 2.2.3 that to reduce the search space, limits on the reordering jumps made

Otherwise, syntax has been used in factored translation models are proposed in [103], where each source word is labelled with additional information. While the additional information does not have be syntactic or morphological in nature, it usually is (e.g., see [17]). This approach enriches the existing segmentation, and does not attempt to find a better representation of the source text, and as such can be seen as complimentary to much of the work discussed in this section, but not an answer to the segmentation problem itself.

7.2 Adaptive Rejection Sampling for High-Order HMMs

In this section we introduce our algorithm that allows us to do inference over high-order HMMs. We begin in Section 7.2.1 by defining the notation which we use throughout this chapter, then continue to give a description of our algorithm in Sections 7.2.2 and 7.2.3. We then describe the nested upper-bounds that are central to our algorithm in Section 7.2.4, and then finally give a formal description of our algorithm in Section 7.2.5. We then finish this section with a description of the method we use for computing the upper-bounds we use in an efficient manner in Section 7.2.6.

7.2.1 Notation

Let $x = \{x_1, x_2, \dots x_\ell\}$ be a given hidden state sequence (e.g., each x_i is an English word) which takes values in $\mathcal{X} = \{1, \cdots, N\}^\ell$ where ℓ is the length of the sequence and N is the number of latent symbols. Subsequences $(x_a, x_{a+1}, \dots, x_b)$ are denoted by x_a^b, where $1 \leq a \leq b \leq \ell$. Let $o = \{o_1, o_2, \dots o_\ell\}$ be the set of observations associated to these words (e.g., o_i is an acoustic realisation of x_i). The notations p, q and q' refer to unnormalised densities, i.e. non-negative measures on $\mathcal{X}$. Since only discrete spaces are considered, we use for short $p(x) = p(\{x\})$. When the context is not ambiguous, sampling according to p means sampling according to the distribution with density $\bar{p}(x) = \frac{p(x)}{p(\mathcal{X})}$, where $p(\mathcal{X}) = \int_{\mathcal{X}} p(x)dx$ is the total mass of the unnormalised distribution p.

7.2.2 Sampling

In the sampling scenario, the objective is to sample a sequence with density $\bar{p}(x)$ proportional to $p(x) = p_{\text{lm}}(x)p_{\text{obs}}(o|x)$, where p_{lm} is the probability of the sequence x under a n-gram model and $p_{\text{obs}}(o|x)$ is the probability of observing the noisy sequence

would limit the moving of sentence ending verbs to the correct location in the translation hypothesis.

o given that the correct/latent sequence is x. Assuming the observations depend only on the current state, this probability becomes

$$p(x) \quad = \quad \prod_{i=1}^{\ell} p_{\text{lm}}(x_i|x_{i-n+1}^{i-1}) p_{\text{obs}}(o_i|x_i) \ . \tag{7.1}$$

To find the most likely sequence given an observation, or to sample sequences from Equation 7.1, standard dynamic programming techniques are used [156, 173] by expanding the state space at each position. However, as the transition order n increases, or the number of latent tokens N that can emit to each observation o_l increases, the dynamic programming approach becomes intractable, as the number of operations increases exponentially in the order of $\mathcal{O}(\ell N^n)$.

If one can find a proposal distribution q that is an upper bound of p — i.e such that $q(x) \geq p(x)$ for all sequences $x \in \mathcal{X}$ — and which it is easy to sample from, the standard *rejection sampling* algorithm can be used:

1. Sample $x \sim q/q(\mathcal{X})$, with $q(\mathcal{X}) = \int_{\mathcal{X}} q(x)dx$;
2. Accept x with probability $p(x)/q(x)$, otherwise reject x;

To obtain multiple samples, the algorithm is repeated several times. However, for simple bounds, the average acceptance rate, which is equal to $p(\mathcal{X})/q(\mathcal{X})$, can be so large that rejection sampling is not practical. In *adaptive rejection sampling* (ARS), the initial bound q is incrementally improved based on the values of the rejected elements. While often based on log-concave distributions which are easy to bound, ARS is valid for any type of bound, and in particular can be applied to the upper bounds on n-gram models introduced by [90] in the context of optimisation. When a sample is rejected, our algorithm assumes that a small set of refined proposals is available, say $q'_1, \ldots, q'_m$, where m is a small integer value. These refinements are *improved* versions of the current proposal q in the sense that they still upper-bound the target distribution p, but their mass is strictly smaller than the mass of q, i.e. $q'(\mathcal{X}) < q(\mathcal{X})$. Thus, each such refinement q', while still being optimistic relative to the target distribution p, has higher average acceptance rate than the previous upper bound q. A bound on the n-gram LM will be presented in Section 7.2.4.

7.2.3 Optimisation

In the case of optimisation, the objective is to find the sequence maximising $p(x)$. Viterbi on high-order HMMs is intractable, but we have access to an upper bound q for which Viterbi is tractable. Sampling from q is then replaced by finding the maximum

point x of q, looking at the ratio $r(x) = p(x)/q(x)$, and accepting x if this ratio is equal to 1, otherwise refining q into q' exactly as in the sampling case. This technique is able to find the exact maximum of p, similarly to standard heuristic search algorithms based on optimistic bounds. We stop the process when q and p agree at the value maximising q which implies that we have found the global maximum.

7.2.4 Upper Bounds for N-gram Models

To apply ARS on the target density given by Equation 7.1 we need to define a random sequence of proposal distributions $\{q^{(t)}\}_{t=1}^{\infty}$ such that $q^{(t)}(x) \geq p(x), \forall x \in \mathcal{X}, \forall t \in \{0, 1, \cdots\}$. Each n-gram $x_{i-n+1}, ..., x_i$ in the hidden layer contributes an n-th order factor $w_n(x_i|x_{i-n+1}^{i-1}) \equiv p_{\text{lm}}(x_i|x_{i-n+1}^{i-1})p_{\text{obs}}(o_i|x_i)$. The key idea is that these n-th order factors can be upper bounded by factors of order $n-k$ by maximising over the head (i.e. prefix) of the context, as if part of the context was "forgotten."

Formally, we define the *max-backoff weights* as:

$$w_{n-k}(x_i|x_{i-n+1+k}^{i-1}) \equiv \max_{x_{i-n+1}^{i-n+k}} w_n(x_i|x_{i-n+1}^{i-1}). \tag{7.2}$$

By construction, the max-backoff weights w_{n-k} are factors of order $n-k$ and can be used as surrogates to the original n-th order factors of Equation 7.1, leading to a nested sequence of upper bounds until reaching binary or unary factors:

$$\begin{aligned}
p(x) &= \Pi_{i=1}^{\ell} w_n(x_i|x_{i-n+1}^{i-1}) && (7.3)\\
&\leq \Pi_{i=1}^{\ell} w_{n-1}(x_i|x_{i-n+2}^{i-1}) && (7.4)\\
&\cdots \\
&\leq \Pi_{i=1}^{\ell} w_2(x_i|x_{i-1}) && (7.5)\\
&\leq \Pi_{i=1}^{\ell} w_1(x_i) := q^{(0)}(x) \ . && (7.6)
\end{aligned}$$

Now, one can see that the loosest bound (7.6) based on unigrams corresponds to a completely factorised distribution which is straightforward to sample and optimise. The bigram bound (7.5) corresponds to a standard HMM probability that can be efficiently decoded (using Viterbi algorithm) and sampled (using backward filtering-forward sampling). [2] In the context of ARS, our initial proposal $q^{(0)}(x)$ is set to the unigram

[2] Backward filtering-forward sampling [173] refers to the process of running the Forward algorithm [156], which creates a lattice of forward probabilities that contains the probability of ending in a latent state at a specific time t, given the subsequence of previous observations o_1^t, for all the previous latent sub-sequences x_1^{t-1}, and then recursively moving backwards, sampling a latent state based on these probabilities.

bound (7.6). The bound is then incrementally improved by adaptively refining the max-backoff weights based on the values of the rejected samples. Here, a refinement refers to the increase of the order of *some* of the max-backoff weights in the current proposal (thus most refinements consist of n-grams with heterogeneous max-backoff orders, not only those shown in Equations 7.3-7.6). This operation tends to tighten the bound and therefore increase the acceptance probability of the rejection sampler, at the price of a higher sampling complexity. It is possible to conceive of several different ways of choosing the weights to refine; in Section 7.2.5 different refinement strategies will be discussed, but the main technical difficulty remains in the efficient exact optimisation and sampling of a HMM with n-grams of variable orders. The construction of the refinement sequence $\{q^{(t)}\}_{t\geq 0}$ can be easily explained and implemented through a Weighted Finite State Automaton (WFSA) referred as a *q-automaton*, as illustrated in the following example.

Example Refinement

We now give a high-level description of the refinement process to give a better intuition of our method. In Figure 7.1(a), we show a WFSA representing the initial proposal $q^{(0)}$ corresponding to an example with an acoustic realisation of the sequence of words (`the`, `two`, `dogs`, `barked`). The weights on edges of this q-automaton correspond to the unigram max-backoffs, so that the total weight corresponds to Equation (7.6).

Considering sampling, we suppose that the first sample from $q^{(0)}$ produces the latent sequence $x_1 =$ (`the, two, dog, barked`). This sequence is marked with bold edges in the drawing. Now, computing the ratio $p(x_1)/q^{(0)}(x_1)$ gives a result much below 1, because from the viewpoint of the model we care about, that is the full model p, the trigram `(the two dog)` is very unlikely. In other words, the ratio $w_3(\texttt{dog}|\texttt{the two})/w_1(\texttt{dog})$ (and, in fact, already the ratio $w_2(\texttt{dog}|\texttt{two})/w_1(\texttt{dog})$) is very low. Thus, with high probability, x_1 is rejected. As defined by the generic Adaptive Rejection Sampling algorithm (see the last paragraph of Section 7.2.2 for a reminder), when this is the case, we produce a refined proposal $q^{(1)}$, represented by the WFSA in Fig. 7.1(b). The new $q^{(1)}$ distribution, that takes into account the weight $w_2(\texttt{dog}|\texttt{two})$ by adding a node (node 6) for the context `two`, is slightly more realistic.

We now sample trials from $q^{(1)}$, which tends to avoid producing `dog` in the context of `two`. If future samples are accepted, then we do not further refine $q^{(1)}$. If, however, a sample is rejected, then the refinement process continues until we start observing that the acceptance rate reaches a fixed threshold value.

The optimisation scenario is similar. Suppose that with $q^{(0)}$, the maximum is x_1, then we observe that $p(x_1)$ is lower than $q^{(0)}(x_1)$, reject suboptimal x_1 and refine $q^{(0)}$ into $q^{(1)}$. We continue this process until the maximum event x' from some q^i is

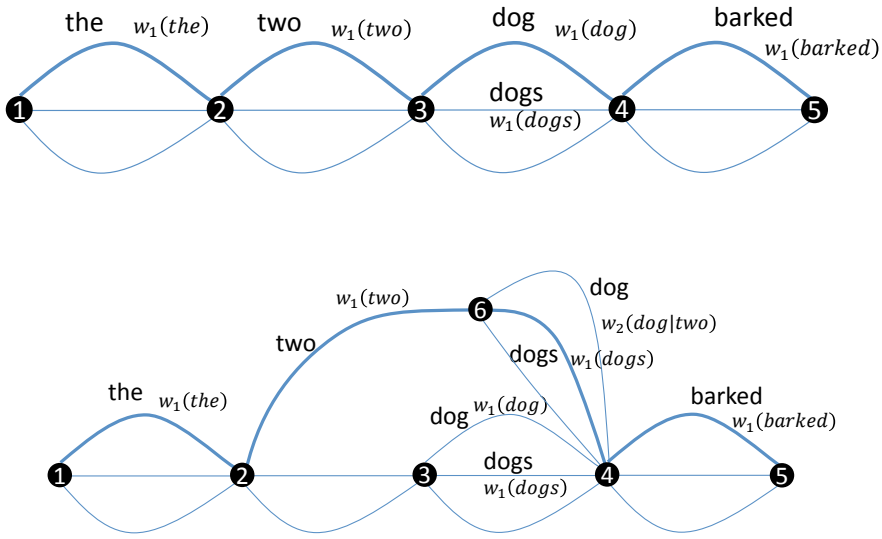


Figure 7.1: An example of an initial q-automaton (a), and the refined q-automaton (b) Each state corresponds to a context (only state 6 has a non-empty context) and each edge represents the emission of a symbol. Thick edges are representing the path for the sampling/decoding of `two dog(s) barked`, thin edges corresponding to alternative symbols. By construction, $w_1(\texttt{dog}) \geq w_2(\texttt{dog|two})$ so that the total weight of (b) is smaller than the total weight of (a).

the same as the maximum event from p. Thus, whereas in the sampling scenario, we would only refine $q^{(i)}$ into $q^{(i+1)}$ when the acceptance rate was below some predefined threshold, in the optimisation setting, we continue infill the weights for the maximum event according to some $q^(i)$ and p are the same.

7.2.5 Algorithm

Having just previously given a more concrete, descriptive, example of our algorithm in both the sampling and optimisation scenarios, we now formally detail the algorithm and procedure for updating a q-automaton with a max-backoff of longer context.

Algorithm 2 ARS for HMM algorithm.

1: **while** not Stop(h) **do**
2: **if** Optimisation **then**
3: Viterbi $x \sim q$
4: **else**
5: Sample $x \sim q$
6: $r \leftarrow p(x)/q(x)$
7: Accept-or-Reject(x, r)
8: Update(h, x)
9: **if** Rejected(x) **then**
10: **for all** $i \in \{2, \cdots, \ell\}$ **do**
11: $q \leftarrow$ UpdateHMM (q, x, i)
12: **return** q along with accepted x's in h

Algorithm 2 gives the pseudo-code of the joint sampling/optimisation strategy. On line 1, h represents the history of all trials so far. The stopping criterion for decoding is dependent on whether the last trial in the history has been accepted, and for sampling whether the ratio of accepted trials relative to all trials exceeds a certain threshold. The WFSA is initialised so that all transitions only take into account the $w_1(x_i)$ max-backoffs, i.e. the initial optimistic-bound ignores all contexts. Then depending on whether we are sampling or decoding, in lines 2-5, we draw an event from our automaton using either the Viterbi algorithm or Forward-Backward sampling. If the sequence is rejected at line 7, then the q-automaton is updated in lines 10 and 11. This is done by expanding all the factors involved in the sampling/decoding of the rejected sequence x to a higher order. That is, while sampling or decoding the automaton using the current proposal $q^{(t)}$, the contexts used in the path of the rejected sequence are replaced with higher order contexts in the new refined proposal $q^{t+1}(x)$.

The update process of the q-automaton represented as a WFSA is described in Algorithm 3. This procedure guarantees that a lower, more realistic weight is used in *all* paths containing the n-gram x_{i-n+1}^{i} while decoding/sampling the q-automaton, where n is the order at which x_{i-n+1}^{i} has been expanded so far. The algorithm takes as input a max-backoff function, and refines the WFSA such that any paths that include this n-gram have a smaller weight thanks to the fact that higher-order max-backoff have automatically smaller weights.

The algorithm requires the following functions:

- ORDER$_i(x)$ returns the order at which the n-gram has been expanded so far at position i.
- S_i returns the states at a position i.

Algorithm 3 UpdateHMM

Input: A triplet (q, x, i) where q is a WFSA, x is a sequence determining a unique path in the WFSA and i is a position at which a refinement must be done.

1: n :=ORDER$_i(x_1^i) + 1$ #implies $x_{i-n+2}^{i-1} \in S_{i-1}$
2: **if** $x_{i-n+1}^{i-1} \notin S_{i-1}$ **then**
3: CREATE-STATE($x_{i-n+1}^{i-1}, i-1$)
4: #move incoming edges, keeping WFSA deterministic
5: **for all** s $\in$ SUF$_{i-2}(x_{i-n+1}^{i-2})$ **do**
6: $e :=$ EDGE(s, x_{i-1})
7: MOVE-EDGE-END(e,x_{i-n+1}^{i-1})
8: #create outgoing edges
9: **for all** $(s, l, \omega) \in T_i(x_{i-n+2}^{i-1})$ **do**
10: CREATE-EDGE(x_{i-n+1}^{i-1},s,l,ω)
11: #update weights
12: **for all** s $\in$ SUF$_{i-1}(x_{i-n+1}^{i-1})$ **do**
13: weight of EDGE$(s, x_i) := w_n(x_i|x_{i-n+1}^{i-1})$
14: **return**

- $T_i(s)$ returns end states, labels and weights of all edges that originate from this state.
- SUF$_i(x)$ returns the states at i which have a suffix matching the given context x. For empty contexts, all states at i are returned.
- EDGE(s, l) returns the edge which originates from s and has label l. Deterministic WFSA, such as those used here, can only have a single transition with a label l leaving from a state s.
- CREATE-STATE(s, i) creates a state with name s at position i.
- CREATE-EDGE(s_1, s_2, l, ω) creates an edge (s_1, s_2) between s_1 and s_2 with weight ω and label l.
- MOVE-EDGE-END(e, s) sets the end of edge e to be the state s, keeping the same starting state, weight and label.

At line 1, the expansion of the current n-gram is increased by one so that we only need to expand contexts of size $n-2$. Line 2 checks whether the context state exists. If it does not, it is created at lines 3-10. Finally, the weight of the edges that could be involved in the decoding of this n-gram are updated to a smaller value given by a higher-order max-backoff weight.

The creation of a new state in lines 3-10 is straightforward: At lines 5-7, incoming edges are moved from states at position $i-2$ with a matching context to the newly created edge. At lines 9-10 edges heading out of the context state are created. They are simply copied over from all edges that originate from the suffix of the context state, as we know these will be legitimate transitions (i.e we will always transition to a state of the same order or lower).

Note that we can derive many other variants of Algorithm 3 which also guarantee a smaller total weight for the q-automaton. We chose to present this version because it is relatively simple to implement, and numerical experiments comparing different refinement approaches (including replacing the max-backoffs with the highest-possible context, or picking a single "culprit" to refine) showed that this approach gives a good trade-off between model complexity and running time.

7.2.6 Computing Max-Backoff Factors

An interesting property of the max-backoff weights is that they can be computed recursively; taking a Trigram LM as an example, we have:

$$\begin{aligned} w_1(x_i) &= \max_{x_i-1} w_2(x_i|x_{i-1}) \\ w_2(x_i|x_{i-1}) &= \max_{x_i-2} w_3(x_i|x_{i-2}^{i-1}) \\ w_3(x_i|x_{i-2}^{i-1}) &= p(x_i|x_{i-2}^{i-1})\, p(o_i|x_i). \end{aligned}$$

The final $w_3(x_i|x_{i-2}^{i-1})$ upper bound function is simply equal to the true probability (multiplied by the conditional probability of the observation), as any extra context is discarded by the trigram language model. It's easy to see that as we refine $q^{(t)}$ by replacing existing max-backoff weights with more specific contexts, the $q^{(t)}$ tends to p at t tends to infinity.

In the HMM formulation, we need to be able to efficiently compute at run-time the max-backoffs $w_1(\texttt{the})$, $w_2(\texttt{dog}|\texttt{the})$, $\cdots$, taking into account smoothing. To do so, we present a novel method for converting language models in the standard ARPA format used by common toolkits such as [182] into a format that we can use. The ARPA file format is a table T composed of three columns: (1) an n-gram which has been observed in the training corpus, (2) the log of the conditional probability of the last word in the n-gram given the previous words (log f$(\cdot)$), and (3) a backoff weight (bow$(\cdot)$) used when unseen $n-$grams 'backoff' to this n-gram.[3]

[3]See `http://www.speech.sri.com/projects/srilm/manpages/ngram-format.5.html`, last accessed on the 1st of March 2012, for further details.

The probability of any n-gram x_{i-n}^{i} (in the previous sense, i.e. writing $p(x_{i-n}^{i})$ for $p(x_i|x_{i-n}^{i-1})$) is then computed recursively as:

$$p(x_{i-n}^{i}) = \begin{cases} \mathrm{f}(x_{i-n}^{i}) & \text{if } x_{i-n}^{i} \in T \\ \mathrm{bow}(x_{i-n}^{i-1})\, p(x_{i-n+1}^{i}) & \text{otherwise.} \end{cases} \tag{7.7}$$

Here, it is understood that if x_{i-n}^{i-1} is in T, then its bow$(\cdot)$ is read from the table, otherwise it is taken to be 1. Different smoothing techniques will lead to different calculations of f(x_{i-n}^{i}) and bow(x_{i-n}^{i-1}), however both backoff and linear-interpolation methods can be formulated using Equation 7.7.

Starting from the ARPA format, we pre-compute a new table MAX-ARPA, which has the same lines as ARPA, each corresponding to an n-gram x_{i-n}^{i} observed in the corpus, and the same f and bow, but with two additional columns: (4) a max log probability (log mf(x_{i-n}^{i})), which is equal to the maximum log probability over all the n-grams extending the context of x_{i-n}^{i}, i.e. which have x_{i-n}^{i} as a suffix; (5) a "max backoff" weight (mbow(x_{i-n}^{i})), which is a number used for computing the max log probability of an n-gram not listed in the table. From the MAX-ARPA table, the max probability w of any n-gram x_{i-n}^{i}, i.e the maximum of $p(x_{i-n-k}^{i})$ over all n-grams extending the context of x_{i-n}^{i}, can then be computed recursively (again very quickly) as:

$$w(x_{i-n}^{i}) = \begin{cases} \mathrm{mf}(x_{i-n}^{i}) & \text{if } x_{i-n}^{i} \in T \\ \mathrm{mbow}(x_{i-n}^{i-1})\, p(x_{i-n}^{i}) & \text{otherwise.} \end{cases} \tag{7.8}$$

Here, if x_{i-n}^{i-1} is in T, then its mbow$(\cdot)$ is read from the table, otherwise it is taken to be 1. Also note that the procedure calls p, which is computed as described in Equation 7.7. Note, that in this description of the MAX-ARPA table, we have ignored the contribution of the observation $p(o_i|x_i)$. This is due to the fact the emission distribution is a constant factor over the different max-backoffs for the same x_i, and so does not alter the computation of the table.

7.3 Validation Experiments

In this section we empirically evaluate our joint, exact decoder and sampler on two tasks; SMS retrieval (Section 7.3.1), and supervised POS tagging (Section 7.3.2). While the first task is slightly superficial; it allows us to look at the application of our algorithm to HMMs with high latent vocabulary spaces. The POS task is a more standard, well known NLPs task, but the latent vocabulary space is limited. So, for the second task, we will examine HMMs with high orders.

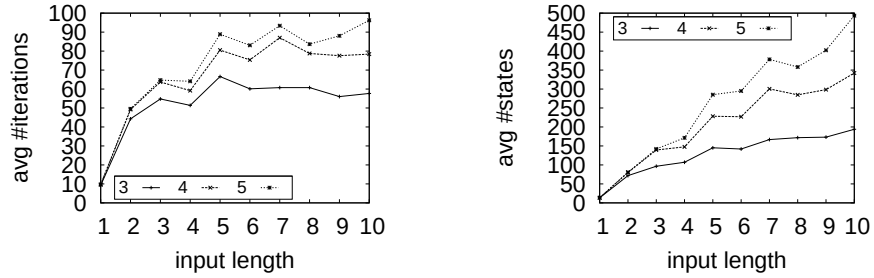


Figure 7.2: On the left we report the average # of iterations taken to decode given different LMs over input sentences of different lengths, and on the right we show the average # of states in the final q-automaton once decoding is completed.

7.3.1 SMS Retrieval

We evaluate our approach on an SMS message retrieval task. A latent variable $x \in \{1, \cdots, N\}^{\ell}$ represents a sentence represented as a sequence of words: N is the number of possible words in the vocabulary and ℓ is the number of words in the sentence. Each word is converted into a sequence of numbers based on a mobile phone numeric keypad. The standard character-to-numeric function num : $\{\mathtt{a}, \mathtt{b}, \cdots, \mathtt{z}, \cdots, ?\} \rightarrow \{1, 2, \cdots, 9, 0\}$ is used. For example, the words `dog` and `fog` are represented by the sequence $(3, 6, 4)$ because num(`d`)=num(`f`)=3, num(`o`)=6 and num(`g`)=4. Hence, observed sequences are sequences of numeric strings separated by white spaces. To take into account typing errors, we assume we observe a noisy version of the correct numeric sequence $(\text{num}(x_{i1}), \cdots, \text{num}(x_{i|x_i|})$ that encodes the word x_i at the i-th position of the sentence x. The noise model is:

$$p(o_i|x_i) \propto \prod_{t=1}^{|x_i|} \frac{1}{k * d(o_{it}, \text{num}(x_{it})) + 1}, \tag{7.9}$$

where $d(a, b)$ is the physical distance between the numeric keys a and b and k is a user provided constant that controls the ambiguity in the distribution; we use 64 to obtain moderately noisy sequences.

We used the English side of the Europarl corpus [101]. The language model was trained using SRILM [182] on 90% of the sentences. On the remaining 10%, we ran-

n:	1	2	3	4	5
q:	7868	615	231	176	118

Table 7.1: The number of n-grams in our variable-order HMM.

domly selected 100 sequences for lengths 1 to 10 to obtain 1000 sequences from which we removed the ones containing numbers, obtaining a test set of size 926.

Decoding Algorithm 2 was run in the optimisation mode. In the left plot of Figure 7.2, we show the number of iterations (running Viterbi then updating q) that the different n-gram models of size 3, 4 and 5 take to do exact decoding of the test set. For a fixed sentence length, we can see that decoding with larger n-gram models leads to a sub-linear increase with regards to n in the number of iterations taken. In the right plot of Figure 7.2, we show the average number of states in our variable-order HMMs.

To demonstrate the reduced nature of our q-automaton, we show in Table 7.1 the distribution of n-grams in our final model for a specific input sentence of length 10. The number of n-grams in the full model is $\sim 3.0 \times 10^{15}$. Exact decoding here is not tractable using existing techniques. Our HMM has only 9008 n-grams in total, including 118 5-grams.

Finally, we show in Table 7.2 an example run of our algorithm in the optimisation setting for a given input. Note that the weight according to our q-automaton for the first path returned by the Viterbi algorithm is high in comparison to the true log probability according to p.

Sampling We refine our q-automaton until we reach a certain fixed cumulative acceptance rate (AR). We also compute a rate based only on the last 100 trials (AR-100), as this tends to better reflect the current acceptance rate.

In Figure 7.3a, we plot the ratio for each point sampled from q, for a single sampling run using a 5-gram model for an example input. The ratios start off at 10^{-20}, but gradually increase as we refine our HMM. After $\sim$ 500 trials, we start accepting samples from p. In Figure 7.3b, we show the respective ARs (bottom and top curves respectively), and the cumulative # of accepts (middle curve), for the same input. Because the cumulative accept ratio takes into account all trials, the final AR of 17.7% is an underestimate of the true accept ratio at the final iteration; this final accept ratio can be better estimated on the basis of the last 100 trials, for which we read AR-100 to be at around 60%.

input:		3637 843 66639 39478 *	
oracle:		does the money exist ?	
viterbi best:		does the money exist .	
Viterbi paths		$\log q(x)$	$\log p(x)$
q^1	does the money exist **)**	-0.11	-17.42
q^{50}	does the **owned** exist **.**	-11.71	-23.54
q^{100}	**ends** the money exist **.**	-12.76	-17.09
q^{150}	does **vis** money exist **.**	-13.45	-23.74
q^{170}	does the money exist **.**	-13.70	-13.70

Table 7.2: Viterbi paths given different q^t. Here, for the given input, it took 170 iterations to find the best sequence according to p, so we only show every 50th path. We note in bold face the words that differ from the viterbi-best sequence.

B:	1	10	20	30	40	50	100
time:	97.5	19.9	15.0	13.9	12.8	12.5	11.4
iter:	453	456	480	516	536	568	700

Table 7.3: In this table we show the average amount of time in seconds and the average number of iterations (iter) taken to sample sentences of length 10 given different values of B.

We note that there is a trade-off between the time needed to construct the forward probability lattice needed for sampling, and the time it takes to adapt the variable-order HMM. To resolve this, we propose to use batch-updates: making B trials from the same q-automaton, and then updating our model in one step. By doing this, we noted significant speed-ups in sampling times. In Table 7.3, we show various statistics for sampling up to AR-100 $= 20$ given different values for B. We ran this experiment using the set of sentences of length 10. A value of 1 means that we refine our automaton after each rejected trial, a value of 10 means we wait until rejecting 10 trials before updating our automaton in one step. We can see that while higher values of B lead to more iterations, as we do not need to re-compute the forward trellis needed for sampling, the time needed to reach the specific AR threshold actually decreases, from 97.5 seconds to 11.4 seconds, an 8.5% speedup. Unless explicitly stated otherwise, further experiments use a $B = 100$.

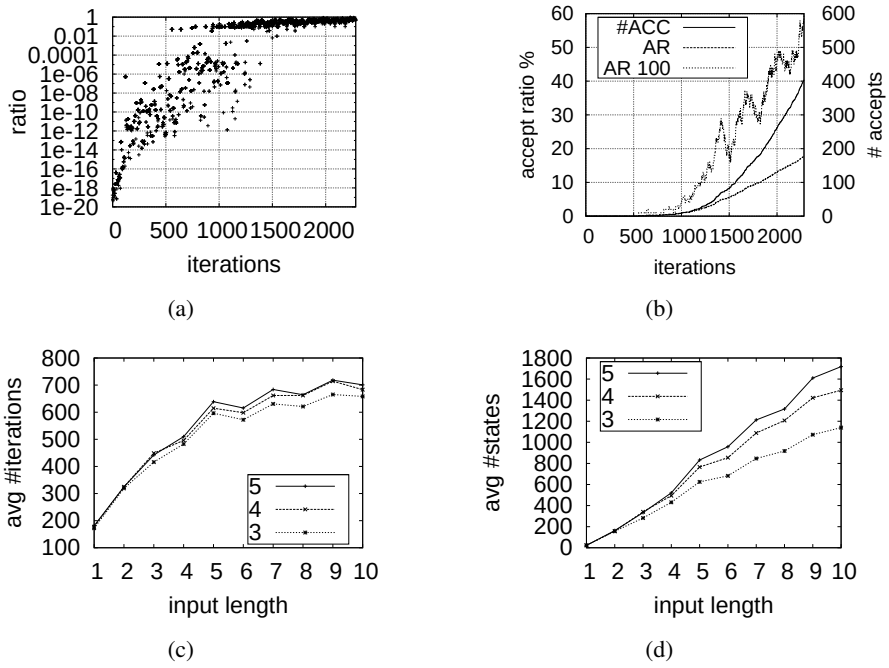


Figure 7.3: In 7.3a, we plot the ratio for each point sampled from q. In 7.3b, we plot the cumulative # of accepts (middle curve), the accept rate (bottom curve), and the accept rate based on the last 100 samples (top curve). In 7.3c, we plot the average number of iterations needed to sample up to an AR of 20% for sentences of different lengths in our test set, and in 7.3d, we show the average number of states in our HMMs for the same experiment.

We now present the full sampling results on our test set in Figure 7.3c and 7.3d, where we show the average number of iterations and states in the final models once refinements are finished (AR-100=20%) for different orders n over different lengths. We note a sub-linear increase in the average number of trials and states when moving to higher n; thus, for length=10, and for $n = 3, 4, 5$, # trials: 3-658.16, 4-683.3, 5-700.9, and # states: 3-1139.5, 4-1494.0, 5-1718.3.

Finally, we show in Table 7.4, the ranked samples drawn from an input sentence, according to a 5-gram LM. After refining our model up to AR-100 $= 20\%$, we contin-

input:			3637 843 66639 39478 *
oracle:			does the money exist ?
best:			does the money exist .
samples	#	$\log q(x)$	$\log p(x)$
does the money exist .	429	-13.70	-13.70
does the money exist **?**	211	-14.51	-14.51
does the money exist **!**	72	-15.49	-15.49
does the **moody** exist .	45	-15.70	-15.70
does the money exist **:**	25	-16.73	-16.73

Table 7.4: Top-5 ranked samples for an example input. We highlight in bold the words which are different to the Viterbi best of the model. The oracle and best are not the same for this input.

ued drawing samples until we had 1000 exact samples from p (out of $\sim$ 4.7k trials). We show the count of each sequence in the 1000 samples, and the log probability according to p for that event. We only present the top-five samples, though in total there were 90 unique sequences sampled, 50 of which were only sampled once.

7.3.2 POS Tagging

Our HMM is the same as that used in [21]; the emission probability of a word given a POS tag x_i is calculated using maximum likelihood techniques. That is, $p(o_i|x_i) = \frac{c(o_i,x_i)}{c(x_i)}$. Unseen words are handled by interpolating longer suffixes with shorter, more general suffixes. We build LMs of up to size 9.[4] We present results on the WSJ Penn Treebank corpus [121]. We use sections 0-18 to train our emission and transitions probabilities, and report results on sections 22-24.

We first present results for our decoding experiments. In Figure 7.4a we show the accuracy results of our different models on the WSJ test set. We see that the best result is achieved with the 5-gram LM giving an accuracy of 95.94%. After that, results start to drop, most likely due to over-fitting of the LM during training and an inability for the smoothing technique to correctly handle this.

[4]Theoretically there is no bound on the LM-order we use, and so could use higher order models. However, the authors noted unexplainable statistics in the models constructed of orders 10 and higher, so do not report results using these LMs.

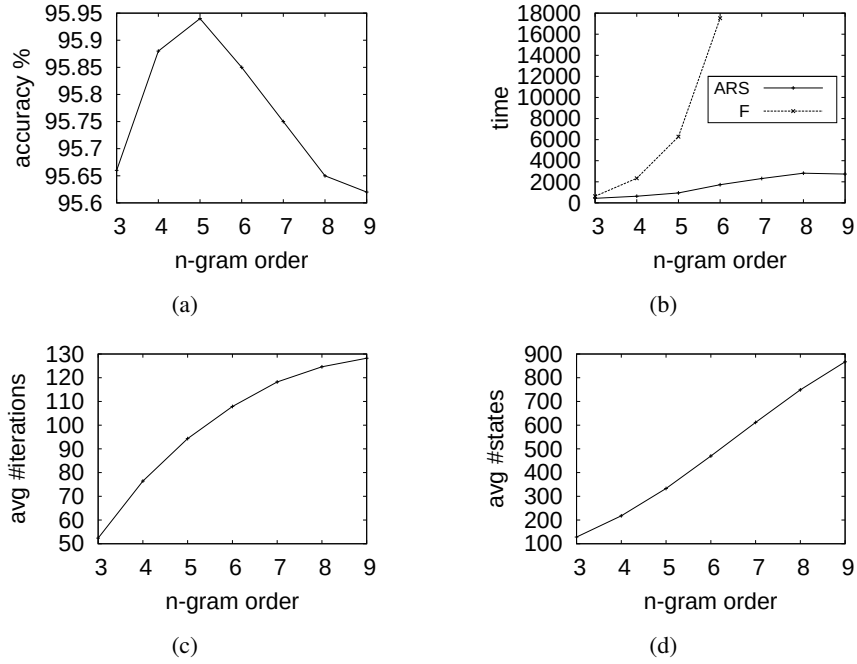


Figure 7.4: In 7.4a, we report the accuracy results given different n-gram models on the WSJ test set. In 7.4b, we show the time taken (seconds) to decode the WSJ test set given our method (ARS), and compare this to the full model (F). In 7.4c, the average number of iterations needed to sample the test set given different n-gram language models is given, and 7.4d shows the average number of states in the variable-order HMMs.

In Figure 7.4b, we compare the time it takes in seconds to decode the test set with the full model at each n-gram size; that is a WFSA with all context states and weights representing the true language model log probabilities. We can see that while increasing the n-gram model size, our method (ARS) exhibits a linear increase in decoding time, in contrast to the exponential factor exhibited when running the Viterbi algorithm over the full WFSA (F). Note for n-gram models of order 7 and higher, we could not decode the entire test set as creating the full WFSA was taking too long.

Finally in both Figures 7.4c and 7.4d, we show the average number of iterations

taken to sample from the entire test set, and the average number of states in our variable-order HMMs, with AR-100=60%. Again we note a linear increase in both figures, in contrast to the exponential nature of standard techniques applied to the full HMM.

7.4 Finnish Segmentation

So far, we have described the method to do exact inference with high-order HMMs, and have demonstrated their validity on two different natural language processing tasks. We now turn our attention to the task of Finnish morphological analysis. We examine Finnish because it has a particularly rich morphology. In particular, it is highly agglutinative, so most words are formed by joining morphemes together. We wish to use high-order HMMs to decompose the Finnish side of a Finnish-English bitext, reducing the high sparsity typically encountered; giving rise to high OOV rates and impoverished statistics that impact detrimentally on word alignment.

We now describe the models and reranker we use in Section 7.4.1, the experimental methodology in Section 7.4.2, and present results using our high-order HMM segmentors in Section 7.4.3.

7.4.1 Models

As well as proposing to use our own higher-order HMMs for segmenting the source Finnish text, we examine the use of an existing toolkit, Morfessor, for the segmentation of Finnish [54].[5] Further, we propose to examine combing the outputs from different systems for improved translation quality, based on the MBR approach. We now explain Morfessor and MBR in turn.

Morfessor

The aim of the model is learn a lexicon of morphemes given an input corpus. This corpus can, for example, be the source side of a bitext used for building an SMT system. Specifically, each word is represented as HMM with the limitation that prefixes come before stems, and stems come before suffixes. As the method is unsupervised, the objective aim is to maximise the posterior estimate of the lexicon given the corpus as:

$$\arg \max_{lexicon} p(lexicon|corpus) = p(corpus|lexicon) \cdot p(lexicon)$$

[5] Available at `http://www.cis.hut.fi/projects/morpho/`

word:	arkityylissä
decomposed word:	arki:arki_N tyyli:tyyli_N ssä:+INE
labeled sequence:	BS-a CS-r CS-r CS-k CS-i BS-t CS-y CS-y CS-l CS-i BA-s CA-s -CA-ä

Table 7.5: In this table we show a randomly chosen Finnish word, its morphological analysis, and the resulting latent sequence, which we use along with the other 999 labeled words for training our high-order HMMs.

A HMM is used to for modelling the $p(lexicon|corpus)$, so that it contains emission (the probability of the morphological type given the morpheme) and transition (the probability of the current morpheme form given the previous form) probabilities. The distribution $p(lexicon)$ operates over two distributions; the probability of a morpheme type (a simple distribution), and the probability of the meaning of a morpheme. The notion of meaning may seem grand at first, but relies on the distributional hypothesis [78] that the meaning of words and morphemes is isomorphically related to their use, which in Morfessor is measured via length, frequency, context, and intra-word perplexity features.

In our experiments, we examine different ways of using the morphological information output by the Morfessor algorithm. We try to strategies. The first, which we call SPLIT-ALL, is to keep all the information, but make the split at each prefix/stem/suffix. The second approach, STEM-ONLY, is to throw away all tokens keeping only the stem. We train Morfessor on the Finnish side of the parallel corpus.

Minimum Bayes Risk System Combination

MBR decoding is a commonly used algorithm for system combination [57, 109, 124, 145, 191]. Note that MBR decoding differs from MAP decoding, as described in Equation 2.2, in that while MAP decoding seeks the most likely output sequence, MBR seeks the output whose loss is the smallest. Practically, this notion of loss is defined using BLEU, meaning the sentence which shares the most n-grams with other sentences in the hypothesis space will be preferred above others, and not necessarily the one with the highest posterior score. MBR is defined as follows:[6]

[6]Note that the definition can also be defined with regards to taking a maximum. See for example [102].

$$\hat{e}_{mbr} = \arg\min_{e \in E} R(e) \tag{7.10}$$

$$= \arg\min_{e \in E} \sum_{e' \in E} L(e, e')p(e'|f) \tag{7.11}$$

$$= \arg\min_{e \in E} \sum_{e' \in E} 1 - BLEU(e, e')p(e'|f). \tag{7.12}$$

As described, MBR quantifies the notion of loss $L(e, e')$ using BLEU. Here, $p(e'|f)$ is taken to be the posterior score for the hypothesis e'; that is the score output by the system, and E is the hypothesis space off all possible translations given f.

Because MBR requires E^2 computations, given large lists, it can be very slow. Thus, typically, the hypothesis space E examined by MBR differs from the Maximum a posteriori (MAP) approach in that is limited to an n-best list or lattice representing the output of a single system (reranking), or the combination of multiple system outputs (combination).[7] In this chapter we work with n-best lists of size N, (limited to some top-n translations per system according to the original MAP score), we set the posterior score used in Equation 7.12 to be:

$$p(e|f) = \frac{p(e, f)}{\sum_{i=1}^{N} p(e, f)}.$$

Note that if we set the loss function to be the following:

$$L(e, e') = \begin{cases} 0 & \text{if } e = e' \\ 1 & \text{otherwise,} \end{cases}$$

then we end with the original MAP decoder; the translation with the highest posterior score will be selected as the top translation candidate.

7.4.2 Methodology

Our baseline SMT system uses the Finnish-English portions of the Europarl corpus [101]. As with other chapters in this thesis, we train a 5-gram language model with the Kneser-Ney and interpolate options. The train, development, and test portions all come from standard respective date splits of the Europarl corpus. For the development and test sets, we randomly sampled without replacements 500 and 2000 sentences respectively.

[7] This is not always the case: see [9, 10].

Model	Sentences	Tokens	Types
English side:	2,933,635	76,717,534	247,910
Baseline	2,933,635	57,173,422	923,208
MORF-SPLIT	2,933,635	85,075,250	177,215
MORF-STEM	2,933,635	57,712,982	177,200
HMM3	2,865,069	90,399,413	473,158
HMM4	2,849,615	92,694,871	435,345
HMM5	2,845,386	93,266,053	430,879
HMM6	2,844,712	93,307,806	430,755
HMM7	2,844,677	93,321,777	430,722
HMM8	2,844,643	93,335,618	430,746
HMM9	2,844,598	93,349,436	430,746

Table 7.6: In this table we present the sizes of the bitext used by each approach examined in this chapter, as well as the total number of words and the number of unique tokens in the Finnish side of each. For comparison purposes, we show statistics for the English side of the Finnish-English bitext of the baseline in the top row.

The high-order HMMs we present in Section 7.2 are supervised, so we require some labeled data to learn the latent segmentations. To this end, we use human, morphologically analysed data, which is publicly available from the Morpho Challenge project [110].[8] From these labels, we can the assign each word in the labeled sequence with a corresponding latent sequence. In this setting, the latent sequence is made up from set of all characters, each with a begin-state, continue-state, begin-affix, and continue-affix token. We give an example of the original word, it's human analysis, and the latent sequence we create in Table 7.5. In total we there are 1000 words with human annotations. We build language models of sizes between 3 and 9-grams, using the SRILM toolkit [183].

In Table 7.6, we present the size of the bitext used for the different models, along with the number of words and unique tokens in the source side of the Finnish-English bitext. For comparison, we show in the second row the same statistics for the English side of the baseline corpus. The Finnish side of the baseline has fewer words, as we would expect given it's rich morphology, yet it has over 900 thousand unique tokens, more than 3 times the number of unique words in the English portion of the bitext.

[8] See `http://research.ics.aalto.fi/events/morphochallenge2010/`, last accessed at 3/8/2012.

Model	BLEU
Baseline	23.93
MORF-SPLIT	23.08
MORF-STEM	20.81
HMM3	22.26
HMM4	21.76
HMM5	21.93
HMM6	21.69
HMM7	21.76
HMM8	21.82
HMM9	21.41

Table 7.7: In this table we show the 1-best results for the baseline system and the approaches using the Morfessor toolkit and our own HMMs.

Different sizes can be accounted for by the processing of the bi-texts using standard pre-processing scripts. Word alignment requires that none of the sentences in either sides of the bitext exceed 100 words, and that none of the sentences are more than 9 times the length of their corresponding sentence on the other side of the parallel corpus. We can see that as we increase the order of our HMM from 3 to 9-grams, the bitext decreases in size. This indicates that the use of more context leads to HMMs that segment more of the text, as more sentences are filtered out prior to the running of word-alignment.

7.4.3 Results

In the second row of Table 7.7, we present the BLEU score for the baseline system. To remind the reader, the baseline system learns from and translates source Finnish sentences which have not been segmented in any way. The baseline gets a BLEU score of 23.93. In the third and fourth rows of Table 7.7, we present results using the Morfessor toolkit. Both approaches lead to lower results then the baseline, though the approach that only keeps the stem (STEM), and removes all affix content, performs significantly worse. Finally, we present results of the segmented SMT systems using our exact HMMs in rows of 5-11 of Table 7.7. Again, all the results are below the baseline. Interestingly, it appears that using more context actually leads to lower BLEU scores, with the 3-gram HMM resulting in the score of 22.26, in comparison to a BLEU score

Model	BLEU	BLEU (distinct)
Independent system MBR rerankings		
Baseline	**23.82**	23.92
MORF-SPLIT	23.23	23.28
MORF-STEM	20.88	20.99
HMM3	22.36	22.48
HMM4	21.89	21.79
HMM5	22.00	22.03
HMM6	21.89	21.88
HMM7	21.95	21.92
HMM8	21.82	21.80
HMM9	21.47	21.48
System combination via MBR reranking		
Baseline + MORF-SPLIT	23.76	**23.94**
Baseline + MORF-STEM	22.41	22.58
Baseline + HMM3	23.81	23.79
Baseline + HMM4	23.72	23.79
Baseline + HMM5	23.61	23.74
Baseline + HMM6	23.68	23.68
Baseline + HMM7	23.75	23.75
Baseline + HMM8	23.08	23.79
Baseline + HMM9	23.54	23.66
Baseline + HMM3-9	23.14	23.29

Table 7.8: Translation scores with various different system combinations prior to running MBR. Those that combine two individual systems used the top 500 translations from each of the respective n- best lists. The combinations presented in the last two rows use the top 100 from each, giving n-best lists of 1000 and 1200 respectively. We highlight in bold the highest scores for each column.

of 21.41 for the 9-gram HMM. Our HMMs segmentors lead to translation systems that perform worse then baseline, and also worse then the MORF-SPLIT approach, but better then the MORF-STEM approach.

Model	1	2	3	4
Baseline	5.7%	34.4%	66.3%	85.5%
MORF-SPLIT	5.1%	19.5%	44.8%	69.0%
MORF-STEM	5.8%	22.3%	52.8%	82.3%
HMM3	4.7%	20.5%	40.9%	61.1%
HMM4	**4.6**%	19.8%	39.6%	59.3%
HMM5	**4.6**%	**19.6**%	39.4%	**58.9**%
HMM6	**4.6**%	19.7%	39.5%	59.3%
HMM7	**4.7**%	19.7%	**39.3**%	59.1%
HMM8	**4.6**%	19.7%	39.4%	**58.9**%
HMM9	4.7%	19.9%	39.6%	59.3%

Table 7.9: We show the OOV rates of the different models on the test set over types (i.e unique words), from the unigram (1) to 4-grams (4). We highlight in bold the lowest OOV rate in each column.

We now examine the impact of straightforward reranking using MBR. We present the results for all approaches in top half of Table 7.8. We present results using n-best lists of size 1000. We also compare the reranked results of non-unique output (second column), and n-best lists that contain only unique hypotheses (final column). Apart from the baseline which sees a non-significant drop in BLEU score, all approaches see improvements, albeit small, above the respective systems which do not use any form of reranking. Such improvements are in line with the literature [55, 109].

Combining the output of different systems has led to even further improvements in the literature. In the bottom half of Table 7.8, we present results of various different combinations. None of the approaches outperform the baseline. Baseline + MORF-SPLIT achieves an insignificant improvement of 0.02 over the baseline when using distinct n-best lists. This differs from previously published work [55] that demonstrated such straightforward combination of n-best lists could lead to system BLEU scores greater than those achieved by the individual systems that were combined.

7.4.4 Discussion

In Table 7.9, we show the OOV rates over types, for unigrams through to 4-grams, for each of the different models. At the unigram level, the HMM models have the OOV rates of between 4.6 and 4.7%, a drop of 1% in comparison to the baseline. If we

look at difference in the number of single word types that are unseen, the baseline has 598 types that are unseen, in comparison to 310 for the the HMM4 model. Further, even though MORF-SPLIT and MORF-STEM are based on the same analysis, as a percentage of types, MORF-STEM actually increases the number of unique types that are unseen. Here, if we look at the unique number of types are unseen, there are only 333. Interestingly, larger drops in OOV rates are seen when looking at the bigram to 4-gram rates, where we see drops in the region of $\sim 25\%$ in absolute percentage terms, from 85.5% to 58.9% for HMM5 and HMM8.

We present two translations, along with their source segmentations, in Table 7.10. If we compare the system output, we can begin to draw some tentative observations. The first observation we can draw, is that throwing away affixes in the STEM approach introduces a large amount of ambiguity. For example, if we look at the first sentence in the Table 7.10, we see that the STEM approach makes a sub-optimal translation for the phrase "jo reagoi nut." Looking at the phrase-table, we see "jo nut" has over 249 translation entries, and "on jo nut" has 213. In comparison, the SPLIT approach has only 4 translations for "on jo reagoi nut" and 5 for "on jo reagoi nut." This trend is the same for the baseline and HMM3 models, and also explains why the STEM approach outputs nonsense for the second example presented in Table 7.10. To give a further example, the STEM approach consistently mistranslates the Finnish word for "minutes" into "charter of fundamental rights." In fact, the consistent mistranslation of entire phrases by the STEM method becomes apparent when looking at its output.

However, drawing conclusions for the other approaches is far more difficult. Comparing the translation of the baseline, SPLIT, and HMM3 approaches, it is hard to see any noticeable trend that would explain the differences in the BLEU scores of the independent systems. In fact, we posit there are cases where the HMM3 or SPLIT approach provide a slightly better translation, but are unfairly penalised due to their being only a single reference translation. Looking at the second example in Table 7.10, the Finnish word "lokeroon" is unseen for the baseline approach. However, it still manages to have higher unigram and bigram clipped counts. In comparison, the SPLIT and HMM3 translations are arguably better, because they manage to capture the meaning of the original source sentence, albeit in an influent way. Yet, they have lower unigram and bigram clipped counts. Evaluation of the models in a setting where additional reference translations are available, is, in the opinion of the author, necessary prior to making a firm judgement on the presented methods.

A clear step for future-work, unexamined due to time limitations, is the application of the segmentation approaches in a backoff step, in combination with the baseline. That is, use the baseline model for seen tokens, but segment the unseen tokens and apply the segmented phrase-tables for those. Looking at the second example in 7.10, we can see how this may lead to a better translation then all of those presented.

Reference	the european union has reacted already .
B-Source	euroopan unioni on jo reagoinut .
B-Translation	the european union has **already reacted** .
SPLIT-Source	euroopa n unioni on jo reagoi nut .
SPLIT-Translation	the european union has **already reacted** .
STEM-Source	euroopa unioni on jo nut .
STEM-Translation	the european union **is** already **committed** .
HMM3-Source	euroopan unio ni on jo reagoi nut .
HMM3-Translation	the european union has **already reacted** .
Reference	however , we cannot put all forms of nationalism into one pigeon-hole .
B-Source	kaikkia nationalismin lajeja ei voida kuitenkaan luokitella samaan lokeroon .
B-Translation	all **types** of nationalism , however, cannot be put **in the same lokeroon** .
SPLIT-Source	kaikki a national ismi n lajeja ei voida kuitenkaan luokit ella samaa n lokero on .
SPLIT-Translation	all **types** of nationalism , however, cannot **be classified in the same box is** .
STEM-Source	kaikki national lajeja ei voida kuitenkaan ella samaa lokero .
STEM-Translation	all **international species may not** , however , **a political agreement** .
HMM3-Source	kaikki a natio n alismi n laje j a ei voida kui te n kaan luoki te lla sama an lokero on .
HMM3-Translation	all **types** of nationalism cannot , however , **be classified in the same class is** .

Table 7.10: We present some example translation from the 1-best output of the different systems, along with the reference sentence (common to all approaches), and the different source segmentations used. We highlight in bold the n-grams that appear in the system output but not the corresponding reference sentences. The prefix "B-" refers to the baseline system.

7.5 Conclusions

In this chapter, our aim was to examine the use of a novel, high-order HMMs for the morphological segmentation of Finnish, a language that is highly agglutinative. The main research question we asked in this chapter was:

RQ 4. Can high-order HMMs be used to recover a partition of the source side of a parallel corpora that reduces sparsity and leads to improved translation quality?

The general research question gave rise to the first specific subquestion:

RQ 4 a. HMMs that have a large latent vocabulary, or use a high amount of context, are often intractable. Can we create high-order HMMs that are tractable given large latent vocabulary spaces, or use large amounts of context, and are still exact?

We presented an approach in this chapter for exact decoding and sampling with an HMM whose hidden layer is a high-order LM. The algorithm we presented innovates on existing techniques in the following ways. First, it is a *joint* approach to sampling and optimisation (i.e. decoding), that is based on introducing a simplified, "optimistic," version $q(x)$ of the underlying language model $p(x)$, and for which it is tractable to use standard dynamic programming techniques, both for sampling and optimisation. We then formulate the problem of sampling and optimisation with the original model $p(x)$ in terms of a novel algorithm which can be viewed as a form of *adaptive rejection sampling* [72, 74], in which a low acceptance rate (in sampling) or a low ratio $p(x^*)/q(x^*)$ (in optimisation, with x^* the argmax of q) leads to a refinement of q, i.e., a slightly more complex and less optimistic q but with a higher acceptance rate or ratio.

Second, it is the first technique that we are aware of which is able to perform *exact sampling* with such models. Known techniques for sampling in such situations rely on approximation techniques such as Gibbs or Beam sampling (see e.g. [184, 196]). By contrast, our technique produces exact samples from the start, although in principle, the first sample may be obtained only after a long series of rejections, and therefore refinements. In practice, our experiments indicate that a good acceptance rate is obtained after a relatively small number of refinements. It should be noted that, in the case of exact optimisation, a similar technique to ours has been proposed in an image processing context [90], but without any connection to sampling. That paper, written in the context of image processing, appears to be little known in the NLP community.

To specifically answer research question 4.a, we validated our approach on two natural language tasks, SMS retrieval and POS tagging, showing that the proposal distributions we obtain require only a fraction of the space that would result from explicitly

representing the HMM, and, importantly, allow for the ability to do exact inference at a reasonable cost. We then turned to the application of our method to the task of text-segmentation, with the aim of reducing sparsity, improving word alignment and test set translation quality. We compared our approach to the unsupervised method of popular toolkit, Morfessor [55], and examined the combination of a variety of differently segmented models via MBR system combination. We asked the specific research question:

RQ 4 b. Does the addition of the additional context, contained in the higher-order models, lead to improved translation quality when segmenting Finnish? To what extent do the application of the different models leads to a reduction in sparsity problems and model learning?

Analysis of different morphological segmentation models found that, while they decreased OOV rates, halving the number of OOV types seen, we did not see any improvements in translation quality, measured with BLEU. None of the segmented approaches were able to achieve higher results then the baseline, with the MORF-SPLIT approach under-performing the baseline by 0.8 BLEU points, and our own HMM3 model under-performing the baseline by 1.65 BLEU points. While the segmentation models based on our novel HMMs did not outperform the baseline in terms of translation quality, as measured by BLEU, they did achieve the lowest OOV rates of 4.6% at the unigram level.

A qualitative analysis of the translation output of the different methods showed that, while the MORF-STEM approach lead to a noticeable decrease in translation quality, it was hard to see any decrease in quality with the MORF-SPLIT and HMM3 approaches. Further, it appeared that the MORF-SPLIT and HMM3 approaches were penalised by the experimental set-up and the fact that there was only a single reference translation. MORF-SPLIT and HMM3 approaches presented translations that had appropriate, but different, phrases to those used in the only reference translation, and were thus penalised in reference to the baseline, even in cases where they managed to capture the main semantic meaning of the sentence and the baseline did not.

The final specific research question we asked was:

RQ 4 c. Can improvements be made that exploit numerous different segmentation of the same input? What methods can we use do this?

Examining the use of our segmenters in a system-combination approach via MBR reranking did not lead to improvements, in contrast to previously published work [55]. At best, the combination of the baseline system with the MORF-SPLIT approach lead to BLEU results that were only 0.02 BLEU points below the baseline, a non-significant

difference. The combination of the systems using morphological analysis from our novel HMMs lead to BLEU scores of 0.15 BLEU points below the baseline.

To summarise, while we presented a novel algorithm that allows for tractable, exact, inference from high-order and large latent vocabulary HMMs, further experiments are needed before firm conclusions can be drawn on their utility for the task of source-side text-segmentation in an SMT system, especially as our results contrast with those in the literature. Also, the examination of a combination of the different approaches prior to translation (and not in a post-processing system combination step) should be conducted, to see if we can achieve the drop in OOV rates and see an increase in BLEU scores.

This chapter finalises Part II, and concludes the technical portion of the thesis. In the next chapter, we revisit the conclusions drawn, and list the unexplored avenues of research that could prove promising for future-work.

Chapter 8

Conclusions

In this thesis we have examined two distinct but complementary approaches to improving existing SMT algorithms, either through the exploration and extraction of new data from which we can learn new models, as discussed in Part I, or through the better exploitation of existing data sets, via the use of morphological and syntactic analysis. Across both parts of the thesis, we have examined the major areas of an SMT system, from initial language identification of online content (Chapter 4) and the acquisition of parallel data from the resulting language bins (Chapter 5), to reranking of output translation n-best lists (Chapter 6), and source preprocessing using supervised and unsupervised morphological analysis algorithms (Chapter 7).

In the remainder of this chapter we conclude the thesis. We first answer the research questions governing the preceding chapters in Section 8.1, and conclude by discussing the potential directions for future work in Section 8.2.

8.1 Main Findings

The general question governing this thesis has been: "How can can we leverage new and existing data resources in a statistical machine translation algorithms for improved translation performance." We have approached this problem in two stages; the first exploring the potential for microblogging platforms as a new source for extracting pseudo-parallel corpora, and in the second we look at exploiting existing data sets via the use of novel paraphrasing, syntax and inference algorithms. This main question, in two parts, has lead us to formulate four main research questions listed in Section 1.3

which have been answered in the previous chapters. In this section we recall the answers.

Given the aim of extracting new translations from the web, and in particular informal content sources such as microblogging platforms, we began by asking in Part I the following research question:

RQ 1. Can we successfully identify the language of a microblog post, given the short, idiomatic and unedited nature exhibited?

We first set about answering this question in Chapter 4 by examining the inherent assumption that such data would be harder to classify, and demonstrated that the short, idiomatic and unedited nature of microblog posts did make the task of language identification harder than otherwise. We noted a significant drop from a classification accuracy of 99.4% on formal texts to 92.5% accuracy on microblog posts. We proposed a number of models based on different additional sources of information on which to classify the language, and in conjunction we also examined a number of ways of integrating these sources of information and creating a confidence score of the classification. We found that using a prior based on the user's previous posts boosted classification accuracy the most. The use of all additional content sources, along with a post-dependent method of combing the respective priors led to an increase in the classification accuracy giving performance close to that of short formal text at 97.4%. Further, in those situations where no microblog specific data exists, we demonstrate that our methods still give a significant improvement of 3%, and 5% if we utilise the user prior.

To verify our approach, we applied the language identifier to 1.1 million posts; representing one day's worth of posts available for data analysis, for which we have no gold language label. We demonstrated that post frequency per hour, when split by the language identified, matched the GMT offsets, thereby validating our language identification method. Further, we demonstrated the unsuitability for the language and country metadata fields as indicators of the language post, either due to incorrect or sparse metadata fields.

Given improved language model classifiers for microblog posts, we then turned to the problem of creating a pseudo-parallel corpora from these new monolingual data sources in Chapter 5. We asked:

RQ 2. Given a set of language identified microblog posts, how can we learn new phrase translations from this data?

We examined two approaches for extracting the multilingual Twitter data for novel term translations, the first based on a self-learning approach, the second on an IR method.

Interestingly, while IR methods have typically outperformed self-learning based approaches (though they are not mutualy exclusive), we were unable to achieve improvements over the baseline when using the IR approach. We posit that this is due, in contrast to previous work, to the fact that we did not constrain our multilingual tweets to a specific topic or event. While our conclusions can only be tentative, we argue that the IR approach is only appropriate when the underlying posts share a topic or theme. Finally, as part of the evaluation process, we released a new labeled data set containing human translations of microblog posts, which will aid in future research on this task.

After that, we began the second part of the thesis in Chapter 6 exploring methods for the exploitation of syntax in a post-editing, reranking setting for improved translation quality. The research questions were:

RQ 3. Can a generative syntactic parser discriminate between human produced and machine output language? Are they useful in an SMT setting for reranking output, where global information may prove beneficial to translation quality?

We began by exploring the ability of an off-the-shelf parser to differentiate between human and machine produced sentences. We concluded that by utilising the raw score of a parser to discriminate between a fluent and disfluent sentence, we could outperform a more standard 4-gram language model. However, we also demonstrated, in accordance with the literature, that a simple metric based on the parser probability performed badly on the more difficult task of differentiating between sentences of varying levels of disfluency. To this end, we examined a large number of features, shallow and deep, to be used with a perceptron reranker. We further presented a simple but novel feature, efficient to compute, based on a non-content aware POS tagger. We demonstrated significant improvements over non-reranked output and output reranked using only lexical features. We concluded by showing that deep features, which are used with the intention of making models more generalisable, do not help within the perceptron reranking setting as they overfit the training data, leading to problems with robustness of results over different test sets.

We finalise the technical content of Part II in Chapter 7 by presenting a novel method that allows for the exact inference of high-order HMMs. Specifically, we ask

RQ 4. Can high-order HMMs be used to recover a partition of the source side of a parallel corpora that reduces sparsity and leads to improved translation quality?

We began by presenting a novel algorithm, based on adaptive rejection sampling, and demonstrated the validity of the approach on a POS tagging and ASR-like tasks. The experimental results empirically demonstrated the space on which the standard dynamic programming algorithms operate are far reduced over the standard space, lead-

ing to algorithms that run in linear as opposed to exponential time. This allows for the application of tractable high-order HMMs to existing NLP and SMT tasks.

Having demonstrated the validity of the approach, we then apply the algorithm to the task of source-side morpheme decomposition. We apply the method as a preprocessing step for a Finnish-English SMT system, where the highly agglutinative aspects of Finnish lead to sparsity problems usually seen in low-resource languages. We compared the method presented in the chapter to an unsupervised algorithm developed specifically for the morphological analysis of Finnish. In contrast to the literature, we found no improvements when using segmented source-side input over an unsegmented baseline, given either the existing aforementioned unsupervised approach, or the novel method we presented in the chapter. We found this to be the case for systems that used an individual segmentation, and for approaches that combined the output of multiple systems using different segmentations.

8.2 Implications for Future Work

There exist several unexplored avenues of research that are either opened or have not yet been fully addressed by the work presented in this thesis. Here we list these, in no particular order.

In Chapter 4, the first technical chapter of this thesis, we proposed a novel language identifier, tailored for the microblog setting, that examined the use of different priors for improved language identification accuracy. In experiments reported, the space of languages in the training, development and test data were restricted to a small set of five languages. This was because these languages were those which the author and colleagues could annotate. However a clear item for future work would include the explicit handling of an 'other' or 'unknown' category, which would prove beneficial for real-world systems, and more sophisticated approaches to combining priors, such as data fusion, that may be worth investigating. Other avenues, perpendicular to those already explored, involve acquiring more data via bootstrapping methods [170] or co-training methods [19], for the identification of tweets that can be used for training. Finally, although most priors examined in this work are specific to microblogging, certain features could be tested with respect to a Short Message Service (SMS) corpus.

In Chapters 5 and 6, we examined two distinct approaches to getting improved translation results; the first by adapting a translation model by learning new in-domain translations from distinct, non-parallel bilingual corpora, and by reranking the output of an n-best list using syntactic language models. While both approaches fall into the camp of more, relevant data, or more, deeper, features, the use of semantic features could prove beneficial in both distinct settings. In the first, it could aid as an anchor to

find more parallel tweets in the target bin for a given, unannotated, source tweet, and in the reranking setting, could be directly used as a feature of our models.

Alongside an examination of such semantic feature types, the use of topic models would also prove a natural extension. To a certain extent, this has already been examined for novel term translation finding [200], however such approaches have only been applied in constrained settings, such as Wikipedia, where the nature of the data means documents are already aligned and imply a high degree of comparability between sentences across the difference languages. A closer integration of the self-training and information retrieval based methods with mono or bilingual topic modelling [18, 126], potentially as a filter, could prove interesting as a way of finding new translations in the noisier microblog setting where there is no data defined, fixed document alignment step.

In Chapter 4, we ignored multilingual tweets. In fact we made the strong assumption that all tweets could only be assigned to one language category; a clear item of future work is the identification and resolution of multi-coded tweets, including the construction of more complex models that are sensitive to within-post language change, possibly via latent methods. This could then feed into not only improved language classifiers, but could be directly used by the methods in Chapter 5 for learning new, Twitter specific, phrase translations.

In Chapter 6 we proposed a simple POS tagger from which features could be extracted for the purpose of n-best list reranking. We demonstrated the benefits of the feature set over deep features which require a full generative parser, giving computational and translation quality benefits. However, we believe an examination in the use of partial parsers [3] may lead to a successful bridge between the benefits of deep features from a full parser and the coverage of data points from POS and other shallow tools, which may also resolve the sparsity problems previously encountered.

In Chapter 7 we proposed a joint algorithm for optimisation and sampling from otherwise intractable HMMs. The interplay between optimisation and sampling is a fruitful area of research that can lead to state-of-the art performances on inference and decoding tasks in the special case of high-order HMM decoding, but the method is generic enough to be generalised to many others models of interest for NLP applications. One family of models is provided by agreement-based models, for example HMM+PCFG, where distribution p takes the form of a product: $p(x) = p_{\text{HMM}}(x)p_{\text{PCFG}}(x)$. Even if the factors $p_{\text{HMM}}(x)$ and $p_{\text{PCFG}}(x)$ can be decoded and sampled efficiently, the product of them is intractable. Dual decomposition is a generic method that has been proposed for handling decoding (i.e., optimisation) with such models, by decoupling the problem into two alternating steps that can each be handled by dynamic programming or other polynomial-time algorithms [165]. This is an approach that has been applied to SMT (phrase-based [39] and hierarchical [164]). However, sampling such distribu-

tions remains a difficult problem. Extending the approach presented in Chapter 7 to such problems would be interesting.

A clear limitation of the current work on our variable order HMMs is a lack of discussion of a bound on the number of iterations needed to optimise or sample p. While the algorithm we presented is guaranteed to find the maximum, or to draw exact samples, from p, we provide no bounds on the number of refinements necessary in either the decoding or sampling scenarios. Yet, if too many refinements are necessary, then the q distribution can become too complex, thereby running into the initial problem that the dynamic programming algorithms encounter with the original p distribution; for example, see [192] for a description of how the intersection of too many automata (in our case, this is akin to the refinement steps we make) can lead to an intractable number of states. Naturally, the potential for this happening is dependent on the ambiguity and complexity inherent in the p distribution under consideration; i.e., the task for which this is being applied. Thus, it is possible that a case-by-case examination of the approach to different NLP tasks will need to be undertaken, including an alteration of the refinement algorithm we presented in Section 7.2.5. The refinement strategy proposed was simple, and by no means the only or best possible. An exploration of different refinement strategies would be an interesting avenue of further study.

On a different tack, the explicit edge representation used by Algorithm 3 is suboptimal; because edges between nodes are explicitly represented, this has a detrimental impact on the running time, as the refinement process then has to copy and move over a large number of edges. We posit that an implicit representation would lead to an algorithm that is computationally faster. Finally, all our HMMs were supervised; that is, the observation and transition models were learnt from labeled data. It would be interesting to examine the relationship between our variable-order HMMs and unsupervised learning approaches (e.g., see [87]).

In Chapters 6 and 7, we looked at the application of tools optimised towards a different criterion, whether it be recovering the best human segmentation, or the gold label parse. However, our end goal has been improved translation accuracy. A limitation is thus the fact we did not optimise towards translation quality, represented by BLEU. For example, Nguyen et al. [133] segment the source-side of their bitext using a gibbs sampler tied to the final translation quality; they learn a segmentation that, primarily, leads to good translations, as opposed to improvements against some other metric measuring the morphological or syntactic correctness. Paul et al. [147] propose a segmenter along the same vain. For parsing, an algorithm with multiple objective learning functions was proposed in [77]; they demonstrate improvements in translation when optimising their parser against METEOR. Directly optimising towards translation may also help prevent problems arising from cascading errors; the problem of using syntactic toolkits in a greedy 1-best, black-box fashion was demonstrated in [67].

Appendix A

Example Tweets Returned by Models in Chapter 5

original tweet
been there . hope you are home right now with a cold beer .
ir retrieved tweet
zo , trainen van vanavond zit er weer op. nu thuis op de bank met een overheerlijk koud biertje
self translation
@loriming er geweest . hoop dat je huis nu met een koud biertje .
original tweet
so i swallow the fear ! the only thing i should be drinking is an ice cold beer ! falou e disse , pink !
ir retrieved tweet
dus ik ben niet de enige die altijd een hekel heeft aan het gaan zitten op een koude wc bril ?
self translation

dus ik wel de angst ! het enige wat ik wil drinken is een ijs koude bier ! falou e disse , roze !
original tweet
candles , hot bath and cold beer . i love my wife !
ir retrieved tweet
rt : ik hou van mijn bad “heart” (replace with utf heart)
self translation
kaarsen , warm bad en koud bier . ik hou van mijn vrouw ! http:/ /yfrog.com/h7sowcij
original tweet
phew finishing work , looking forward to a cold beer !!!
ir retrieved tweet
wat een storm , phew !
self translation
phew laatste werk , hopen op een koude bier !!!
original tweet
positively knackered , ca n’t wait for dinner round my folks and a cold beer . bliss .
ir retrieved tweet
heb nt een facbook aangmaakt voor mijn moeder xd
self translation
positief knackered , ca n’t wachten voor diner ronde mijn beste mensen en een koude bier . bliss .
original tweet
cold beer & a fishing pole by trent willmon or cornbread by band of heathens
ir retrieved tweet
krijg koude handen van dat bier !!! #isooknooitgoed
self translation

@ajwilliams04 koude bier & een visserij - pool van trent willmon of cornbread van band of heathens

original tweet

all stuff moved from a to b , time for a refreshing and well - deserved cold beer

ir retrieved tweet

nu eerst tijd voor een welverdiend bad ! #rust

self translation

alle dingen verplaatst van a naar b , tijd voor een verfrissend en welverdiend koud bier

original tweet

if you are looking for somewhere to get out of the heat it is cooler in the #adelhills plus we have air - conditioning and ice cold beer !

ir retrieved tweet

op 20 april de airco aan , de warmte stijgt ons naar het hoofd !

self translation

als je op zoek bent naar een uitweg te vinden uit de warmte is het koeler in de #adelhills en we hebben de airco en ijs koud bier !

original tweet

not a worry in the world a cold beer in my hand , life is good today !

ir retrieved tweet

we hoeven ons geen zorgen te maken over de sneuwhoogtes schneehoehen :)

self translation

@davephillips13 geen zorgen hoeven te maken in de wereld een koud biertje in mijn hand , het leven is goed vandaag !

original tweet

just found this one – & gt ; cold beer : advertising for student council elections . is at it again .

ir retrieved tweet

het bier is op , voor straf maar aan de baco , uit een soepkom , dat dan weer wel

self translation
zojuist vernomen dat een - & gt , koude bier : reclame voor de gemeenteraadsverkiezingen . @oakleydavid is aan het weer .

original tweet
rewarding my study - filled day with a cold beer . immediately followed by another cold beer and more studying .
ir retrieved tweet
gebruik een keer het woord afvallen in een tweet en word meteen gevolgd door dr . frank
self translation
het belonen van mijn studie gevulde dag met een koud biertje . meteen gevolgd door een koude bier en meer studeren .

original tweet
damn nitrogen in my bloodstream , a cold beer would be the only thing that could make this better
ir retrieved tweet
kan je ervoor zorgen dat ' k mijn hakken voor weekend krijg ? heb een feestje ' k wil ze aandoen xx
self translation
grote stikstof in mijn bloed , een koud biertje is het enige dat ervoor kan zorgen dat deze beter

original tweet
rt : cold beer on a friday night , a pair of jeans that fit just right , and the radiooo onnnnonnnonnn
ir retrieved tweet
glee op de radiooo
self translation
rt @lehoop18 : koude bier op een vrijdag avond , een paar jeans , dat past alleen recht , en de radiooo onnnnonnnonnn

original tweet
i ’m ready for my mom ’s pozole which is the best on the planet ! and and cold beer
ir retrieved tweet
ik denk dat man city na real madrid en barcelona de beste selectie van de wereld heeft hoor .
self translation
ik ben klaar voor mijn mom ’s pozole dat is de beste van de wereld ! en en koud bier .
original tweet
i could go for a cold beer and its 5am . .
ir retrieved tweet
ook al haat ik haar kapot erg , toch zou ik dood gaan voor haar .
self translation
ik zou gaan voor een koude bier en haar 5am . .

Table A.1: In this table we list all the target tweets in the corpus of 1 million tweets which contained the phrase “cold beer”, and then show the tweet retrieved using the IR approach, and the translation of the tweet using an SMT system trained on out-of-domain Europarl data.

Bibliography

[1] S. Abdul-Raud and H. Schwenk. On the use of Comparable Corpora to improve SMT performance. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 16–23, Athens, Greece, 2009. (Cited on pages 61, 76, and 78.)

[2] S. Abdul-Rauf and H. Schwenk. Parallel sentence generation from comparable corpora for improved SMT. *Machine Translation*, 25(4):341–375, 2011. (Cited on page 61.)

[3] S. Abney. Part-of-Speech Tagging and Partial Parsing. In *Corpus-Based Methods in Language and Speech*, pages 118–136, 1996. (Cited on page 145.)

[4] A. Agarwal and A. Lavie. Meteor, M-BLEU and M-TER: Evaluation Metrics for High-Correlation with Human Rankings of Machine Translation Output. In *Proceedings of the ACL 2008 Third Workshop on Statistical Machine Translation (WMT 2008)*, pages 115–118, Columbus, OH, USA, 2008. (Cited on page 26.)

[5] L. Ahrenberg and M. Merkel. On Translation Corpora and Translation Support Tools: A Project Report. *Languages in Contrast. Papers from a Symposium on Text-based cross-linguistic studies*, 1996. (Cited on page 3.)

[6] A. Ahsan, P. Kolachina, S. Kolachina, and D. M. Sharma. Coupling Statistical Machine Translation with Rule-based Transfer and Generation. In *Proceedings of the Association of Machine Translation in the Americas (AMTA 2010)*, pages 53–60, Denver, CO, USA, 2010. (Cited on page 4.)

[7] D. Altheide. *Qualitative Media Analysis*. Sage Publications Inc, 1996. (Cited on page 33.)

[8] A. Arun and P. Koehn. Online Learning Methods for Discriminative Training of Phrase Based Statistical Machine Translation. In *Proceedings of the Eleventh Machine Translation Summit of the International Association for Machine Translation (MT Summit XI)*, pages 15–20, Copenhagen, Denmark, 2007. (Cited on page 86.)

[9] A. Arun, C. Dyer, B. Haddow, P. Blunsom, A. Lopez, and P. Koehn. Monte Carlo inference and maximization for phrase-based translation. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009)*, pages 102–110, Boulder, Colorado, 2009. (Cited on page 131.)

[10] A. Arun, B. Haddow, P. Koehn, A. Lopez, C. Dyer, and M. Osborne. Monte Carlo Techniques for Phrase-Based Translation. *Machine Translation Journal*, 24(2):103–121, 2010. (Cited on page 131.)

[11] T. Baldwin and M. Lui. Language Identification: The Long and the Short of the Matter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Asso-*

ciation for Computational Linguistics: Human Language Technologies (NAACL-HLT 2010), pages 229–237, Los Angeles, CA, USA, 2010. (Cited on pages 33 and 36.)

[12] P. Banerjee, J. Du, B. Li, S. K. Naskar, A. Way, and J. V. Genabith. Combining Multi-Domain Statistical Machine Translation Models using Automatic Classifiers. In *Proceedings of Association of Machine Translation in the Americas (AMTA 2010)*, pages 141–150, Denver, CO, USA, 2010. (Cited on page 62.)

[13] S. Banerjee and A. Lavie. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, pages 65–72, Ann Arbor, MI, USA, 2005. (Cited on page 26.)

[14] A. Bhargava and G. Kondrak. Language Identification of Names with SVMs. In *Proceedings of the Human Language Technologies: The 11th Annual Conference of the North American Chapter of the ACL (HLT-NAACL 2010)*, pages 693–696, Los Angeles, CA, USA, 2010. (Cited on page 36.)

[15] D. M. Bikel. Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine. In *Proceedings of the second international conference on Human Language Technology Conference (HLT 2002)*, pages 178–182, San Diego, CA, 2002. (Cited on page 87.)

[16] J. A. Bilmes and K. Kirchhoff. Factored Language Models and Generalized Parallel Backoff. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, pages 4–6, Edmonton, Canada, 2003. (Cited on page 85.)

[17] A. Birch, M. Osborne, and P. Koehn. CCG Supertags in Factored Statistical Machine Translation. In *Proceedings of the ACL 2007 Second Workshop on Statistical Machine Translation (WMT 2007)*, pages 9–16, Prague, Czech Republic, 2007. (Cited on pages 85, 96, and 114.)

[18] D. Blei and M. Jordan. Modeling Annotated Data. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 127–134, Toronto, Canada, 2003. (Cited on page 145.)

[19] A. Blum and T. Mitchell. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the Eleventh Annual Conference on Computational learning theory (COLT 1998)*, pages 92 – 100, New York, NY, USA, 1998. (Cited on page 144.)

[20] P. Blunsom, T. Cohn, and M. Osborne. A Discriminative Latent Variable Model for Statistical Machine Translation. In *Proceedings of the Human Language Technology Conference of the 46th Annual Meeting of the Association for Computational Linguistics (HLT-ACL 2008)*, pages 200–208, Columbus, OH, USA, 2008. (Cited on page 86.)

[21] T. Brants. TnT - A Statistical Part-of-Speech Tagger. In *Proceedings of the Sixth conference of Applied Natural Language Processing (ANLP 2001)*, pages 224–231, Washington, DC, USA, 2001. (Cited on page 127.)

[22] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean. Large Language Models in Machine Translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 858–867, Prague, Czech Republic, 2007. (Cited on page 18.)

[23] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, August 1996. (Cited on page 112.)

[24] P. F. Brown, J. Cocke, S. A. Pietra, V. J. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A Statistical Approach to Machine Translation. *Computational Linguistics*, 16(2):79–85, 1990. (Cited on pages 4 and 16.)

[25] P. F. Brown, V. J. Pietra, P. V. de Souza, J. C. Lai, and R. L. Mercer. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479, 1992. (Cited on page 85.)

[26] P. F. Brown, V. J. Pietra, S. A. Pietra, and R. L. Mercer. The mathematics of Statistical Machine Translation: parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993. (Cited on pages 4, 14, and 16.)

[27] C. Callison-Burch, M. Osborne, and P. Koehn. Re-evaluating the role of BLEU in Machine Translation Research. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*, pages 249–256, Trento, Italy, 2006. (Cited on page 27.)

[28] C. Callison-Burch, P. Koehn, C. Monz, and O. Zaidan. Findings of the 2011 Workshop on Statistical Machine Translation. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation (WMT 2011)*, pages 22–64, Edinburgh, Scotland, 2011. (Cited on page 62.)

[29] S. Carter and C. Monz. Parsing Statistical Machine Translation Output. In *Proceedings of the Language & Technology Conference (LTC 2009)*, pages 270–274, Poznań, Poland, 2009. (Cited on pages 11, 86, and 87.)

[30] S. Carter and C. Monz. Discriminative Syntactic Reranking for Statistical Machine Translation. In *Proceedings of Association of Machine Translation in the Americas (AMTA 2010)*, pages 3–12, Denver, CO, USA, 2010. (Cited on pages 11 and 86.)

[31] S. Carter and C. Monz. Syntactic Discriminative Language Model Rerankers for Statistical Machine Translation. *Machine Translation*, 25(4):317–339, 2011. (Cited on page 11.)

[32] S. Carter, C. Monz, and S. Yahyaei. The QMUL System Description for IWSLT 2008. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2008)*, pages 104–107, Hawaii, USA, 2008. (Cited on page 11.)

[33] S. Carter, M. Tsagkias, and W. Weerkamp. Twitter hashtags: Joint Translation and Clustering. In *Proceedings of the 3rd International Conference on Web Science (WebSci 2011)*, Koblentz, Germany, 2011. (Cited on page 11.)

[34] S. Carter, M. Tsagkias, and W. Weerkamp. Semi-Supervised Priors for Microblog Language Identification. In *Dutch-Belgian Information Retrieval workshop (DIR 2011)*, Amsterdam, The Netherlands, 2011. (Cited on page 11.)

[35] S. Carter, M. Dymetman, and G. Bouchard. Exact Sampling and Decoding in High-Order Hidden Markov Models. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 26–37, Jeju, South Korea, 2012. (Cited on page 11.)

[36] S. Carter, W. Weerkamp, and E. Tsagkias. Microblog Language Identification: Overcoming the Limitations of Short, Unedited and Idiomatic Text. *Language Resources and Evaluation Journal*, 2012. (Cited on page 11.)

[37] W. Cavnar and J. Trenkle. N-Gram-Based Text Categorization. In *Proceedings of Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR 1994)*, pages 161–175, Las Vegas, NV, USA, 1994. (Cited on page 36.)

[38] P.-C. Chang and K. Toutanova. A Discriminative Syntactic Word Order Model for Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pages 9–16, Prague, Czech Republic, 2007. (Cited on page 86.)

[39] Y.-W. Chang and M. Collins. Exact Decoding of Phrase-Based Translation Models through Lagrangian Relaxation. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP 2011)*, pages 26–37, Edinburgh, Scotland, 2011. (Cited on pages 112 and 145.)

[40] P. Charoenpornsawat, V. Sornlertlamvanich, and T. Charoenporn. Improving translation quality of rule-based machine translation. In *Proceedings of 19th International Conference on Computational Linguistics (COLING 2002)*, pages 1–6, Taipei, Taiwan, 2002. (Cited on page 3.)

[41] S. F. Chen and J. Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics (ACL 1998)*, pages 310–318, Montreal, Canada, 1998. (Cited on pages 17 and 18.)

[42] S. F. Chen and J. Goodman. An Empirical Study of Smoothing Methods for Language Modelling. Technical Report TR-10-98, University of Harvard, Harvard, CA, USA, 1998. (Cited on pages 17 and 93.)

[43] X. Chen, H. Wang, and X. Lin. Learning to Rank with a Novel Kernel Perceptron Method. In

Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009), pages 505–512, Honk Kong, China, 2009. (Cited on page 94.)

[44] D. Chiang. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 263–270, Ann Arbor, MI, USA, 2005. (Cited on pages 20 and 83.)

[45] D. Chiang. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228, 2007. (Cited on pages 20 and 83.)

[46] D. Chiang, Y. Marton, and P. Resnik. Online Large-margin Training of Syntactic and Structural Translation Features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 224–233, Honolulu, Hawaii, 2008. (Cited on page 86.)

[47] D. Chiang, W. Wang, and K. Knight. 11,001 New Features for Statistical Machine Translations. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT 2009)*, pages 218–226, Boulder, CO, USA, 2009. (Cited on pages 86 and 96.)

[48] T. Cohn. Efficient Inference in Large Conditional Random Fields. In *Proceedings of the 17th European Conference on Machine Learning (ECML 2006)*, pages 606–613, Berlin, Germany, 2006. (Cited on page 112.)

[49] M. Collins. Three Generative, Lexicalized Models for Statistical Parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics (ACL-EACL 1997)*, pages 16–23, Madrid, Spain, 1997. (Cited on page 87.)

[50] M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, Philadelphia, PA, USA, 1999. (Cited on page 98.)

[51] M. Collins and N. Duffy. New Ranking Algorithms for Parsing and Tagging: kernels over Discrete Structures, and the Voted Perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 263–270, Philadelphia, PA, USA, 2002. (Cited on pages 94 and 95.)

[52] M. Collins, B. Roark, and M. Saraclar. Discriminative Syntactic Language Modeling for Speech Recognition. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 507–514, Ann Arbor, MI, USA, 2005. (Cited on pages 86, 91, 94, and 96.)

[53] K. Crammer and Y. Singer. Pranking with Ranking. In *Proceedings of the Twenty-Fifth Annual Conference on Advances in Neural Information Processing Systems (NIPS 2001)*, pages 641–647, Vancouver, B.C, Canada, 2001. (Cited on page 94.)

[54] M. Creutz and K. Lagus. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR 2005)*, pages 106–113, Espoo, Finland, 2005. (Cited on pages 111 and 129.)

[55] A. de Gispert, S. Virpioja, M. Kurimo, and W. Byrne. Minimum Bayes risk combination of translation hypotheses from alternative morphological decompositions. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies, Companion Volume: Short Papers (NAACL-HLT 2009)*, pages 73–76, Boulder, CO, USA, 2009. (Cited on pages 110, 113, 135, and 139.)

[56] A. E. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from Incomplete Data via the EM algorithm. *Journal of the Royal Statistical Society*, B(39):1–38, 1977. (Cited on page 16.)

[57] J. DeNero, D. Chiang, and K. Knight. Fast consensus decoding over translation 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 (ACL-IJCNLP 2009)*, pages 567–575, Suntec, Singapore, 2009. (Cited on page 130.)

[58] M. Denkowski and A. Lavie. METEOR-NEXT and the METEOR Paraphrase Tables: Improved

Evaluation Support For Five Target Languages. In *Proceedings of the ACL 2010 Joint Workshop on Statistical Machine Translation and Metrics MATR*, pages 339–342, Uppsala, Sweden, 2010. (Cited on page 26.)

[59] M. Denkowski and A. Lavie. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation (WMT 2011)*, pages 85–91, Edinburgh, Scotland, 2011. (Cited on page 26.)

[60] T. G. Dietterich. Ensemble Methods in Machine Learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15, 2000. (Cited on page 40.)

[61] G. Doddington. The NIST automated measure and its relation to IBMs BLEU. In *LREC-2002 Workshop Machine Translation Evaluation: Human Evaluators Meet Automated Metrics*, Las Palmas, Spain, 2002. (Cited on page 27.)

[62] L. Dugast, J. Senellart, and P. Koehn. Statistical post-editing on SYSTRAN's rule-based translation system. In *Proceedings of the ACL 2007 Second Workshop on Statistical Machine Translation (WMT 2007)*, pages 220–223, Prague, Czech Republic, 2007. (Cited on page 4.)

[63] C. Dyer, S. Muresan, and P. Resnik. Generalizing Word Lattice Translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, pages 1012–1020, Columbus, OH, USA, 2008. (Cited on page 113.)

[64] J. L. Elsas, V. R. Carvalho, and J. G. Carbonell. Fast Learning of Document Ranking Functions with the Committee Perceptron. In *Proceedings of the International Conference on Web search and Web Data Mining (WSDM 2008)*, pages 55–64, Stanford, CA, USA, 2008. (Cited on pages 94 and 95.)

[65] A. Emami, K. Papineni, and J. Sorensen. Large-Scale Distributed Language Modeling. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, pages 37–40, Honolulu, Hawaii, 2007. (Cited on page 93.)

[66] P. F. Felzenszwalb, D. P. Huttenlocher, and J. M. Kleinberg. Fast Algorithms for Large-State-Space HMMs with Applications to Web Usage Analysis. In *Proceedings of Advances in Neural Information Processing Systems 16 (NIPS 2003)*, Vancouver, B.C, Canada, 2003. (Cited on page 112.)

[67] J. R. Finkel, C. D. Manning, and A. Y. Ng. Solving the Problem of Cascading Errors: Approximate Bayesian Inference for Linguistic Annotation Pipelines. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 618–626, Sydney, Australia, 2006. (Cited on page 146.)

[68] F. Fleuret and D. Geman. Coarse-to-Fine Face Detection. *International Journal of Computer Vision*, 41(1-2):85–107, 2001. (Cited on page 111.)

[69] Y. Freund and R. E. Schapire. Large Margin Classification Using the Perceptron Algorithm. *Machine Learning*, 37(3):277–296, 1999. (Cited on pages 94 and 95.)

[70] W. Gale and K. Church. A Program for Aligning Sentences in Bilingual Corpora. *Computational Linguistics*, 19(1), 1993. (Cited on page 29.)

[71] S. I. Gallant. Perceptron-Based Learning Algorithms. *IEEE Transactions on Neural Networks*, 1(2): 179–191, 1999. (Cited on page 95.)

[72] W. R. Gilks and P. Wild. Adaptive Rejection Sampling for Gibbs Sampling. *Applied Statistics*, 42(2): 337–348, 1992. (Cited on page 138.)

[73] G. Golovchinsky and M. Efron. Making Sense of Twitter Search. In *Proceedings of CHI 2010 Workshop on Microblogging: What and How Can We Learn From It?*, Atlanta, GA, USA, 2010. (Cited on page 33.)

[74] D. Görür and Y. W. Teh. Concave Convex Adaptive Rejection Sampling. Technical report, Gatsby Computational Neuroscience Unit, 2008. (Cited on page 138.)

[75] T. Gottron and N. Lipka. A Comparison of Language Identification Approaches on Short, Query-Style Texts. In *Advances in Information Retrieval, 32nd European Conference on IR Research (ECIR 2010)*, pages 611–614, Milton Keynes, England, 2010. (Cited on page 36.)

[76] B. Haddow and P. Koehn. Analysing the Effect of Out-of-Domain Data on SMT Systems. In *Pro-*

ceedings of the Seventh Workshop on Statistical Machine Translation (WMT 2012), pages 422–432, Montreal, Canada, 2012. (Cited on page 59.)

[77] K. B. Hall, R. McDonald, J. Katz-Brown, and M. Ringgaard. Training dependency parsers by jointly optimizing multiple objectives. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1489–1499, Edinburgh, Scotland, 2011. (Cited on page 146.)

[78] Z. S. Harris. Review of Louis H. Gray, Foundations of Language. *Language*, 16(3):216–231, 1940. (Cited on page 130.)

[79] S. Hasan, O. Bender, and H. Ney. Reranking Translation Hypotheses Using Structural Properties. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*, pages 41–48, Trento, Italy, 2006. (Cited on page 85.)

[80] L. Hong, G. Convertino, and E. H. Chi. Language matters in Twitter: a large scale study. In *Proceedings of AAAI Conference on Weblogs and Social Media (ICWSM 2011)*, pages 518–521, Barcelona, Spain, 2011. (Cited on pages 50 and 51.)

[81] M. Hopkins and J. May. Tuning as Ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1352–1362, Edinburgh, Scotland, July 2011. (Cited on page 23.)

[82] Z. Huang, Y. Chang, B. Long, J.-F. Crespo, A. Dong, S. Keerthi, and S.-L. Wu. Iterative Viterbi A* Algorithm for K-Best Sequential Decoding. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 611–619, Jeju, Korea, 2012. (Cited on page 112.)

[83] M. Huck, D. Vilar, D. Stein, and H. Ney. Lightly-Supervised Training for Hierarchical Phrase-Based Machine Translation. In *Proceedings of the EMNLP 2011 First Workshop on Unsupervised Learning in NLP*, pages 91–96, Edinburgh, Scotland, 2011. (Cited on page 61.)

[84] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury. Twitter Power: Tweets as Electronic Word of Mouth. *Journal of the American Society for Information Science and Technology*, 60(11):2169–2188, 2009. (Cited on page 33.)

[85] L. Jehl, F. Hieber, and S. Riezler. Twitter Translation using Translation-Based Cross-Lingual Retrieval. In *Proceedings of the 7th Workshop on Statistical Machine Translation (WMT 2012)*, pages 410–421, Montreal, Canada, 2012. (Cited on pages 62, 66, 76, 78, 79, and 80.)

[86] M. Jeong, C.-Y. Lin, and G. G. Lee. Efficient Inference of CRFs for Large-Scale Natural Language Data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing: Short Papers (ACL-IJCNLP 2009)*, pages 281–284, Suntec, Singapore, 2009. (Cited on page 112.)

[87] M. Johnson. Why Doesn't EM Find Good HMM POS-Taggers? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 296–305, Prague, Czech Republic, 2007. (Cited on page 146.)

[88] K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 60(5):493–502, 2004. (Cited on page 61.)

[89] N. Kaji, Y. Fujiwara, N. Yoshinaga, and M. Kitsuregawa. Efficient Staggered Decoding for Sequence Labeling. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 485–494, Uppsala, Sweden, 2010. (Cited on page 112.)

[90] A. C. Kam and G. E. Kopec. Document image decoding by heuristic search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:945–950, 1996. (Cited on pages 112, 115, and 138.)

[91] H. Ke and R. Haenlein. Users of the world unite! The challenges and opportunities of Social Media. *Business Horizons*, 53(1):59–68, 2010. (Cited on page 6.)

[92] M. Khalilov and J. Fonollosa. Syntax-based reordering for statistical machine translation. *Computer Speech and Language journal (Elsevier)*, 25:761–788, October 2011. (Cited on page 113.)

[93] M. Khalilov and K. Simaan. Context-Sensitive Syntactic Source-Reordering by Statistical Transduction. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 38–46, Chiang Mai, Thailand, 2011. (Cited on page 113.)

[94] M. Khalilov, J. Fonollosa, and M. Dras. A New Subtree-Transfer Approach to Syntax-Based Reordering for Statistical Machine Translation. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT 2009)*, pages 198–204, Barcelona (Spain), May 2009. (Cited on page 113.)

[95] C. Kit, H. Pan, and J. J. Webster. Example-Based Machine Translation: A New Paradigm. In B. S. Chan, editor, *Translation and Information Technology*, pages 57–78. Chinese U of HK Press, 2002. (Cited on page 3.)

[96] R. Kneser and H. Ney. Improved Backing-off for m-gram Language Modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 1995)*, pages 181–184, Detroit, MI, USA, 1995. (Cited on page 18.)

[97] K. Knight. Decoding Complexity in Word-Replacement Translation Models. *Computational Linguistics*, 25(4):607–615, 1999. (Cited on page 20.)

[98] P. Koehn. Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In *Proceedings of the Sixth Conference of the Association for Machine Translation in the Americas (AMTA 2004)*, pages 115–124, Washington, DC, USA, 2004. (Cited on page 20.)

[99] P. Koehn. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 388–395, Barcelona, Spain, 2004. (Cited on pages 28 and 103.)

[100] P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the Tenth Machine Translation Summit of the International Association for Machine Translation (MT Summit X)*, pages 79–86, Phuket, Thailand, 2005. (Cited on pages 29, 33, and 66.)

[101] P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of Tenth Machine Translation Summit of the International Association for Machine Translation (MT Summit X)*, pages 79–86, Phuket, Thailand, 2005. (Cited on pages 123 and 131.)

[102] P. Koehn. *Statistical Machine Translation*. Cambridge University Press, 2010. (Cited on page 130.)

[103] P. Koehn and H. Hoang. Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 868–876, Prague, Czech Republic, 2007. (Cited on pages 85 and 114.)

[104] P. Koehn and K. Knight. Empirical Methods for Compound Splitting. In *Proceedings of the Tenth Conference on European chapter of the Association for Computational Linguistics (EACL 2003)*, pages 187–193, Budapest, Hungary, 2003. (Cited on page 113.)

[105] P. Koehn and C. Monz. Manual and Automatic Evaluation of Machine Translation between European Languages. In *Proceedings of the NAACL 2006 Workshop on Statistical Machine Translation (WMT 2006)*, pages 102–121, New York, NY, USA, 2006. (Cited on page 25.)

[106] P. Koehn, F. J. Och, and D. Marcu. Statistical Phrase-Based Translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, pages 48–54, Edmonton, Canada, 2003. (Cited on pages 20 and 83.)

[107] 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 the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pages 177–180, Prague, Czech Republic, 2007. (Cited on pages 15, 20, 23, 28, and 87.)

[108] T. Koo, A. M. Rush, M. Collins, T. Jaakkola, and D. Sontag. Dual Decomposition for Parsing with Non-Projective Head Automata. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP 2010)*, pages 1–11, MIT,MA,USA, 2010. (Cited on page 112.)

[109] S. Kumar, W. Byrne, and S. Processing. Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, pages 169–176, Boston, MA, USA, 2004. (Cited on pages 130 and 135.)

[110] M. Kurimo, S. Virpioja, V. Turunen, and K. Lagus. Morpho challenge 2005-2010: Evaluations and results. In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 87–95, Uppsala, Sweden, 2010. (Cited on pages 110 and 132.)

[111] P. Lambert, H. Schwenk, C. Servan, and S. Abdul-Rauf. Investigations on Translation Model Adaptation Using Monolingual Data. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation (WMT 2011)*, pages 284–293, Edinburgh, Scotland, July 2011. (Cited on page 61.)

[112] P. Langlais and M. Simard. Merging Example-Based and Statistical Machine Translation: An Experiment. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation (AMTA 2002)*, pages 104–113, Tiburon, CA, USA, 2002. (Cited on page 4.)

[113] A. Lavie and A. Agarwal. METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In *Proceedings of the ACL 2007 Second Workshop on Statistical Machine Translation (WMT 2007)*, pages 228–231, Prague, Czech Republic, 2007. (Cited on page 26.)

[114] A. Lavie and M. Denkowski. The Meteor Metric for Automatic Evaluation of Machine Translation. *Machine Translation*, 23:105–115, 2009. (Cited on page 26.)

[115] A. Li. Results of the 2005 NIST Machine Translation Evaluation. In *Machine Translation Evaluation Workshop*, 2005. (Cited on page 27.)

[116] Z. Li and S. Khudanpur. Large-scale Discriminative n-gram Language Models for Statistical Machine Translation. In *Proceedings of the The Eighth Conference of the Association for Machine Translation in the Americas (AMTA 2008)*, pages 133–142, Waikiki, Hawaii, 2008. (Cited on pages 86, 94, 95, and 101.)

[117] P. Liang, A. Bouchard-Côté, D. Klein, and B. Taskar. An End-to-End Discriminative Approach to Machine Translation. In *Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL 2006)*, pages 761–768, Sydney, Australia, July 2006. (Cited on page 86.)

[118] C. Y. Lin and F. J. Och. ORANGE: a Method for Evaluating Automatic Evaluation Metrics for Machine Translation. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 501–507, Geneva, Switzerland, 2004. (Cited on pages 90 and 94.)

[119] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. Coarse-to-Fine Pedestrian Localization and Silhouette Extraction for the Gait Challenge Data Sets . In *Proceedings of the 2006 IEEE International Conference on Multimedia and Expo (ICME 2006)*, pages 1009–1012, Toronto, ON, Canada, 2006. (Cited on page 111.)

[120] M. Marcus, G. Kim, M. A. Marcinkiewicz, R. Macintyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. The Penn Treebank: Annotating Predicate Argument Structure. In *Proceedings of the Human Language Technology Workshop (HLT 1994)*, pages 114–119, Plainsboro, NJ, USA, 1994. (Cited on page 87.)

[121] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330, 1993. (Cited on page 127.)

[122] B. Martins and M. Silva. Language Identification in Web Pages. In *Proceedings of the 2005 ACM symposium on Applied Computing (SAC 2005)*, pages 764–768, Santa Fe, NM, USA, 2005. (Cited on page 36.)

[123] K. Massoudi, M. Tsagkias, M. de Rijke, and W. Weerkamp. Incorporating Query Expansion and Quality Indicators in Searching Microblog Posts. In *33rd European Conference on Information Re-*

trieval (ECIR 2011), pages 362–367, Dublin, Ireland, 2011. (Cited on page 33.)

[124] E. Matusov, N. Ueffing, and H. Ney. Computing Consensus Translation from Multiple Machine Translation Systems Using Enhanced Hypotheses Alignment. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*, pages 33–40, Trento, Italy, 2006. (Cited on page 130.)

[125] R. McDonald. Characterizing the Errors of Data-driven Dependency Parsing Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the Conference on Natural Language Learning (EMNLP-CoNLL 2007)*, pages 121–131, Prague, Czech Republic, 2007. (Cited on page 89.)

[126] D. Mimno, H. M. Wallach, J. Naradowsky, D. A. Smith, and A. McCallum. Polylingual Topic Models. In *Proceeding of the conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 880–889, Singapore, 2009. (Cited on page 145.)

[127] B. Mohit and R. Hwa. Localization of Difficult-to-Translate Phrases. In *Proceedings of the ACL 2007 Second Workshop on Statistical Machine Translation (WMT 2007)*, pages 248–255, Prague, Czech Republic, 2007. (Cited on page 96.)

[128] S. Mozes, O. Weimann, and M. Ziv-Ukelson. Speeding Up HMM Decoding and Training by Exploiting Sequence Repetitions. In *Proceedings of the 18th annual symposium on Combinatorial Pattern Matching (CPM 2007)*, pages 4–15, London, ON, Canada, 2007. (Cited on page 112.)

[129] D. S. Munteanu and D. Marcu. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *Computational Linguistics*, 31(4):477–504, 2005. (Cited on pages 60, 61, and 66.)

[130] D. S. Munteanu and D. Marcu. Extracting parallel sub-sentential fragments from non-parallel corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics (COLING-ACL 2006)*, pages 81–88, 2006. (Cited on pages 61 and 62.)

[131] M. Nagao. A Framework of a Mechanical Translation between Japanese and English by Analogy Principle. *Artificial and Human Intelligence*, pages 173–180, 1984. (Cited on page 3.)

[132] P. Nakov, C. Liu, W. Lu, and H. T. Ng. The NUS Statistical Machine Translation System for IWSLT 2009. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2009)*, Tokyo, Japan, 2009. (Cited on page 113.)

[133] T. Nguyen, S. Vogel, and N. A. Smith. Nonparametric Word Segmentation for Machine Translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 815–823, 2010. (Cited on pages 113 and 146.)

[134] nist02. The 2002 NIST Open Machine Translation Evaluation Plan (MT02), 2002. (Cited on page 25.)

[135] nist09. The 2009 NIST Open Machine Translation Evaluation Plan (MT09), 2009. (Cited on pages 26 and 27.)

[136] F. J. Och. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, pages 160–167, Sapporo, Japan, 2003. (Cited on pages 23 and 87.)

[137] F. J. Och and H. Ney. Improved Statistical Alignment Models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL 2000)*, pages 440–447, Hong Kong, China, 2000. (Cited on pages 28 and 87.)

[138] F. J. Och and H. Ney. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 295–302, Philadelphia, PA, USA, 2002. (Cited on pages 14 and 23.)

[139] F. J. Och and H. Ney. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(4):417–449, 2004. (Cited on page 14.)

[140] F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. Syntax for Statistical Machine Translation. Technical Report IRCS-00-07, Johns Hopkins 2003 Summer Workshop, Baltimore, MD, USA, 2003. (Cited on

pages 85 and 96.)

[141] F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. A Smorgasbord of Features for Statistical Machine Translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, pages 161–168, Boston, MA, USA, 2004. (Cited on pages 85, 86, and 91.)

[142] A. Oghina, M. Breuss, M. Tsagkias, and M. de Rijke. Predicting IMDb Movie Ratings using Social Media. In *34th European Conference on Information Retrieval (ECIR 2012)*, pages 503–507, 2012. (Cited on page 33.)

[143] T. O'Reilley. What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *Communications and Strategies*, 65:17–38, 2007. (Cited on page 6.)

[144] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 311–318, Philadelphia, PA, USA, 2002. (Cited on pages 26 and 88.)

[145] S. B. Park. Computing Consensus Translation From Multiple Machine Translation Systems. In *In Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU 2001*, pages 351–354, Santa Barbara, CA, USA, 2001. (Cited on page 130.)

[146] R. Parkinson. *Cracking Codes: The Rosetta Stone and Decipherment*. University of California Press, 1999. (Cited on page 4.)

[147] M. Paul, A. Finch, and E. Sumita. Integration of Multiple Bilingually-Learned Segmentation Schemes into Statistical Machine Translation. In *Proceedings of the ACL 2010 Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR (WMT 2010)*, pages 400–408, Uppsala, Sweden, 2010. (Cited on pages 113 and 146.)

[148] S. Petrov. *Coarse-to-fine Natural Language Processing*. PhD thesis, University of California at Berkeley, 2009. (Cited on pages 111 and 112.)

[149] S. Petrov and D. Klein. Improved Inference for Unlexicalized Parsing. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference (ACL-HLT 2007)*, pages 404–411, Rochester, NY, USA, 2007. (Cited on page 112.)

[150] S. Petrov and D. Klein. Sparse Multi-Scale Grammars for Discriminative Latent Variable Parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 867–876, Honolulu, Hawaii, 2008.

[151] S. Petrov and D. Klein. Discriminative Log-Linear Grammars with Latent Variables. In *Proceedings of Advances in Neural Information Processing Systems 20 (NIPS 2008)*, pages 1153–1160, Vancouver, B.C., Canada, 2008. (Cited on page 112.)

[152] S. Petrov, A. Haghighi, and D. Klein. Coarse-to-Fine Syntactic Machine Translation using Language Projections. In *Proceedings of the Conference on Empical Methods In Natural Language Processing (EMNLP 2008)*, pages 108–116, Honolulu, Hawaii, 2008. (Cited on page 112.)

[153] B. Poblete, R. Garcia, M. Mendoza, and A. Jaimes. Do All Birds Tweet the Same? Characterizing Twitter Around the World. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM 2011)*, pages 1025–1030, Glasgow, Scotland, 2011. (Cited on pages 50 and 51.)

[154] M. Post and D. Gildea. Parsers as Language Models for Statistical Machine Translation. In *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas (AMTA 2008)*, pages 172–181, 2008. (Cited on pages 85 and 91.)

[155] M. Pryzbocki. NIST machine translation 2004 evaluation summary of results . In *Machine Translation Evaluation Workshop*, 2004. (Cited on page 27.)

[156] L. R. Rabiner. A Tutorial on Hidden Markov Models and selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989. (Cited on pages 115 and 116.)

[157] S. D. Richardson, W. B. Dolan, A. Menezes, and M. Corston-Oliver. Overcoming the customization bottleneck using example-based MT. In *Proceedings of the workshop on Data-driven Methods in Machine Translation (DMMT 2001)*, pages 1–8, 2001. (Cited on page 4.)

[158] B. Roark, M. Saraclar, and M. Collins. Corrective Language Modeling For Large Vocabulary ASR With The Perceptron Algorithms. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP 2004)*, pages 749–752, Montreal, Canada, 2004. (Cited on page 95.)

[159] B. Roark, M. Saraclar, M. Collins, and M. Johnson. Discriminative Language Modeling with Conditional Random Fields and the Perceptron Algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, pages 47–54, Barcelona, Spain, 2004. (Cited on pages 93 and 94.)

[160] B. Roark, M. Saraclar, and M. Collins. Discriminative n-gram language modeling. *Computer Speech and Language*, 21(2):373–392, 2007. (Cited on pages 86, 93, and 95.)

[161] S. Robertson. Relevance Weighting of Search Terms. *Journal of Documentation*, 60(5):503–520, 1974. (Cited on page 61.)

[162] S. Robertson. Understanding Inverse Document Frequency: On theoretical arguments for IDF. *Journal of the American Society for Information Science*, 27(3):129–146, 2004. (Cited on page 61.)

[163] F. Rosenblatt. The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain. *Neurocomputing: Foundations of Research*, 65:6:386–408, 1958. (Cited on page 94.)

[164] A. M. Rush and M. Collins. Exact Decoding of Syntactic Translation Models through Lagrangian Relaxation. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP 2011)*, pages 26–37, Edinburgh, Scotland, 2011. (Cited on pages 112 and 145.)

[165] A. M. Rush, D. Sontag, M. Collins, and T. Jaakkola. On Dual Decomposition and Linear Programming Relaxations for Natural Language Processing. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP 2010)*, pages 1–11, MIT,MA,USA, 2010. (Cited on page 145.)

[166] N. S, O. Franz, L. G, and N. H. An Evaluation Tool for Machine Translation: Fast Evaluation for MT Research. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*, pages 39–45, Athens, Greece, 2000. (Cited on page 26.)

[167] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors. In *Proceedings of the 19th International Conference on World Wide Web (WWW 2010)*, pages 851–860, Raleigh, NC, USA, 2010. (Cited on page 33.)

[168] G. Salton, editor. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice Hall, Englewood Cliffs, NJ, USA, 1971. (Cited on page 63.)

[169] F. Sánchez-martínez, M. L. Forcada, and A. Way. Hybrid rule-based example based MT: Feeding apertium with sub-sentential translation units. In *Proceedings of the 3rd Workshop on Example-Based Machine Translation*, pages 11–18, Dublin, Ireland, 2009. (Cited on page 4.)

[170] R. E. Schapire. The Strength of Weak Learnability. *Machine Learning*, 5(2):197–227, 1990. (Cited on page 144.)

[171] H. Schwenk. Investigations on Large-Scale Lightly-Supervised Training for Statistical Machine Translation. In *International Workshop on Spoken Language Translation (IWSLT 2008)*, pages 182–189, Hawaii, USA, 2008. (Cited on pages 61, 75, and 79.)

[172] H. Schwenk and Senellart. Translation Model Adaptation for an Arabic/French News Translation System by Lightly-Supervised Training. In *Proceedings of the Twelfth Machine Translation Summit of the International Association for Machine Translation (MT Summit XII)*, pages 38–315, 2009. (Cited on page 61.)

[173] S. L. Scott. Bayesian Methods for Hidden Markov Models: Recursive Computing in the 21st Century. *Journal of the American Statistical Association*, 97:337–351, 2002. (Cited on pages 115 and 116.)

[174] Semiocast. Half of Messages on Twitter are not in English, Japanese is the Second Most Used Lan-

guage. 2010. (Cited on pages 50, 51, and 59.)

[175] L. Shen, A. Sarkar, and F. J. Och. Discriminative Reranking For Machine Translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, pages 177–184, Boston, MA, USA, 2004. (Cited on pages 85, 86, and 94.)

[176] S. M. Siddiqi and A. W. Moore. Fast Inference and Learning in Large-State-Space HMMs. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 800–807, Bonn, Germany, 2005. (Cited on page 112.)

[177] M. Simard, C. Goutte, and P. Isabelle. Statistical Phrase-based Post-editing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2007)*, pages 508–515, 2007. (Cited on page 4.)

[178] N. Singh-Miller and C. Collins. Trigger-Based Language Modeling using a Loss-Sensitive Perceptron Algorithm. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, pages 25–28, Honolulu, Hawaii, 2007. (Cited on page 94.)

[179] A. Smith and J. Brenner. Twitter Use. Technical report, Pew Internet & American Life Project, 2012. (Cited on page 59.)

[180] J. R. Smith, C. Quirk, and K. Toutanova. Extracting Parallel Sentences from Comparable Corpora using Document Level Alignment. In *Proceedings of the Human Language Technologies: The 11th Annual Conference of the North American Chapter of the ACL (HLT-NAACL 2010)*, pages 403–411, Los Angeles, CA, USA, 2010. (Cited on page 61.)

[181] M. Snover, N. Madnani, B. Dorr, and R. Schwartz. TER-Plus: Paraphrase, Semantic, and Alignment Enhancements to Translation Edit Rate. *Machine Translation*, 23:117–127, 2009. (Cited on page 26.)

[182] A. Stolcke. SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference of Spoken Language Processing (INTERSPEECH 2002)*, pages 257–286, Denver, CO, USA, 2002. (Cited on pages 28, 121, and 123.)

[183] A. Stolcke. SRILM – An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP 2002)*, pages 901–904, Denver, CO, USA, 2002. (Cited on page 132.)

[184] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006. (Cited on page 138.)

[185] C. Tillman. A Unigram Orientation Model for Statistical Machine Translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics Companion Volume: Short Papers (HLT-NAACL 2004)*, pages 101–104, Boston, MA, USA, 2004. (Cited on page 18.)

[186] C. Tillman. A Beam-Search Extraction Algorithm for Comparable Data. In *of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*, pages 225–228, Suntec, Singapore, 2009. (Cited on page 61.)

[187] C. Tillman and J.-m. Xu. A Simple Sentence-level Extraction Algorithm for Comparable Data. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies Companion Volume: Short Papers (NAACL-HLT 2009)*, pages 93–96, Boulder, CO, USA, 2009. (Cited on page 61.)

[188] C. Tillman and T. Zhang. A Localized Prediction Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting on Association Computational Linguistics (ACL 2005)*, pages 557–564, Ann Arbor, MI, USA, 2005. (Cited on page 18.)

[189] C. Tillman and T. Zhang. A Discriminative Global Training Algorithm for Statistical MT. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics (ACL-COLING 2006)*, pages 721–728, Sydney, Australia, 2006. (Cited on page 86.)

[190] C. Tillman, S. Vogel, H. Ney, A. Zubiaga, and H. Sawaf. Accelerated Dp Based Search For Statistical Translation. In *Proceedings of the 8th European Conference on Speech Communication and Technology (EUROSPEECH 1997)*, pages 2667–2670, Rhodes, Greece, 1997. (Cited on page 26.)

[191] R. Tromble, S. Kumar, F. Och, and W. Macherey. Lattice Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 620–629, Honolulu, Hawaii, 2008. (Cited on page 130.)

[192] R. W. Tromble and J. Eisner. A Fast Finite-State Relaxation Method for Enforcing Global Constraints on Sequence Decoding. In *Proceedings of the Human Language Technology Conference of the North American Association for Computational Linguistics (HLT-NAACL 2006)*, pages 423–430, New York, NY, USA, 2006. (Cited on page 146.)

[193] Y. Tsuruoka and J. Tsujii. Bidirectional Inference with the Easiest-First Strategy for Tagging Sequence Data. In *In Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP 2005)*, pages 467–474, Vancouver, B.C, Canada, 2005. (Cited on page 112.)

[194] A. Tumasjan, T. Sprenger, P. Sandner, and I. Welpe. Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM 2010)*, pages 178–185, Washington, DC, USA, 2010. (Cited on page 33.)

[195] N. Ueffing, G. Haffari, and A. Sarkar. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pages 25–32, Prague, Czech Republic, 2007. (Cited on page 62.)

[196] J. Van Gael, Y. Saatci, Y. W. Teh, and Z. Ghahramani. Beam Sampling for the Infinite Hidden Markov Model. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, pages 1088–1095, Helsinki, Finland, 2008. (Cited on page 138.)

[197] B. Vauquois. A survey of formal grammars and algorithms for recognition and transformation in mechanical translation. In *Proceedings of International Federation for Information Processing Congress (IFIP 1968)*, pages 254–260, 1968. (Cited on page 2.)

[198] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen. Microblogging During Two Natural Hazards Events: What Twitter May Contribute to Situational Awareness. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems (CHI 2010)*, pages 1079–1088, Atlanta, GA, USA, 2010. (Cited on page 33.)

[199] L. von Ahn and L. Dabbish. Labeling Images with a Computer Game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2004)*, pages 319–326, Vienna, Austria, 2004. (Cited on page 63.)

[200] I. Vulić, W. De Smet, and M.-F. Moens. Identifying word translations from comparable corpora using latent topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers (ACL-HLT 2011)*, pages 479 – 484, Portland, OR, USA, 2011. (Cited on page 145.)

[201] T. Watanabe and E. Sumita. Example-based Decoding for Statistical Machine Translation. In *Proceedings of the Ninth Machine Translation Summit of the International Association for Machine Translation (MT Summit IX)*, pages 410–417, New Orleans, LO, USA, 2003. (Cited on page 4.)

[202] T. Watanabe, J. Suzuki, J. Tsukada, and H. Isozaki. Online Large-Margin Training for Statistical Machine Translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 764–773, Prague, Czech Republic, 2007. (Cited on page 86.)

[203] W. Weaver. Translation. In W. N. Locke and A. D. Boothe, editors, *Machine Translation of Languages: Fourteen Essays*, pages 15–23. The Technology Press of the Massachusetts Institute of Technology, 1955. Reprint of memorandum written in July 1949. (Cited on page 1.)

[204] W. Weerkamp, M. Tsagkias, and S. Carter. How People use Twitter in Different Languages. In

Proceedings of the 3rd International Conference on Web Science (WebSci 2011), Koblentz, Germany, 2011. (Cited on page 11.)

[205] J. Xu, R. Weischedel, and C. Nguyen. Evaluating a probabilistic model for cross-lingual information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2001)*, pages 105–110, New Orleans, LA, USA, 2001. (Cited on pages 78 and 80.)

[206] X. Xue, J. Jeon, and W. B. Croft. Retrieval models for question and answer archives. In *Proceedings of the 31st Annual International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 475–482, Singapore, 2008. (Cited on page 80.)

[207] Y. Yang and X. Liu. A Re-examination of Text Categorization Methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999)*, pages 42–49, Berkeley, CA, USA, 1999. (Cited on page 44.)

[208] D. Yu, S. Wang, Z. Karam, and L. Deng. Language Recognition Using Deep-Structured Conditional Random Fields. In *2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP 2010)*, pages 5030–5033, Dallas, TX, USA, 2010. (Cited on page 36.)

[209] O. F. Zaidan and C. Callison-Burch. The Arabic Online Commentary Dataset: an Annotated Dataset of Informal Arabic with High Dialectal Content. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 37–41, Portland, Oregon, USA, June 2011. (Cited on page 62.)

[210] O. F. Zaidan and C. Callison-Burch. Crowdsourcing Translation: Professional Quality from Non-Professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 1220–1229, Portland, OR, USA, 2011. (Cited on page 62.)

[211] R. Zhang, K. Yasuda, and E. Sumita. Improved Statistical Machine Translation by Multiple Chinese Word Segmentation. In *Proceedings of the ACL 2008 Third Workshop on Statistical Machine Translation (WMT 2008)*, pages 16–223, Columbus, OH, USA, 2008. (Cited on page 113.)

[212] A. Zollmann and A. Venugopal. Syntax Augmented Machine Translation via Chart Parsing. In *Proceedings of the NAACL 2006 Workshop on Statistical Machine Translation (WMT 2006)*, pages 138–141, New York, NY, USA, 2006. (Cited on page 20.)

[213] A. Zollmann, A. Venugopal, F. Och, and J. Ponte. A Systematic Comparison of Phrase-Based, Hierarchical and Syntax-Augmented Statistical MT. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1145–1152, Manchester, England, 2008. (Cited on page 20.)

Samenvatting

Al snel na het ontstaan van de computerwetenschappen onderkenden onderzoekers het grote belang van machinaal vertalen, maar het is pas recentelijk dat er algoritmes zijn die in staat blijken om automatische vertalingen te genereren die bruikbaar zijn voor het grote publiek. Moderne vertaalsystemen zijn afhankelijk van tweetalige corpora, het moderne equivalent van de Steen van Rosetta, van waaruit ze kruisverbanden tussen die talen kunnen leren om die vervolgens te gebruiken om zinnen die niet in het trainingscorpus zitten te vertalen. Deze tweetalige data is essentieel en als die ontoereikend is of buiten het domein valt, dan neemt de vertaalkwaliteit af. Om de vertaalkwaliteit te verbeteren moeten we zowel de methodes perfectioneren die bruikbare vertalingen extraheren uit additionele meertalige bronnen, als de samengestelde modellen van een vertaalsysteem verbeteren om zo beter gebruik te kunnen maken van bestaande meertalige datasets.

In dit proefschrift concentreren we ons op deze twee gerelateerde problemen. Onze aanpak is tweeledig en het proefschrift volgt dan ook die structuur. In deel I bestuderen we het probleem van het extraheren van vertalingen van het web, met een focus op het gebruik van de groeiende populariteit van microblogplatformen. We presenteren nieuwe methodes voor taalherkenning in microblogberichten en maken een uitgebreide analyse van bestaande methodes die deze berichten doorzoeken op nieuwe vertalingen. In deel II bestuderen we het gerelateerde probleem van het verbeteren van taalmodellen voor het herordenen van top-x-lijsten en een morfologische analyse aan de bronkant. We starten met een analyse van een groot aantal syntactische kenmerken voor het herordenen van de top-x-lijsten die door automatische vertaalsystemen worden gegenereerd. Vervolgens presenteren we een nieuw algoritme dat exacte gevolgtrekkingen door hoge orde hidden Markov modellen mogelijk maakt, wat we vervolgens gebruiken voor het

segmenteren van bronteksten. Dit proefschrift geeft inzicht in het verzamelen van relevante trainingsdata en presenteert nieuwe modellen die beter gebruikmaken van bestaande meertalige corpora.