Training and Models for Statistical Machine Translation

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Diplom–Informatiker Arne Mauser

aus

Groß-Umstadt

Berichter:
Professor Dr.–Ing. Hermann Ney
Professor Dr. Stefan Riezler

Tag der mündlichen Prüfung: 6. November 2015

Acknowledgments

Many people have accompanied and inspired me along this journey and I am very grateful. While any attempt in listing everyone will be incomplete, I would like to express my gratitude to the following people:

Hermann Ney for his academic guidance and teaching relentless critical thinking. The environment and opportunities he offered me were truly inspiring and formative.

Stefan Riezler for honoring me with his constructive in-depth advice.

Nicola Ueffing, who took me in as a student and introduced me to the topics that are covered in this thesis. Evgeny Matusov who then supervised my Diploma Thesis and later became my office-mate and co-author.

I would also like to thank my colleagues, co-authors, and co-competitors for being inspiring, sharing their knowledge, and the good times: Björn Hoffmeister, Carmen Heger, Christian Buck, Christian Gollan, Christian Kohlschein, Daniel Keysers, David Rybach, David Vilar, Gregor Leusch, Jan-Thorsten Peter, Joern Wuebker, Malte Nuhn, Markus Freitag, Matthias Huck, Minwei Feng, Oliver Bender, Patrick Dötsch, Pavel Golik, Ralf Schlüter, Richard Zens, Saša Hasan, Saab Mansour, Shahram Khadivi, Stefan Hahn, Stefan Peitz, Thomas Deselaers, Volker Steinbiss, Yuqi Zhang, Zoltán Tüske.

On the final stretches of this journey it was Loredana who nudged me into the right direction and Tara who helped me be determined with the last corrections.

Finally, the big thanks goes to my parents and sister for their support all the way.

Abstract

Machine translation is the application of machines to translate text or speech from one natural language to another. Nowadays machines typically refer to computers and computer software.

One common algorithmic approach to machine translation is the statistical phrase-based translation model. With statistical models, the two key aspects are the model itself and the training thereof. This work describes and analyzes both these aspects.

We will present an algorithm and procedure for consistent training of phrase-based statistical translation models. The training involves a forced alignment method, where a phrase alignment is found between the source training instances and the given translation, using all models and components of the translation process. We will show that this leads to smaller models with higher translation quality.

We will investigate different smoothing techniques that improve the generalization of our translation models when applied to previously unseen, out-of-domain test data. We introduce phrase count features and consistently trained word-lexicon models. Both attempts result in improvements in translation quality.

We will show models that use machine learning components to improve lexical selection. In order to train these proposed extended lexicon models, we will use efficient algorithms and data structures for training large-scale sparse maximum entropy models. These methods reduce the time and memory required to train the extended lexicon models by several orders of magnitude.

By combining this work on consistent training of phrase-based and efficient machine-learning we show the feasibility of a full discriminative model for phrase-based statistical machine translation.

The resulting machine translation systems improve translation quality on a range of tasks and language pairs, tested in public evaluation campaigns.

Kurzfassung

Maschinelle Übersetzung ist die Nutzung von Maschinen, um Text oder Sprache von einer natürlichen Sprache in eine andere zu übersetzen. Heutzutage sind diese Maschinen in der Regel Computer und Software.

Ein verbreiteter algorithmischer Ansatz in der maschinellen Übersetzung ist das statistische phrasen-basierte Übersetzungsmodell. Zwei Hauptaspekte in statistischen Modellen sind zum einen das Modell selbst und zum anderen das Training dieses Modells. Diese Arbeit beschreibt und analysiert diese beiden Aspekte.

In dieser Arbeit wird ein Algorithmus für das konsistente Training von phrasen-basierten statistischen Übersetzungsmodellen vorgestellt. Diese Art von Training benötigt eine Methode zur erzwungenen Alignierung, bei der eine Phrasenalignierung zwischen Quell- und Zielsatz der Trainingsbeispiele gefunden wird, die alle Modellkomponenten des phrasen-basierten Modells benutzt. Es wird gezeigt, dass damit kleinere Modelle und bessere Übersetzungsqualität erreicht werden können.

Darüber hinaus werden in dieser Arbeit unterschiedliche statistische Glättungsmethoden untersucht, die die Verallgemeinerung der Übersetzungsmodelle auf ungesehenen Daten verbessern. Es werden Merkmale basierend auf Phrasenhäufigkeit vorgestellt und trainierte Wortlexikonmodelle. Beide Ansätze zeigen Verbesserungen in der Übersetzungsqualität.

Es wird gezeigt, dass maschinelles Lernen in erweiterten Lexikonmodellen die Wortwahl in der maschinellen Übersetzung verbessern können. Um die erweiterten Lexikonmodelle zu trainieren, werden effiziente Algorithmen und Datenstrukturen benutzt, die das Training großer, dünn besetzter Modelle ermöglichen. Diese Methoden verkürzen die Trainingszeit und den benötigten Hauptspeicher um mehrere Größenordnungen.

Als eine Kombination der beiden Hauptaspekte dieser Arbeit, dem konsistenten Training von phrasen-basierten Übersetzungsmodellen und effizienten Algorithmen für maschinelles Lernen von dünn besetzten Modellen, wird die Machbarkeit eines gänzlich diskriminativen phrasen-basierten Übersetzungsmodells gezeigt.

In der Analyse der vorgestellten Methoden wird auf einer Reihe von Korpora und Sprachpaaren verbesserte Übersetzungsqualität gezeigt. Einige der Methoden sind auch in unabhängigen, öffentlichen Evaluationen und Wettbewerben erfolgreich getestet worden.

Inhaltsverzeichnis

1 Introduction

This work is based in part on already published research results and is organized as follows. We start with introductory chapters that describe the necessary foundations of machine translation (Chapters 1 and 3) that we will refer to in later chapters and describe the goals of this work in Chapter 2.

Chapter 4 is an extension of [Wuebker & Mauser$^+$ 10]. In addition to what was presented in that paper, we extend the description of the algorithm and experimental evaluation. Some parts of Chapter 5 have been published in [Mauser & Zens$^+$ 06], while other parts of Chapters 5 and 6 extend the idea and infrastructure of [Wuebker & Mauser$^+$ 10] to train more than just the phrase table and do consistent training for more components of the translation system. Much of the material from these two chapters has not yet been published.

Chapter 7 builds on one of the models presented in [Mauser & Hasan$^+$ 09] and extends the description and analysis of the model and its effect on machine translation quality. The idea of using discriminative models to guide machine translation is then extended into previously unpublished material in Chapter 8.

After a description of additional results and the data used for the experiments in Chapter 9, we conclude with a brief summary and examine potential future directions.

1.1 Machine Translation

Machine translation is the application of machines to translate text or speech from one natural language to another. Nowadays machines typically refer to computers and computer software.

With the internet, global commerce and migration, our world has become increasingly multilingual. Economic globalization has lead to business and trade being done across nations, continents, cultures and languages. In Europe, many nations have formed a union with a common market, harmonized laws and even a common currency. The European Union with its 28 member states has 24 official languages. The EU not only needs official documents like laws, resolutions and the transcripts of the sessions of the European parliament to be available in all 24 languages. The open markets also require multilingual correspondence, product descriptions, user manuals, contracts. In 2013 the European Commission alone translated 2.02 million pages, each containing 1500 typed characters, costing about 300 million Euro (2011 number).[1]

It is not on the European Union that generates a need for translation. Other parts of the world are becoming increasingly important as trade partners, offering markets with

[1] `http://ec.europa.eu/dgs/translation/faq/index_en.htm`

growth rates far beyond the possibilities of developed countries. In 2010, China was the third most important trade partner for Germany with a volume of about 130 billion Euro. For the United States of America, China is the second largest single trade partner with a volume of 456 billion US Dollars in 2010[2]. German and English language skills are not common in China[3], nor is the mandarin Chinese language widespread in Germany or other parts of the Western world. All this trade involves communication in written and spoken form and creates a strong demand for translation services.

Considering the rate of evolution in computer science, Machine Translation already has a long history. The first ideas for machine translation emerged with the rise of the industrial age and the progressing use of mechanical devices to perform work. In the first half of the 20th century, there were already more sophisticated and concrete ideas of how these machines could work. Actual systems, however that could perform small translation tasks were not available until the late 1970s. The first successful approaches, based on hand-written rules and lexica, still required a large amount of manual work. Until the 1990s, machine translation research was dominated by these approaches. Then, refined statistical models and increasing computing power gave rise to numerous new approaches.

Machine translation might be one of the hardest problems in natural language processing and maybe in computer science as a whole. Translation is hard from three completely different perspectives.

First, translation is always to some degree an interpretation of the original text. Some lexical and grammatical constructs are not available in the target language and this must be described or transformed. Also, the implied cultural context can be difficult to translate. There have been arguments such as the "Chinese Room" [Searle 80] that translation is not possible without world knowledge and true understanding of the meaning of the translated text.

The interpretation aspect of translation leads to another, more technical challenge: in most cases, there is no unique "correct translation." Even though there might be a distinction between adequate and inadequate translations, in general the decision is often subjective. There might be hundreds or even thousands of adequate translations for a single sentence, using different wording, word order, style or tone. Experiments show that when presented with automatic translations, human judges often disagree on the quality of translation hypotheses [Callison-Burch & Koehn$^+$ 10]. A key ingredient to the scientific process is measuring. In the case of translation, measuring and comparing the quality of translation output is a challenge by itself. While several automatic evaluation measures have been proposed and some accepted by the scientific community, the problem of evaluation remains an active field of research [Callison-Burch & Koehn$^+$ 10].

A purely technical reason for the difficulty of machine translation lies in the structure of the problem itself. Not only mapping the vocabulary changes from one language to another, but also the word order. It has been shown that this makes MT an NP-hard problem [Knight 99].

Translation as a problem is hard; language is ambiguous, it is hard to evaluate, and it is

[2] `http://www.census.gov/foreign-trade/balance/`

[3] According to Wikipedia, less than 1% of the Chinese population speaks English `http://en.wikipedia.org/wiki/List_of_countries_by_English-speaking_population`

computationally complex. Nevertheless machine translation has been a field of research for over 60 years and is currently more active than ever.

1.2 The Statistical Approach to Machine Translation

From a statistical point of view, the translation task can be addressed as a decision problem. For a given source sentence, the most probable translation is to be selected. The true mechanisms involved in the translation process are not known. The design of a statistical translation system is dominated by finding suitable models to reflect the linguistic phenomena in the parallel training data.

We assume that the reader is familiar with the very basics of probability theory such as conditional and joint distributions, priors and posteriors.

As we follow [Brown & Della Pietra$^+$ 93] in our description, we name the source and the target language sentences by f and e respectively, for the translation languages French and English processed in the aforementioned publication. We would like to estimate a statistical model for the probability that a hypothesized target language sentence was generated by a given source sentence. For every possible pair of source sentence $f_1^J = f_1, \ldots, f_j, \ldots f_J$ and target sentence $e_1^I = e_1, \ldots, e_i, \ldots e_I$, we assign a probability $Pr(e_1^I|f_1^J)$ that a human interpreter would generate e_1^I from f_1^J. Using Bayes' rule, we can decompose this probability into

$$Pr(e_1^I|f_1^J) = \frac{Pr(e_1^I)Pr(f_1^J|e_1^I)}{Pr(f_1^J)}\,. \tag{1.1}$$

The target sentence $\hat{e}_1^{\hat{I}}$ with the highest probability $Pr(e_1^I|f_1^J)$ provides the "best" translation, i.e. the translation where we expect to make the least sentence errors. Since the denominator has no influence on the decision about the best target sentence, it suffices to maximize $Pr(e_1^I)Pr(f_1^J|e_1^I)$ over all target sentences e_1^I. This leads to

$$\hat{e}_1^{\hat{I}} = \underset{e_1^I, I}{\operatorname{argmax}}\, Pr(e_1^I)Pr(f_1^J|e_1^I)\,. \tag{1.2}$$

Equation 1.2 implies three subtasks involved in machine translation: (1) the estimation of the language model probability $Pr(e_1^I)$, (2) the translation probability $Pr(f_1^J|e_1^I)$, and (3) searching for the target sentence that maximizes their product. Figure 1.1 shows the general architecture of a statistical MT system modeled using Bayes' decision rule.

For modeling a translation process that assigns source words to their corresponding target words [Brown & Della Pietra$^+$ 93] introduce the alignment A as a hidden variable. An alignment is defined as a set of correspondences between words in parallel texts. The translation probability is then:

$$Pr(f_1^J|e_1^I) = \sum_A Pr(f_1^J, A|e_1^I)\,. \tag{1.3}$$

An alignment describes a mapping of source sentence words to target sentence words. A common representation of these correspondences is an *alignment matrix* (Figure 1.2).

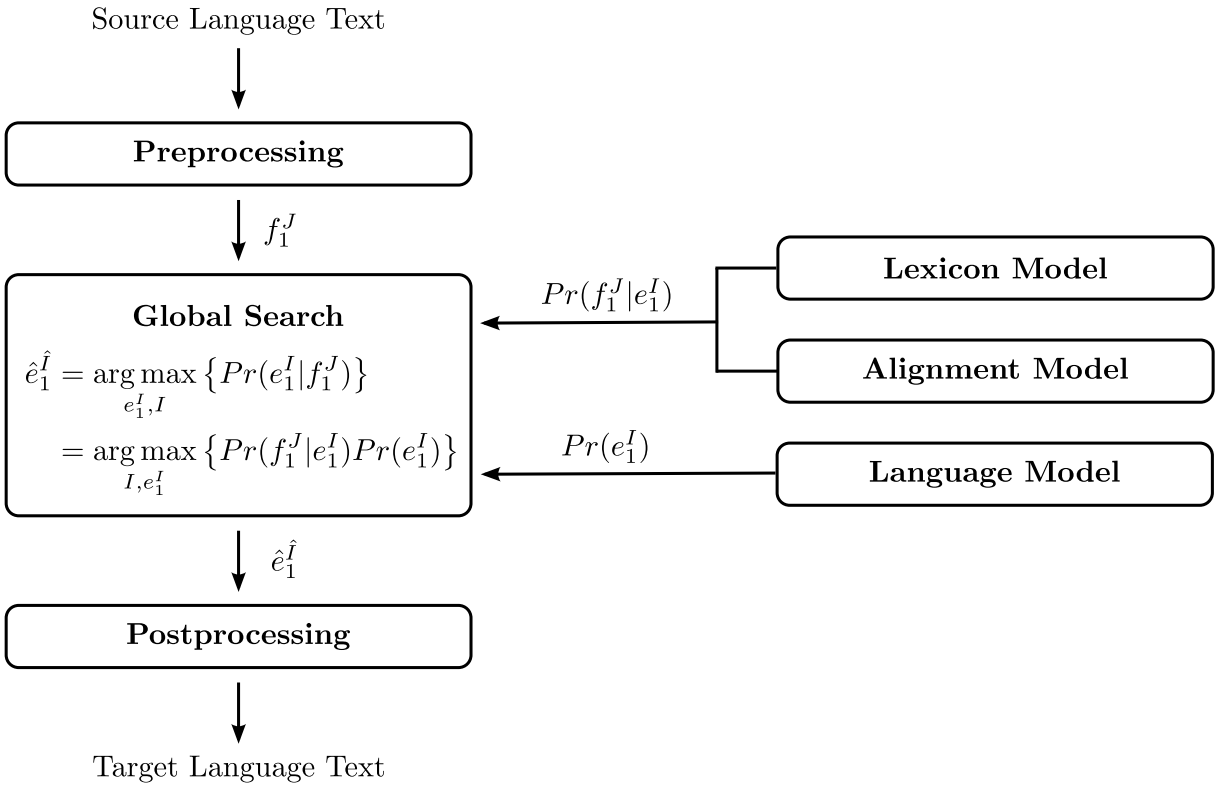


Figure 1.1: The architecture of a statistical machine translation system. A given source language text is transformed in the preprocessing step. For finding the best translation, search is carried out over all possible translations, incorporating statistical models for the lexicon, alignment and the target language. After a final transformation step, the system provides a target language output.

Even for humans, it is often difficult to give precise information on word correspondence. Difficulties may arise from idiomatic expressions, loose translations, and/or missing function words. The underlying problem is the subjective notion of "correspondence" in these cases. Typical effects of bilingual alignments like reorderings, omissions and one-to-many phrasal alignments further complicate the alignment task. These phenomena require a very general representation of an alignment. For a given source language string f_1^J and a target language string e_1^I [Och & Ney 03] define an alignment A as the Cartesian product of the word positions that is

$$A \subseteq \{(i, j) : \ j = 1, \ldots, J; i = 1; \ldots, I\}$$

Technically, there are 2^{IJ} different alignments. In this general formulation, it is difficult to deal with the alignment task algorithmically. Therefore, the alignment models typically impose restrictions on the alignment representation. While some approaches require that a source word is aligned to *exactly* one target word, others define alignments as a mapping from source-to-target language positions. By restricting the alignments to be one-to-one correspondences between non-empty words, a further simplification is

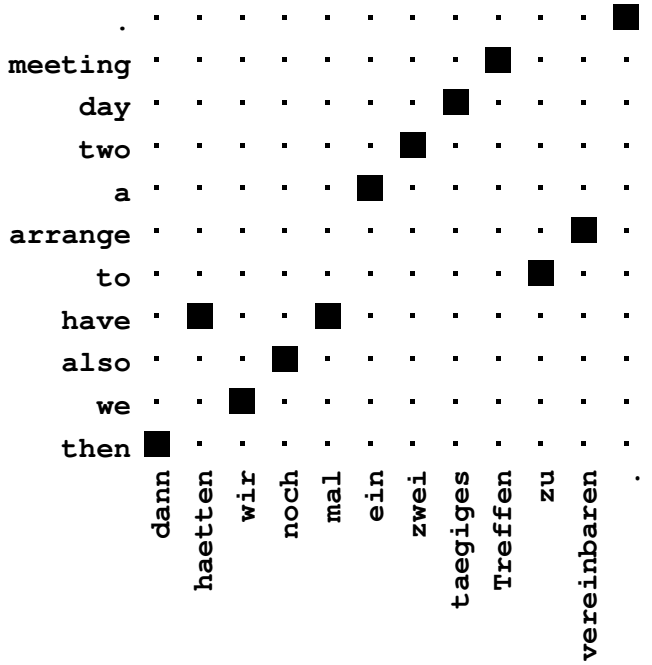


Figure 1.2: Example alignment matrix from the VERBMOBIL corpus (See Section 9.2 for a description of the corpora used in this work.). Coverage constraints are enforced for source and target words. This results in the German expletive "mal" being aligned to "have".

achieved [Melamed 00]. Some linguistic phenomena however can not be captured under this restriction.

Without loss of generality, the translation probability from Equation 1.3 can be further decomposed as:

$$Pr(f_1^J, A|e_1^I) = Pr(J|e_1^I) \cdot Pr(A|e_1^I, J) \cdot Pr(f_1^J|A, e_1^I, J) \tag{1.4}$$

This results in three distinct models: (1) The sentence length model $Pr(J|e_1^I)$, (2) the alignment model $Pr(A|e_1^I, J)$, and (3) the lexicon model $Pr(f_1^J|A, e_1^I, J)$. The sentence length is usually not modeled explicitly. Below, we present some of the most common approaches in modeling the lexicon and alignment probability distributions.

1.2.1 Single-Word based Models

Some of the models we will propose in subsequent chapters, will make use of the modeling that is introduced with single-word based translation models. We therefore take some time in this section to introduce the notation, concepts and common incarnations of single-word based models.

The alignment in single-word based models is usually quite restricted. An alignment is seen as a function $a : \{1, \ldots, J\} \mapsto \{0, \ldots, I\}$ that assigns a target sentence word to each word of the source sentence. Most models allow for a relaxation of this constraint, where

there is an abstract *empty word* e_0 in the target sentence. A source word aligned to e_0 has no correspondence in the target sentence and is not translated. The notation of this kind of alignment usually is $a_1^J := a_1, \ldots, a_j, \ldots, a_J$. The source word f_j is aligned to the target word e_{a_j}. This form of functional alignments allows for a convenient factorization of the lexicon probability:

$$Pr(f_1^J|a_1^J, e_1^I, J) = \prod_{j=1}^{J} Pr(f_j|f_1^{j-1}, a_1^J, e_1^I, J) \tag{1.5}$$

$$\cong \prod_{j=1}^{J} p(f_j|e_{a_j}) \,. \tag{1.6}$$

In single-word based modeling, the distribution is reduced to the probability of the source word given the aligned target word. No context information is taken into account in the lexicon model. The lexicon probability is independent of the surrounding source words or previously aligned target words. No target word can be aligned to more than one source word.

A thorough comparison of many single word based alignment models can be found in [Och & Ney 03]. They compare the models proposed by [Brown & Della Pietra$^+$ 93], the Hidden-Markov-Model [Vogel & Ney$^+$ 96] and a heuristic approach with respect to alignment quality. Here we just give a short sketch of the most important models. The first three models, IBM1, IBM2, and HMM, factorize the alignment probability as:

$$Pr(a_1^J|e_1^I, J) = \prod_{j=1}^{J} Pr(a_j|a_1^{j-1}, e_1^I, J) \tag{1.7}$$

The models differ in their assumptions for the alignment model.

1.2.1.1 IBM1 [Brown & Della Pietra$^+$ 93]

[Brown & Della Pietra$^+$ 93] proposed *Model 1*. In the literature, it is usually referred to as IBM1. Being a relatively simple model, it has some computational advantages and has found application in the field of natural language processing. Examples are lexicon construction [Wu & Xia 94], idiom detection [Melamed 97] and cross-language information retrieval.

In IBM1, the alignment probability is modeled with a uniform distribution of all target positions

$$Pr(a_j|a_1^{j-1}, e_1^I, J) = p(a_j|I) = \frac{1}{I+1} \,. \tag{1.8}$$

While using a uniform distribution for the alignment probability, the translation only relies on the lexicon model. The computational advantage of the model lies in the global optimum when using maximum likelihood training with the EM algorithm [Dempster & Laird$^+$ 77]. This ensures the independence of the result of the optimizations from the initial parameter settings.

1.2.1.2 IBM2 [Brown & Della Pietra+ 93]

In IBM2, alignment probabilities are conditioned on the lengths of the source sentence J and the target sentence, and I, and the source sentence position j

$$Pr(a_j|a_1^{j-1}, e_1^I, J) = p(a_j|j, I, J) \tag{1.9}$$

1.2.1.3 Hidden Markov Model for Word Alignment [Vogel & Ney+ 96]

In the HMM alignment model, the conditioning on the source sentence position j is substituted by the previously aligned position a_j

$$Pr(a_j|a_1^{j-1}, e_1^I, J) = p(a_j|a_{j-1}, I, J)\,. \tag{1.10}$$

This corresponds to a first-order hidden Markov model.

1.2.1.4 IBM3 [Brown & Della Pietra+ 93]

Model 3 introduces the concept of *fertility*. Fertility ϕ_i is the number of source words aligned to the target word i

$$\phi_i := \sum_{j=1}^{J} \delta(a_j, i)\,. \tag{1.11}$$

The fertility in IBM3 is modeled as a probability distribution $p(\phi|e)$, the probability that e corresponds to exactly ϕ source words. The decomposition of the alignment probability differs from IBM1 and IBM2. Instead of source positions, factorization is carried out over the target positions. The alignment representation is augmented to sets b_i, consisting of the source positions aligned to the target word e_i. From the column-wise alignment representation a_1^J, we now define a row-wise alignment b_0^I

$$b_i := \{j|a_j = i\}\,. \tag{1.12}$$

The definition of the alignments a_1^J as a function of the source words ensures that the b_i form a partition of the source positions. Since the alignments a_1^J were not constrained with respect to the target positions, each of these sets, b_i, can have a cardinality greater than 1. The cardinality of b_i is the fertility of e_i

$$\phi_i = |b_i|\,. \tag{1.13}$$

In the decomposition of the translation probability $Pr(f_1^J|e_1^I)$ for Model 3, the alignment is now represented by b_i as hidden variable:

$$Pr(f_1^J|e_1^I) = \sum_{b_0^I} Pr(f_1^J, b_0^I|e_1^I) \tag{1.14}$$

$$= \sum_{b_0^I} Pr(b_0^I|e_1^I) \cdot Pr(f_1^J|e_1^I, b_0^I) \tag{1.15}$$

The sentence length is given by the target word fertilities.

$$J = \sum_{i=0}^{I} \phi_i \,. \tag{1.16}$$

While the lexicon probability $Pr(f_1^J|e_1^I, b_0^I)$ is still modelled without context information, the alignment probability is decomposed as

$$Pr(b_0^I|e_1^I) = Pr(b_0|b_1^I, e_1^I) \cdot \prod_{i=1}^{I} Pr(b_i|b_1^{i-1}, e_1^I). \tag{1.17}$$

$$= p(b_0|b_1^I) \cdot \prod_{i=1}^{I} p(b_i|e_i) \tag{1.18}$$

$$= p(\phi_0|\phi_1^I) \cdot \prod_{i=1}^{I} p(\phi_i|e_i) \cdot \phi_i! \cdot \prod_{j \in b_i} p(j|i, J) \,. \tag{1.19}$$

The fertility of the empty word is modelled separately from the other words, conditioned on the overall fertility.

1.2.2 IBM4 [Brown & Della Pietra$^+$ 93]

IBM4 introduces more context dependencies into the alignment model. Every word depends on the previously aligned word and the word classes of the surrounding words. There are two different alignment models: (1) $p_{=1}(\delta j| \cdots)$ for the first word in each set b_i, and (2) $p_{>1}(\delta j| \cdots)$ for the subsequent words from left to right.

$$p(b_i|b_{i-1}, e_i) = p(\phi_i|e_i) \cdot p_{=1}(b_{i1} - \overline{b_{\rho(i)}}| \cdots) \prod_{k=2}^{\phi_i} p_{>1}(b_{i\ k} - b_{i,k-1}| \cdots) \tag{1.20}$$

The function $\rho(i)$ gives the largest $i' < i$ with $|b_{i'}| > 0$ and $\overline{b_{\rho(i)}}$ is the average of all elements in $b_{\rho(i)}$. The details of the model are omitted here. See [Brown & Della Pietra$^+$ 93] for a detailed description of IBM4.

The probabilities for the models are obtained in iterative training procedures, usually with up to ten iterations for each model. The IBM models are intended to be trained in sequence. Complex models require initialization with the probabilities obtained by the basic models. Depending on the language pairs, varieties of the order of models have shown to be useful [Och & Ney 03].

1.2.3 Bayes Decision Rule For Machine Translation

In this model, among all possible target language sentences, we choose the sentence with the highest probability:

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\operatorname{argmax}} \left\{ Pr(e_1^I|f_1^J) \right\} \tag{1.21}$$

$$= \underset{I,e_1^I}{\operatorname{argmax}} \left\{ Pr(e_1^I) \cdot Pr(f_1^J|e_1^I) \right\} \tag{1.22}$$

This decomposition into two knowledge sources is known as the source-channel approach to statistical machine translation [Brown & Cocke$^+$ 90]. It allows for independent modeling of the target language model $Pr(e_1^I)$ and the translation model $Pr(f_1^J|e_1^I)$[4].

The target language model describes the well-formedness of the target language sentence. The translation model links the source language sentence to the target language sentence. The argmax operation denotes the search problem, i.e., the generation of the output sentence in the target language.

[4]The notational convention will be as follows: we use the symbol $Pr(\cdot)$ to denote general probability distributions with (nearly) no specific assumptions. In contrast, for model-based probability distributions, we use the generic symbol $p(\cdot)$.

2 Scientific Goals

- **Consistent, integrated training for phrase-based statistical machine translation.**

  In previous work, a phrase-based statistical machine translation system has been a combination of individually trained models: language model, translation model, reordering models, etc.. We present a consistent, integrated training procedure that allows the training of multiple models involved in the translation process simultaneously. This makes the training consistent with the translation, yields significant improvements in the translation quality, and reduces the model size. We give a detailed formulation of the training problem for phrase-based statistical machine translation and analyze the training algorithm in detail.

- **Better generalization for phrase-based models.**

  Translations models are trained without generalization in mind. Existing methods to improve translation model generalization use additional models or simple heuristics. We develop consistently-trained phrase table smoothing methods that improve generalization in translation as well as in training.

- **Smaller translation systems.**

  In a common setup, large portions of the phrase table are never used in translation. When phrases are extracted and counted from word-aligned data, there is no indication beyond relative frequency of how useful a phrase will be later in the translation process. Using our consistent phrase model training procedure, we show that we can reduce the size of the phrase table to a fraction of the original size while maintaining or even improving translation quality.

- **Improved reordering.**

  Reordering in translation is not only a computationally hard problem, it is also a challenge in modelling and training. We propose improved reordering models that are trained in a consistent training framework that allows it to jointly learn translation and reordering models. We propose different types of models and compare against existing implementations.

- **Improved lexical selection by learning fine-grained lexical models.**

  Conventional phrase-based statistical machine translation systems only use limited, local context for translation. We propose an additional model that uses full source context for translating and disambiguating translations, and show improvements in translation quality.

- **Efficient representation and training for sparse log-linear models.**

  We develop a sparse model representation that allows training with a very large number of features and classes. This representation makes our models feasible for machine translation. We compare several techniques for pruning and feature selection.

- **Conditional Random Fields (CRFs) for machine translation**

  We apply CRFs to the machine translation problem and implement efficient approximate training and search methods using lattices. The proposed method allows the integration of a large numbers of overlapping features in the translation model. The CRF model shows improvements over baseline phrase-based models for text translation and translation related tasks.

3 Phrase-Based Statistical Machine Translation

3.1 Introduction

In this chapter, we continue from the general Bayes decomposition presented in Section 1.2.3, and briefly describe the phrase-based translation model that will be used throughout this work.

3.1.1 Log-linear model

A generalization of the classical source-channel approach is the direct modeling of the posterior probability $Pr(e_1^I|f_1^J)$. Using a log-linear model [Och & Ney 01], we obtain:

$$Pr(e_1^I|f_1^J) = \frac{\exp\left(\sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J)\right)}{\sum_{e'^{I'}_1} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(e'^{I'}_1, f_1^J)\right)} \tag{3.1}$$

The denominator represents a normalization factor that depends only on the source sentence f_1^J. Therefore, we can omit it during the search process. As a decision rule, we obtain:

$$\hat{e}_1^{\hat{I}} = \underset{I, e_1^I}{\operatorname{argmax}} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J) \right\} \tag{3.2}$$

This is a generalization of the source-channel approach. It has the advantage that additional models $h(\cdot)$ can be easily integrated into the overall system. The model scaling factors λ_1^M are trained with respect to the final translation quality measured by an error criterion using lattice-based minimum error rate training [Macherey & Och$^+$ 08].

3.1.2 Phrase-Based Approach

The basic idea of phrase-based translation is to segment the given source sentence into phrases, then translate each phrase and finally compose the target sentence from these phrase translations. This idea is illustrated in Figure 3.1. Formally, we define a segmentation of a given sentence pair (f_1^J, e_1^I) into K blocks:

$$k \rightarrow s_k := (i_k; b_k, j_k), \text{ for } k = 1 \ldots K. \tag{3.3}$$

Here, i_k denotes the last position of the k^{th} target phrase; we set $i_0 := 0$. The pair (b_k, j_k) denotes the start and end positions of the source phrase that is aligned to the k^{th} target phrase; we set $j_0 := 0$. Phrases are defined as non-empty contiguous sequences of words. We constrain the segmentations so that all words in the source and the target sentence are covered by exactly one phrase. Thus, there are no gaps and there are no overlaps. Note that the segmentation s_1^K contains the information on the phrase-level reordering.

For a given sentence pair (f_1^J, e_1^I) and a given segmentation s_1^K, we define the bilingual phrases as:

$$\tilde{e}_k := e_{i_{k-1}+1} \dots e_{i_k} \tag{3.4}$$

$$\tilde{f}_k := f_{b_k} \dots f_{j_k} \tag{3.5}$$

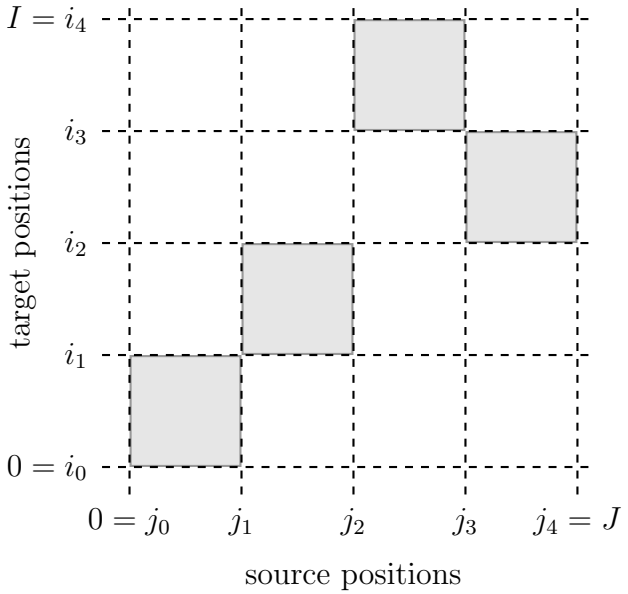


Figure 3.1: Illustration of the phrase segmentation.

The segmentation s_1^K is introduced as a hidden variable in the translation model. The models $h(\cdot)$ depend not only on the sentence pair (f_1^J, e_1^I), but also on the segmentation s_1^K, i.e., we have models $h(f_1^J, e_1^I, s_1^K)$. To compute the normalized posterior probability of a translation e_1^I, a sum over all possible translations and segmentations has to be carried out. In practice, we use the maximum approximation for the sum over the segmentations.

3.1.3 Source cardinality synchronous search

For single-word based models, source cardinality synchronous search is described in [Tillmann & Ney 03]. The idea is that the search proceeds synchronously with the cardinality of the already translated source positions. Here, we use a phrase-based version

of this idea. To make the search problem feasible, the reorderings are constrained as in [Zens & Ney$^+$ 04].

3.2 Models used during search

When searching for the best translation for a given input sentence, we use a log-linear combination of several models (3.1) as decision criterion. In this section, we will describe the models that are used in translation. More specifically the models are: (1) a phrase translation model, (2) a word-based translation model, (3) word and phrase penalty, (4) a target language model and a (5) reordering model.

3.2.1 Phrase-based model

The phrase-based translation model is the main component of our translation system. The hypotheses are generated by concatenating the target language phrases. The corresponding pairs of source and corresponding target phrases are extracted from the word-aligned bilingual training corpus by the phrase extraction algorithm described in detail in [Zens & Och$^+$ 02]. The main idea is to extract phrase pairs that are consistent with the word alignment that is the words of the source phrase are aligned only to words in the target phrase, and vice versa. This criterion is identical to the alignment template criterion described in [Och & Tillmann$^+$ 99]. Formally, given an alignment A containing alignments (j, i), a phrase $(f_{j_1}^{j_2}, e_{i_1}^{i_2})$ is consistent with the alignment if:

$$\begin{aligned} & \forall (j,i) \in A : j_1 \leq j \leq j_2 \leftrightarrow i_1 \leq i \leq i_2 \\ \wedge \quad & \exists (j,i) \in A : j_1 \leq j \leq j_2 \wedge i_1 \leq i \leq i_2. \end{aligned}$$

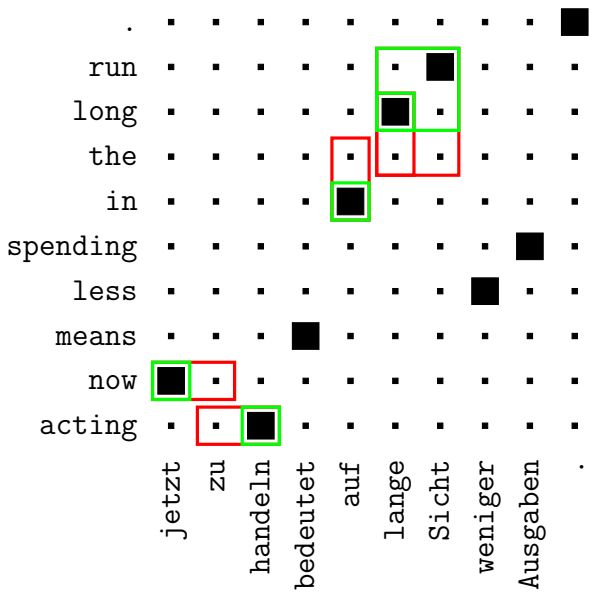
We use relative frequencies to estimate the phrase translation probabilities:

$$p_{\mathrm{H}}(\tilde{f}|\tilde{e}) = \frac{N_{\mathrm{H}}(\tilde{f}, \tilde{e})}{N_{\mathrm{H}}(\tilde{e})} \tag{3.6}$$

Here, the number of co-occurrences of a phrase pair $(\tilde{f}, \tilde{e})$ that are consistent with the word alignment is denoted as $N_{\mathrm{H}}(\tilde{f}, \tilde{e})$. We use the subscript H to distinguish these heuristic phrase counts and probabilities from other estimation methods presented later. If one occurrence of a target phrase $\tilde{e}$ has more than one possible translation, each contributes to $N_{\mathrm{H}}(\tilde{f}, \tilde{e})$, with $1/N_{\mathrm{Hcand}}$ with N_{Hcand} being the number of candidates. This occurs, when not all words in a sentence are aligned. Figure 3.2 illustrates situations where fractional counts can occur.

The marginal count $N_{\mathrm{H}}(\tilde{e})$ is the number of occurrences of the target phrase $\tilde{e}$ in the training corpus. The resulting feature function is:

$$h_{\mathrm{Phr}}(f_1^J, e_1^I, s_1^K) = \log \prod_{k=1}^{K} p(\tilde{f}_k|\tilde{e}_k) \tag{3.7}$$



phrase	translations	count
now	jetzt	0.5
	jetzt zu	0.5
acting	handeln	0.5
	zu handeln	0.5
auf	in	0.5
	in the	0.5
lange	long	0.5
	the long	0.5
lange Sicht	long run	0.5
	the long run	0.5

Figure 3.2: Example for the heuristic treatment of unaligned words in phrase extraction. Only the phrases that are affected by the unaligned word heuristic are shown. The green boxes show phrases that would be extracted without any extension across unaligned words, and the red boxes show the extension of the phrases to include the unaligned words.

To obtain a more symmetric model, we use the phrase-based model in both directions $p(\tilde{f}|\tilde{e})$ and $p(\tilde{e}|\tilde{f})$.

3.2.2 Phrase Count Features

The reliability of the phrase probability estimation is largely dependent on the amount and quality of the training data. Generally, the probability of rare phrases tends to be over-estimated, but as they do not occur often, they might be errors originating from mistranslations or erroneous word alignments. Therefore, we also include features based on the actual count of the bilingual phrase pair. For a detailed description see Section 5.2.2.

3.2.3 Word-based Lexicon Model

We are using relative frequencies to estimate the phrase translation probabilities. Most of the longer phrases occur only once in the training corpus. Therefore, pure relative frequencies overestimate the probability of those phrases. To overcome this problem, we use a word-based lexicon model to smooth the phrase translation probabilities.

The score of a phrase pair is computed similarly to the IBM model 1, but here, we sum

only within a phrase pair, and not over the whole target language sentence:

$$h_{\mathrm{Lex}}(f_1^J, e_1^I, s_1^K) = \log \prod_{k=1}^{K} \prod_{j=b_k}^{j_k} \sum_{i=i_{k-1}+1}^{i_k} p(f_j|e_i) \tag{3.8}$$

The word translation probabilities $p(f|e)$ are estimated as relative frequencies from the word-aligned training corpus. The word-based lexicon model is also used in both directions $p(f|e)$ and $p(e|f)$.

For a detailed analysis of the effect of different word lexica see section 5.2.

3.2.4 Word and Phrase Penalty Model

In addition, we use two simple heuristics, namely word penalty h_{WP} and phrase penalty h_{PP}:

$$h_{\mathrm{WP}}(f_1^J, e_1^I, s_1^K) = I \tag{3.9}$$

$$h_{\mathrm{PP}}(f_1^J, e_1^I, s_1^K) = K \tag{3.10}$$

These two models affect the average sentence and phrase lengths. The model scaling factors can be adjusted to prefer longer sentences and longer phrases.

3.2.5 Target Language Model

We use the SRI language modeling toolkit [Stolcke 02] to train a standard n-gram language model. The resulting feature function is:

$$h_{\mathrm{LM}}(f_1^J, e_1^I, s_1^K) = \log \prod_{i=1}^{I} p(e_i|e_{i-n+1}^{i-1}) \tag{3.11}$$

The smoothing technique we apply is the modified Kneser-Ney discounting with interpolation. We use a 6-gram language model for all tasks.

3.2.6 Reordering Model

The baseline phrase-based translation system use a very simple reordering model that is also used in, for instance, [Och & Tillmann$^+$ 99, Bender & Zens$^+$ 04]. It assigns costs based on the jump width, which is the number of source words that the beginning of a new phrase deviates from the position at following the end of the previous phrase. It is defijed as follows:

$$h_{\mathrm{RM}}(f_1^J, e_1^I, s_1^K) = \sum_{k=1}^{K} |b_k - j_{k-1} - 1| + J - j_K \tag{3.12}$$

In the literature, various additional reordering models have been proposed. We will present these alternative models in detail in Chapter 5.5.2.

4 Training

4.1 Introduction

As mentioned in [Ney 01], there are three major aspects of the statistical approach to machine translation:

- The *modeling* aspects or how the statistical dependencies of the target and source language sentences are structured.
- The *training* that defines how to estimate the free parameters of our model from the training data.
- The *search* procedure that finds the best translation of a given source sentence, using the translation model and the parameters obtained in training.

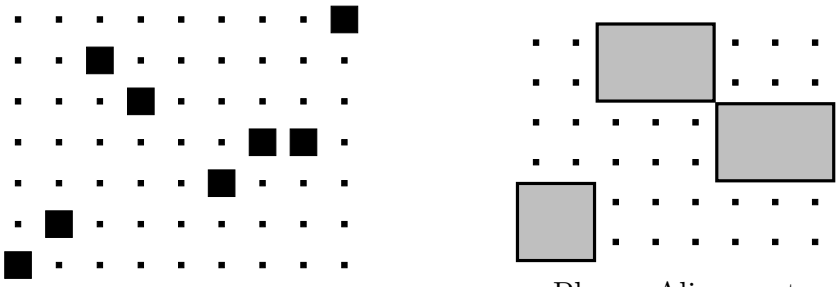
In this chapter, we describe how we perform the training of the phrase-based translation model outlined in Section 1.2.3.

A phrase-based SMT system takes a source sentence and produces a translation by segmenting the sentence into phrases and translating those phrases separately [Koehn & Och$^+$ 03]. The phrase translation table (often short phrase table), which contains the bilingual phrase pairs and the corresponding translation probabilities, is one of the main components of a statistical machine translation system. The most common method for obtaining the phrase table is by using heuristic extractions from automatically word-aligned bilingual training data [Och & Tillmann$^+$ 99]. In this method, all phrases of the sentence pair that match the constraints given by the alignment are extracted. This includes overlapping phrases. At extraction time it does not matter whether the phrases are extracted from a highly probable alignment or from an unlikely one. It is just counted, how often a phrase is seen in the training data. For a heuristically extracted pair of source phrase $\tilde{f}$ and target phrase $\tilde{e}$, the count of the phrase in the training data is $N_{\mathrm{H}}(\tilde{f}, \tilde{e})$.

As presented in Equation (3.6), phrase model probabilities are typically defined as relative frequencies of phrases extracted from word-aligned parallel training data. To obtain a conditional probability, the joint counts $N_{\mathrm{H}}(\tilde{f}, \tilde{e})$ are normalized by the marginal counts of the source and the target phrase.

$$p_{\mathrm{H}}(\tilde{f}|\tilde{e}) = \frac{N_{\mathrm{H}}(\tilde{f}, \tilde{e})}{N_{\mathrm{H}}(\tilde{e})} \qquad \text{(cf. 3.6)}$$

where $N_{\mathrm{H}}(\tilde{e})$ is the count of the target phrase $\tilde{e}$ in the training corpus. $N_{\mathrm{H}}(\tilde{e})$ is sometimes called marginal count. As described in Section 1.2.3, the translation process is implemented as a weighted log-linear combination of several models $h_m(e_1^I, s_1^K, f_1^J)$ including the logarithm of the phrase probability in $p_{\mathrm{H}}(\tilde{f}|\tilde{e})$ and in the other direction



Word Alignment Phrase Alignment

Figure 4.1: Illustration of word and phrase alignment.

$p_{\mathrm{H}}(\tilde{e}, \tilde{f})$. The phrase model is combined with a language model, word lexicon models, word and phrase penalties, and possibly others (see Section 3.2 for details). The best translation, $\hat{e}_1^{\hat{I}}$, as defined by the models, is then written as:

$$\hat{e}_1^{\hat{I}} = \underset{I, e_1^I}{\operatorname{argmax}} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, s_1^K, f_1^J) \right\} \qquad \text{(cf. 3.2)}$$

In this work, we propose to directly train our phrase models by applying a forced alignment procedure, where we use the decoder to find a phrase alignment between source and target sentences of the training data, and then updating phrase translation probabilities based on that alignment. In contrast to heuristic extraction, the proposed method provides a way of training the phrase models consistently with their use in translation. We use a modified version of a phrase-based decoder to perform the forced alignment. This way we ensure that all models used in training are identical to the ones used at decoding time. An illustration of the basic idea can be seen in Figure 4.1. In the literature, this method by itself has been shown to be problematic because it suffers from overfitting [DeNero & Gillick$^+$ 06], [Liang & Buchard-Côté$^+$ 06]. Since our initial phrases are extracted from the same training data that we want to align, very long phrases or uncommon phrases can become a part of a likely segmentation. As these phrases tend to occur in only a few training sentences, the training algorithm tends to overestimate their probability and neglects shorter phrases, which better generalize to unseen data and thus are more useful for translation. In order to counteract these effects, our training procedure applies leaving-one-out on the sentence level. Our results show that this leads to better translation quality.

We model the phrase translation model as an HMM where individual phrase translations are treated as the emission probabilities. In phrase-based translation, there is typically no direct dependency of the current phrase translation on the previous translations, so it can be described as a zero order markov model. As in the word-based hidden markov model described in Section 1.2.1.3, the hidden variable in case of the phrase model is the alignment.

We can employ Baum-Welch training that uses the EM algorithm to obtain a maximum likelihood estimate of the unknown parameters of a hidden markov model.

Ideally, we would compute expectations by enumerating all possible segmentations and alignments during training. However, this has been shown to be infeasible for real-world data [DeNero & Klein 08]. As training uses a modified version of the translation decoder, it is straightforward to apply pruning as in regular decoding. Additionally, we consider three ways for computing model expectations:

1. the single-best Viterbi alignment;
2. the n-best alignments;
3. all possible alignments remaining after pruning.

The remainder of this chapter is structured as follows. After reviewing the related work in the following section, we give a detailed description of the forced alignment in Section 4.5. In Section 4.8.1, we describe our leaving-one-out approach to counteract over-fitting. Finally, Section 4.11 explains the estimation of phrase models.

4.2 Related Work

It has been pointed out in literature that training phrase models poses some challenges. For a generative model, [DeNero & Gillick$^+$ 06] give a detailed analysis of the issues and arising problems. They introduce a model similar to the one we propose here and train it with the EM algorithm. Their results show that the trained model can not reach a performance competitive with the heuristically extracted phrase table.

Our work differs from [DeNero & Gillick$^+$ 06] by specifically addressing the problems described there. To limit the effects of over-fitting, we apply the leaving-one-out and cross-validation methods in training. In addition, we do not restrict the training to phrases consistent with the word alignment, as was done in [DeNero & Gillick$^+$ 06]. This allows us to recover from flawed word alignments.

In [Liang & Buchard-Côté$^+$ 06] a discriminative translation system is described. For training of the parameters for the discriminative features they propose a strategy they call *bold updating*. It is similar to our forced alignment training procedure described in Section 4.4.

For the hierarchical phrase-based approach, [Blunsom & Cohn$^+$ 08] present a discriminative rule model and show the difference between using only the Viterbi alignment in training and using the full sum over all possible derivations. Similarly, [Peitz & Mauser$^+$ 12] use a forced alignment procedure to estimate rule probabilities in a hierarchical phrase-based system.

Forced alignment can also be utilized to train a phrase segmentation model, as is shown in [Shen & Delaney$^+$ 08]. They report small but consistent improvements by incorporating this segmentation model, which works as an additional prior probability on the monolingual target phrase.

In [Andrés Ferrer & Juan 09] and later [Andrés Ferrer 10], phrase models are trained by a semi-hidden Markov model. They train a conditional "inverse" phrase model of the target phrase given the source phrase. Additionally to the phrases, they model the segmentation

sequence that is used to produce a phrase alignment between the source and the target sentence. Note that these models only produce monotone phrase alignments. They used a phrase length limit of 4 words with longer phrases not resulting in further improvements. To counteract over-fitting, they interpolate the phrase model with IBM1 probabilities that are computed on the phrase level. We also include these word lexica, as they are standard components of the phrase-based system.

It is shown in [Andrés Ferrer & Juan 09] that Viterbi training produces almost the same results as full Baum-Welch training. They report improvements over a phrase-based model that uses an inverse phrase model and a language model. Experiments are carried out on a custom subset of the English-Spanish Europarl corpus.

Our approach is similar to the one presented in [Andrés Ferrer & Juan 09] in that we compare Viterbi and a training method based on the forward-backward algorithm. But instead of focusing on the statistical model and relaxing the translation task by using monotone translation only, we use a full and competitive translation system as a starting point with reordering and all models included.

In [Marcu & Wong 02], a joint probability phrase model is presented. The learned phrases are restricted to the most frequent n-grams up to length 6 and all unigrams. Monolingual phrases have to occur at least 5 times to be considered in training. Smoothing is applied to the learned models so that probabilities for rare phrases are non-zero. In training, they use a greedy algorithm to produce the Viterbi phrase alignment and then apply a hill-climbing technique that modifies the Viterbi alignment by merge, move, split, and swap operations to find an alignment with a better probability in each iteration. The model shows improvements in translation quality over the single-word-based IBM Model 4 [Brown & Della Pietra$^+$ 93] on a subset of the Canadian Hansards corpus.

The joint model by [Marcu & Wong 02] is refined by [Birch & Callison-Burch$^+$ 06] who use high-confidence word alignments to constrain the search space in training. They observe that due to several constraints and pruning steps, the trained phrase table is much smaller than the heuristically extracted one, while preserving translation quality.

The work by [DeNero & Buchard-Côté$^+$ 08] describes a method to train the joint model described in [Marcu & Wong 02] with a Gibbs sampler. They show that by applying a prior distribution over the phrase translation probabilities they can prevent over-fitting. The prior is composed of IBM1 lexical probabilities and a geometric distribution over phrase lengths which penalizes long phrases. The two approaches differ in that we apply the leaving-one-out procedure to avoid over-fitting, where they explicitly define a prior distribution.

In [Moore & Quirk 07], the phrase segmentation is explicitly dropped to avoid the over-fitting issues reported by [DeNero & Gillick$^+$ 06]. The segmentation-free approach resembles the usual, heuristic phrase extraction method, but uses an iterative EM-based training. The procedure of [Moore & Quirk 07] trains the phrase model independently of other models used in the translation system.

The results and improvements given in literature are summarized in Table 4.1.

Publication	Model	BLEU[%]
[Marcu & Wong 02]	IBM4 (single-word-based)	21.6*
100k FrEn Hansards ≤ 20	Learned phrase-based	23.3*
[Birch & Callison-Burch$^+$ 06]	Heuristic phrase-based	28.4
730k EsEn Europarl (WMT06)	Learned phrase-based	26.2
[DeNero & Gillick$^+$ 06]	Heuristic phrase-based	38.5*
25k FeEn Europarl	Learned phrase-based	38.1*
	Learned + Heuristic (interpolation)	38.8*
[Andrés Ferrer & Juan 09]	Heuristic (only $p(f\|e)$)	24.1*
300k EsEn Europarl ≤ 20	Learned	26.9*
[Shen & Delaney$^+$ 08]	Heuristic phrase-based	39.6
40k ZhEn IWSLT08	Learned segmentation model	40.3

* custom test set

Table 4.1: Results and improvements of phrase model training presented in the literature

4.3 Publications and Team Work

The work presented in this chapter is the result of a collaboration with Joern Wuebker and Hermann Ney published in [Wuebker & Mauser$^+$ 10]. The general, high-level idea was presented by Hermann Ney. The detailed idea and algorithm development was done by the author of this thesis (Section 4.4).

The implementation was shared between Joern Wuebker and the author of this thesis. The author of this thesis directly implemented the techniques described in 4.5, 4.7, 4.8.2, 4.9, 4.10, 4.11.1, and 4.11.2.

Systematic experimentation for the paper presented in [Wuebker & Mauser$^+$ 10] was designed by the author of this thesis and executed by Joern Wuebker. Analysis and verification of the results was done in collaboration of all authors of the paper.

All experimentation and analysis going beyond [Wuebker & Mauser$^+$ 10] was done solely by the author of this thesis.

4.4 Phrase Model

As described in Equation (3.6), the conventional phrase model relies on a heuristic estimate of bilingual phrase pair counts:

$$p_{\mathrm{H}}(\tilde{e}|\tilde{f}) = \frac{N_{\mathrm{H}}(\tilde{f},\tilde{e})}{N_{\mathrm{H}}(\tilde{f})}.$$

In this chapter, we will replace these heuristic counts with the phrase counts $N_{\mathrm{FA}}(\tilde{f},\tilde{e})$, $N_{\mathrm{FA}}(\tilde{f})$, and the $N_{\mathrm{FA}}(\tilde{e})$ counts that we obtained in our consistent training procedure and obtain an new phrase translation probability

$$p_{\mathrm{FA}}(\tilde{e}|\tilde{f}) = \frac{N_{\mathrm{FA}}(\tilde{f},\tilde{e})}{\sum_{\tilde{e}'} N_{\mathrm{FA}}(\tilde{f},\tilde{e}')}. \tag{4.1}$$

The formal definition of counts using all log-linear models:

$$N_{\mathrm{FA}}(\tilde{f},\tilde{e}) := \sum_{(f_1^J,e_1^I)} \sum_{K,s_1^K} p(s_1^K|e_1^I,f_1^J) \sum_{k=1}^{K} \Big[\delta(\tilde{f}_k,\tilde{f}) \cdot \delta(\tilde{e}_k,\tilde{e})\Big]$$

In the following section, we will describe the training of this model, i. e. how to efficiently obtain the counts $N_{\mathrm{FA}}(\cdot)$.

4.5 Forced Alignment

The training process is divided into three parts. First we obtain all models needed for a normal translation system. From this alignment, we then estimate new phrase models, while keeping all other models unchanged. In this section, we describe our forced alignment procedure that is the basic training procedure for the models proposed here.

The idea of forced alignment is to perform a phrase segmentation and alignment of each sentence pair of the training data using the full translation system as in decoding. What we call segmentation and alignment here corresponds to the "concepts" used by [Marcu & Wong 02]. In principle, we apply our normal phrase-based decoder on the source side of the training data and constrain the translations to the corresponding target sentences from the training data. Given this task, there are a few changes to the decoding algorithm and phrase matching process to improve efficiency.

Formally, the task of forced alignment is to search for the best phrase segmentation and alignment given a source sentence f_1^J and target sentence e_1^I that fully covers both sentences. As introduced in Equation (3.3), a segmentation of a sentence into K phrases is defined by:

$$k \rightarrow s_k := (i_k, b_k, j_k), \text{ for } k = 1,\ldots,K \qquad \text{(cf. 3.3)}$$

where for each segment i_k is the last position of the kth target phrase. b_k, j_k are the start and end positions of the source phrase aligned to the kth target phrase. Consequently, we can modify Equation (3.2) to define the best segmentation of a sentence pair as:

$$\hat{s}_1^{\hat{K}} = \underset{K,s_1^K}{\operatorname{argmax}} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, s_1^K, f_1^J) \right\} \tag{4.2}$$

We use the identical models as in translation: conditional phrase probabilities $p(\tilde{f}_k|\tilde{e}_k)$ and $p(\tilde{e}_k|\tilde{f}_k)$, within-phrase lexical probabilities, reordering model as well as word and phrase penalty. A language model is not used in this case, as the system is constrained to the given target sentence and thus the language model score would have no effect on the alignment. The same applies to the word penalty: as we have to cover all target words in the alignment, the penalty for all possible alignment hypotheses is identical. The omission of the language model in the alignment results in a major reduction of search effort.

To find the best alignment, $\hat{s}_1^{\hat{K}}$, we have to choose the number of phrases, K, used in the alignment, the segmentation of the source sentence into phrases, and the order in which

these phrases are chosen, and the length of the target phrase that we align with each source phrase.

The alignment process can be interpreted as a sequence of K decisions where we select

1. a source phrase described by the start and end position b_k and j_k from the unaligned words of the source sentence, and
2. the last target position i_k to be aligned to this source phrase.

At the end of the sequence, we have all source sentence words and all target sentence words aligned. In one sequence, each word can only be aligned using one phrase. Overlapping phrases are not permitted. On the target side, we keep track of this constraint by monotonously processing the target sentence from left to right. On the source side, we allow for reordering and therefore have to keep track of the aligned source words at each step. This is done using a coverage set $C \subseteq \{1, \ldots, J\}$ that contains all source sentence indices that have already been covered in the previous decisions within the sequence. We start with the empty set $C_0 = \emptyset$ and reach full source sentence coverage in the final step $C_K = \{1, \ldots, J\}$. Figure 4.2 illustrates the sequence of decisions.

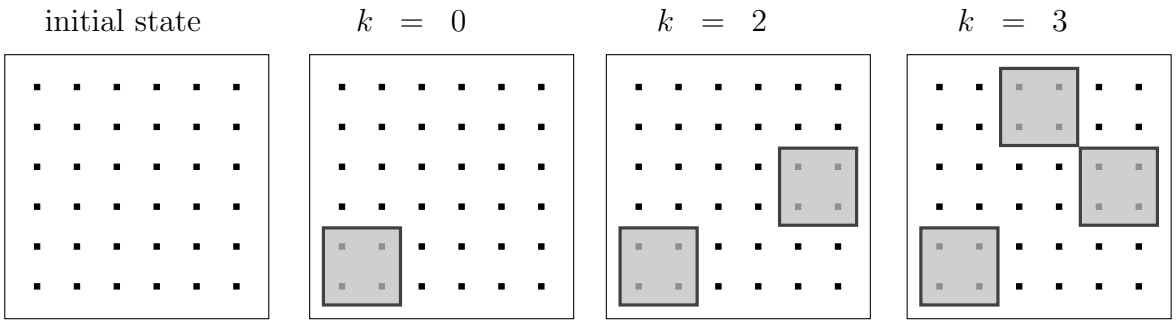
At each step, we have to ensure that the words covered by the currently aligned source phrase do not overlap with the source phrases aligned in previous steps i.e. $\{b_k, \ldots, j_k\} \cap C_{k-1} = \emptyset$. The power set of the full coverage set can also be interpreted as the states of a graph where the edges are labeled with the alignment decisions. The initial state is C_0 where no word is covered and the final state is C_K where all words are covered. We will keep this notion of the alignment search graph in the desctiptions of the algorithms that we propose in this chapter. In many cases, the state in the coverage graph will be extended with information that goes beyond the coverage. What information will be stored in the extended state depends on the models that are used in the search and alignment algorithms.

Each step in the alignment sequence or each arc in the alignment search graph has some associated model score. For the description of the alignment algorithm, we will only use the most relevant models from those described in Section 3.2, to simplify the description. In general, we have to distinguish local costs that depend only on the phrase pair that is currently aligned and non-local costs that might depend on some information from the previous phrase pair. In our case, the local costs are the translation model (TM) components of the alignment score shown in Equation (4.3):

$$\begin{aligned} q_{TM}(i_{k-1}+1, i_k, b_k, j_k) = & \lambda_{\text{Phr}} \cdot \log p_{\text{Phr}}(\tilde{f}|\tilde{e}) + \lambda_{\text{iPhr}} \cdot \log p_{\text{iPhr}}(\tilde{e}|\tilde{f}) \\ & \lambda_{\text{Lex}} \cdot \log p_{\text{Lex}}(\tilde{f}|\tilde{e}) + \lambda_{\text{iLex}} \cdot \log p_{\text{iLex}}(\tilde{e}|\tilde{f}) \\ & + \lambda_{\text{pp}} \end{aligned} \tag{4.3}$$

The non-local costs are given by the reordering model (RM). In regular decoding, we would also use a language model as a non-local cost component. As mentioned above, this is not needed here because there is no variance in the target words. In this description, we use only the distance-based jump costs from Equation (3.12):

$$q_{RM}(j_{k-1}, b_k) = \lambda_{\text{dist}} \cdot |b_k - j_{k-1} - 1|. \tag{4.4}$$



$C_0 = 000000$	$C_1 = 110000$	$C_2 = 110011$	$C_3 = 111111$
$i_0 = 0$	$i_1 = 1$	$i_2 = 3$	$i_3 = 5$
$b_0 = 0$	$b_1 = 0$	$b_2 = 4$	$b_3 = 2$
$j_0 = 0$	$j_1 = 1$	$j_2 = 5$	$j_3 = 3$

Figure 4.2: Illustration of word and phrase alignment. Source positions $i = 0, \ldots, 4$ from left to right, target positions $j = 0, \ldots, 4$ from bottom to top. The phrase alignment is produced phrase-by-phrase $k = 0, 1, 2$.

Two alignment hypotheses can have identical coverage vectors but different last source word positions. In order to correctly compute the distortion costs for each hypothesis, we have to include the last source position in the state information. We also have to include the last target position in the state to efficiently compute reordering penalties. The states in the alignment search space can be identified by a triple (C, i, j), where C is the coverage set, i is the last target word position covered in the current alignment step and j is the last source word position covered in the current alignment step.

In an alignment sequence step starting from state (C_k, i_{k-1}, j_{k-1}) and deciding to align source words $b_k, \ldots, j_k$ with the target words $i_{k-1}, \ldots, i_k$, the successor state is $(C_k \cup \{b_k, \ldots, j_k\}, i_k, j_k)$. The score is updated by

$$q_{TM}(i_{k-1} + 1, i_k, b_k, j_k) + q_{RM}(j_{k-1}, b_k). \tag{4.5}$$

Finding the best overall alignment amounts to finding the best path in the search graph described above. To construct an efficient dynamic programming algorithm, we first have to construct a recursive formulation of our problem that uses the distribution of our search problems into small steps as described above. In addition to the cost functions described in Equation (4.3) and (4.4), we need to define an auxiliary quantity $Q(C, i, j)$, where C is a coverage vector, i is the first uncovered target position, and j is the last source position covered. The initial condition is that the coverage vector is empty, i.e. $C = \emptyset$, and the last positions are set to 0. As previously, we start counting word indices starting from 1. For each state defined by the triple (C, i, j), we find the best predecessor by maximizing over all possible predecessor source phrase final positions j'', all possible source phrase starting positions j', and all target phrase starting positions i'. The maximization over j' and i' is constrained by the maximum phrase length in the source and the target, respectively.

Algorithm 1 Alignment search for all phrase alignments

Input: source sentence f_1^J, target sentence e_1^I, alignment options $E(j, j', i, i')$ for $1 \leq j < j' \leq J$ and $1 \leq i < i' \leq I$, translation model $q_{TM}(i, i', j, j')$, distortion model $q_{DM}(j, j')$

Output: cost and back-pointer data structure $Q(C, i, j)$ where element is a priority queue of back-pointers sorted with best scores first

```
1: for source cardinality c = 1 to J do
2:   for source phrase length l = 1 to min{L_s, c} do
3:     for all source coverage C' ⊂ {1,...,J} : |C'| = c − k do
4:       for all source start positions j ∈ {1,...,J} : C' ∩ {j,...,j+l} = ∅ do
5:         source coverage C = C' ∪ {j,....j+l}
6:         for all states i, j' ∈ Q(C', ·, ·) do
7:           for all target end positions i' ∈ E(j, j+l, i, ·) do
8:             score = Q(C', i, j').head() + q_TM(i, i', j, j+l) + q_DM(j', j)
9:             if score > Q(C, i', j+l) then
10:              Q(C, i', j+l).enqueue(score, (C', i, j'))
```

Another constraint is that the coverage vector C actually contains the source positions that would be covered by the source phrase $\{j', \ldots, j\}$.

The best result $\hat{Q}$ is then the maximum over all possible triples with full coverage in the source and the target i.e. maximizing over all possible end positions j in $(\{1, \ldots, J\}, I, j)$. In order to be fair in the comparison of reordering costs for all alignment hypotheses, we need to include the costs for finalizing the sentence. Formally, the dynamic programming recursion equations are:

$$
\begin{aligned}
Q(\emptyset, 0, 0) &= 0 \\
Q(C, i, j) &= \max_{\substack{j'', j' : j' \leq j < j' + L_s \wedge \{j', \ldots, j\} \subseteq C \\ i' : i' < i \leq i' + L_t}} \left\{ Q(C \setminus \{j', \ldots, j\}, i', j'') + q_{TM}(i', i, j', j) + q_{RM}(j'', j') \right\} \\
\hat{Q} &= \max_j \left\{ Q(\{1, \ldots, J\}, I, j) + q_{RM}(j, J+1) \right\}
\end{aligned}
$$

Starting from the dynamic programming Equation (4.6) above, we can derive an algorithmic description that computes the maximizing argument we are interested in: the best alignment. The search algorithm for finding the best alignment is given in Algorithm 1.

As in normal search for phrase-based machine translation, the alignment is organized by increasing source cardinality c, the number of source words already aligned. While the target sentences is covered sequentially from left to right, the source positions can have gaps in partial hypotheses.

In each step of the outermost loop, we try to construct a new partial hypothesis that covers exactly c by choosing a phrase of length l and combining it with partial hypotheses that has $c - l$ words covered, where that phrase matches uncovered source and target positions. In order to properly keep track of the scoring and coverage, we not only have to account for covered positions in each partial hypothesis but also have to remember the

last position that was covered in the source sentence in order to properly compute the distortion model.

The score of the new hypothesis is the sum of the partial hypothesis costs so far, the translation costs of the new phrase and the cost for the distortion model that scores the deviation from the monotonous source word sequence. This process is repeated, until all source words are covered.

In practice, there are situations, where the algorithm can not find a full alignment of the source and the target sentence with the given set of phrases, leaving source words, target words, or both, uncovered by any phrase. Sentences for which the decoder can not find an alignment are discarded for the phrase model training. In our experiments, this is the case for roughly 1-5% of the training sentences.

Reasons for not finding an alignment tend to be caused by the bad original word alignments or non-literal translations. In the case of bad word alignments, the phrase extraction constraints prevent necessary phrases from being extracted. This also happens in cases when the translations do not exactly match the source, for example they contain more information or clarifications. During phrase alignment, it can also be that the reordering constraints used during search cannot reproduce the original alignment.

4.6 Pruning

As [DeNero & Klein 08] have shown, finding the best phrase alignment is NP-complete. We cannot expect to find the optimal solution efficiently in all cases. Our Goal is to develop an algorithm that can find a good solution in a limited amount of time using dynamic programming [Bellman 57] and beam-search [Jelinek 98]. To manage the runtime and memory requirements, especially for longer sentences, we employ a beam-search procedure [Jelinek 98]. Pruning is done in several stages of the search process by restricting the active hypotheses to a fixed maximum number of best candidates (histogram pruning, [Steinbiss & Tran$^+$ 94]). As described in [Zens 08], we use three types of pruning:

1. **Observation pruning.** The number of target translation options of each source phrase is limited to the best N_O candidates before the start of the alignment procedure.
2. **Lexical pruning per coverage.** For all alignment hypotheses that share the same source coverage C, we keep the best N_L. As we apply reordering only on the source sentence and process the target sentence monotonously, the maximum number of alignment hypotheses with the same source coverage is the number of target words. Therefore N_L is only effective, if the target sentence length is longer than N_L. Note that this is only true for the alignment case, not for translation.
3. **Coverage pruning per cardinality.** For each cardinality c in the search for the best alignment, we keep only the N_C best source coverage hypotheses. The evaluation is done based on the best scoring alignment hypothesis that exists for this coverage vector. If a source coverage is pruned, all alignment hypothesis with this source coverage are removed.

Note that we do not perform any threshold pruning as described in [Zens 08] or [Koehn & Hoang$^+$ 07], because histogram pruning allows for a hard maximum restriction of the search effort. In combintation with other restriction, this type of pruning helps to keep the time and memory needed to do alignment very far from the theoretically exponential bounds. This is even more important since we need to align the entire training data. Threshold pruning is less effective here, as the beam size depends on model weights and therefore the number of hypotheses explored changes with every change of parameters or weights. A discussion of the effects of threshold pruning in translation can be found in [Cettolo & Federico 04].

Algorithm 2 extends Algorithm 1 with pruning and the use of rest costs. During pruning, hypotheses with identical coverage C are compared and only the best ones are kept. There are two checks in lines 8 and 12, where partial hypotheses are discarded. The decision is based on the partial costs that have been accumulated so far, the cost of the current alignment and a rest cost estimate $R(C, j+l)$ that is a heuristic for the remaining costs of aligning the remaining unaligned parts of the sentence pair. As we want to keep the alignment very similar to the decoding process used for translation, we use the same heuristics as [Och 03].

The comparison of the partial alignment hypotheses based on coverage is done to ensure that only truly comparable hypotheses, hypotheses that align the same source words, are competing with each other. Comparing hypotheses based on cardinality only will prefer hypotheses that align "easy" words first, leading to sub-optimal alignments.

Algorithm 2 Alignment search for all phrase alignments with pruning and rest-cost estimation.

Input: source sentence f_1^J, target sentence e_1^I, alignment options $E(j, j', i, i')$ for $1 \leq j < j' \leq J$ and $1 \leq i < i' \leq I$, translation model $q_{TM}(i, i', j, j')$, distortion model $q_{DM}(j, j')$

Output: cost and back-pointer data structure $Q(C, i, j)$ where element is a priority queue of back-pointers sorted with best scores first

1: **for** source cardinality $c = 1$ **to** J **do**
2: **for** source phrase length $l = 1$ **to** $\min\{L_s, c\}$ **do**
3: **for all** source coverage $C' \subset \{1, \ldots, J\} : |C'| = c - k$ **do**
4: **for all** source start positions $j \in \{1, \ldots, J\} : C' \cap \{j, \ldots, j+l\} = \emptyset$ **do**
5: source coverage $C = C' \cup \{j, \ldots . j+l\}$
6: **for all** states $i, j' \in Q(C', \cdot, \cdot)$ **do**
7: partial score $q = Q(C', i, j') + q_{DM}(j', j)$
8: **if** $q + R(C, j+l) + q_{TM}(j, j+l)$ isTooBadForCoverage C **then**
9: CONTINUE
10: **for all** target end positions $i' \in E(j, j+l, i, \cdot)$ **do**
11: partial score $q' = q + R(C, j+l) + q_{TM}(i, i', j, j+l)$
12: **if** q' isTooBadForCoverage C **then**
13: BREAK
14: score $= Q(C', i, j').head() + q_{TM}(i, i', j, j+l) + q_{DM}(j', j)$
15: **if** score $> Q(C, i', j+l)$ **then**
16: $Q(C, i', j+l).enqueue(score, (C', i, j'))$

Table 4.2: Notation used in the algorithmic description of the alignment (Algorithm 2).

$Q(C, i, j)$	score of alignment hypothesis with source coverage set C, last covered target position i and last covered source position j
$B(C, i, j)$	back pointer of hypothesis (C, i, j)
$A(C, i, j)$	maximizing argument of hypothesis (C, i, j)
L_S	maximum source phrase length
$q_{TM}(i, i', j, j')$	local alignment score for aligning $f_j, \ldots, f_{j'}$ with $e_i, \ldots, e_{i'}$ (Equation (4.3))
$q_{RM}(j, j')$	reordering score for starting a phrase at position j' after having last aligned source position j (Equation (4.4))
x isTooBadForCoverage C	check if the score of x would be pruned in coverage C
pruneCardinality c	apply pruning for all hypothesis of coverage c

Algorithm 3 Reconstruct best phrase alignment

Input: alignment cost and back-pointer lists $Q(C, i, j)$, source sentence length J, target sentence length I

Output: best alignment ß

1: $k = 0$, $i = I$, $j = J$, $C = \{1, \ldots, J\}$
2: **while** $i > 1$ **and** $j > 1$ **do**
3: $\quad score, i', j' = Q(C, i, j).head()$
4: $\quad s_k = (i, j', j)$
5: $\quad C = C \setminus \{j', \ldots, j\}$, $k = k + 1$, $i = i'$, $j = j'$
6: K=k-1
7: reverse(s_1^K)

4.7 Word Graph and n-best Generation

For some of the methods developed in this thesis, there is the need to construct an explicit graph-based representation of the alignment hypothesis space. One reason is the computation of phrase expectations using the forward-backward algorithm [Rabiner 90]. Another reason is the extraction of n-best lists for the best alignments, either for using these alignments in an extended Viterbi-training, for use in external applications, or for manual inspection.

The generation of word-graphs for phrase-based machine translation in general is described in [Koehn 03] and [Zens & Ney 05]. The difference in the case of alignment graphs is that all paths not only cover the same source sentence, but also generate the same target sentence. The only difference in the paths in the graphs lies in the segmentation into phrases and the alignment of the phrases. In our case, the graph's edges consist of phrases and are annotated not only with the target phrase but also the model scores, the source phrase, and reordering information.

To produce the n-best list, we use the A* algorithm presented in [Ueffing & Och$^+$ 02]. Note that for the alignment graphs all n-best hypothesis will lead to the same translation.

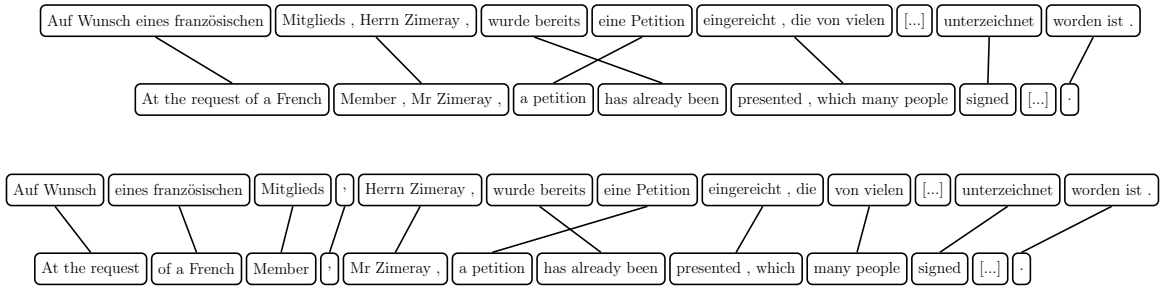


Figure 4.3: Segmentation example from forced alignment. Top: without leaving-one-out. Bottom: with leaving-one-out.

Duplicate filtering, that is sometimes used to produce unique n-best lists in translation is not applicable here.

4.8 Preventing Over-Fitting

One of the most severe problems in the phrase alignment procedure reported in the literature is related to over-fitting on the training data [DeNero & Gillick$^+$ 06, Liang & Buchard-Côté$^+$ 06].

When given a bilingual sentence pair, we can usually assume there are a number of equally correct phrase segmentations and corresponding alignments. For example, it may be possible to transform one valid segmentation into another by splitting some of its phrases into sub-phrases or by shifting phrase boundaries. This is different from word-based translation models, where a typical assumption is that each target word corresponds to only one source word. As a result of this ambiguity, different segmentations are recruited for different examples during training. That in turn leads to over-fitting which shows in overly determinized estimates of the phrase translation probabilities. In addition, [DeNero & Gillick$^+$ 06] found that the trained phrase table shows a highly peaked distribution in opposition to the more flat distribution resulting from heuristic extraction, leaving the decoder only a few translation options at decoding time.

4.8.1 Leaving-One-Out

Leaving-one-out, as we apply it here, can improve the phrase alignment in situations, where the probability of rare phrases and alignments might be overestimated. The training data that consists of N parallel sentence pairs, f_n and e_n, for $n = 1, \ldots, N$, is used for both the initialization of the translation model $p(\tilde{f}|\tilde{e})$ and the phrase model training. While in this way we can make full use of the available data and avoid unknown words during training, it has the drawback that it can lead to over-fitting. All phrases extracted from a specific sentence pair f_n, e_n can be used for the alignment of this sentence pair. This includes longer phrases, which only match in very few cases in the data. Therefore, those long phrases are trained to fit only a few sentence pairs, strongly overestimating their

translation probabilities and failing to generalize. In the extreme case, whole sentences will be learned as phrasal translations. The average length of the used phrases is an indicator of this kind of over-fitting, as the number of matching training sentences decreases with increasing phrase length. We can see an example in Figure 4.3. Without leaving-one-out the sentence is segmented into a few long phrases, which are unlikely to occur in the data to be translated. Phrase boundaries seem to be unintuitive and based on particularities of the sentence pair at hand. With leaving-one-out the phrases are shorter and therefore better suited for generalizing to unseen data. Also, the segmentation is closer to what a human would deem reasonable.

Previous attempts have dealt with the over-fitting problem by limiting the maximum phrase length [DeNero & Gillick$^+$ 06, Marcu & Wong 02] and by smoothing the phrase probabilities by lexical models on the phrase level [Andrés Ferrer & Juan 09]. However, [DeNero & Gillick$^+$ 06] experienced similar over-fitting with short phrases due to the fact that the same word sequence can be segmented in different ways, leading to specific segmentations being learned from specific training sentence pairs. Our results confirm these findings. To deal with this problem, instead of simple phrase length restrictions, we propose to apply the leaving-one-out method, which is also used for language modeling techniques [Kneser & Ney 95].

We have to distinguish two cases for our training procedure. For the first iteration, where we initialize our model probabilities with the heuristic phrase counts, we can easily remove the counts for a single sentence. For further iterations, this cannot be done in the same way. We therefore rely on cross-validation for higher iterations which is described in Section 4.8.2.

When using leaving-one-out, we modify the phrase translation probabilities for each sentence pair. For a training example f_n, e_n, we have to remove all phrases $N_{H,n}(\tilde{f}, \tilde{e})$ that were extracted from this sentence pair from the phrase counts that we used to construct our phrase translation table. The same holds for the marginal counts $N_{H,n}(\tilde{e})$ and $N_{H,n}(\tilde{f})$. Starting from Equation 3.6, the leaving-one-out phrase probability for training sentence pair n is

$$p_{l1o,n}(\tilde{f}|\tilde{e}) = \frac{N_{\mathrm{H}}(\tilde{f}, \tilde{e}) - N_{\mathrm{H},n}(\tilde{f}, \tilde{e})}{N_{\mathrm{H}}(\tilde{e}) - N_{\mathrm{H},n}(\tilde{e})} \tag{4.6}$$

To be able to perform the re-computation in an efficient way, we store the source and target phrase marginal counts $N_{\mathrm{H}}(\tilde{f})$ and $N_{\mathrm{H}}(\tilde{e})$ for each phrase pair in the phrase table. A phrase extraction is performed for each training sentence pair separately using the same word alignment as for the initialization. It is then straightforward to compute the phrase counts after leaving-one-out using the phrase probabilities and marginal counts stored in the phrase table.

While this works well for more frequent observations, singleton phrases are assigned a probability of zero. We refer to singleton phrases as phrase pairs that occur only in one sentence. For these sentences, the decoder needs the singleton phrase pairs to produce an alignment. Therefore we retain those phrases by assigning them a probability close to zero. We evaluated with two different strategies for this, which we call standard and length-based leaving-one-out. Standard leaving-one-out assigns a fixed probability α to singleton phrase pairs. This way the decoder will prefer using more frequent phrases for

Table 4.3: Avg. source phrase lengths in forced alignment without leaving-one-out and with standard and length-based leaving-one-out.

	avg. phrase length
without l1o	2.5
standard l1o	1.9
length-based l1o	1.6

the alignment, but is able to resort to singletons if necessary. However, we found that with this method longer singleton phrases are preferred over shorter ones, because fewer of them are needed to produce the target sentence. In order to better generalize to unseen data, we would like to give the preference to shorter phrases. This is done by length-based leaving-one-out, where singleton phrases are assigned the probability $\beta^{(|\tilde{f}|+|\tilde{e}|)}$ with the source and target phrase lengths $|\tilde{f}|$ and $|\tilde{e}|$ and fixed $\beta < 1$. In our experiments we set $\alpha = e^{-20}$ and $\beta = e^{-5}$. Table 4.3 shows the decrease in average source phrase length by application of leaving-one-out.

4.8.2 Cross-Validation

For the first iteration of the phrase training, leaving-one-out can be implemented efficiently as described in Section 4.8.1. The phrases that were extracted from the current sentence are removed from the phrase table. For most phrases, this means that the count and the marginal are reduced by the exact count of the phrase pair in the current sentence. We keep the original counts and marginal counts for each phrase so we can efficiently compute discounted leaving-one-out phrase probabilities at decoding time by using the same phrase extraction algorithm that was used to create the initial phrase table. In subsequent iterations, this becomes more complicated. When using n-best lists or graphs, phrase counts for a particular sentence cannot easily be replicated as they would depend on the phrase table and the 'actual alignment from the previous iteration. An exact computation of the discounted leaving-one-out phrase probabilities would require to either (1) redo the computation from the previous iteration; or (2) to store the fractional phrase counts from the previous iteration. This storage would have to be individually for each sentence so that sentence-level discounting can be applied.

As an approximate solution to storing phrase counts for each training sentence pair, we propose a cross-validation strategy that trains on and stores the counts of larger batches of data. Instead of re-computing phrase probabilities on the sentence-level, they are now re-computed on the batch level. All phrase counts collected in the previous iteration from the currently aligned batch are subtracted from the phrase counts used in the alignment. The counts for each individual batch in the current iteration of the alignment are then stored on disk again for the next iteration. While this still creates some overhead in keeping the batch counts stored on disk even after the iteration has finished and the new phrase tables are computed, this is significantly less than if done for individual sentences.

Conceptually, leaving-one-out is a special case of the cross-validation where the batch size

is exactly one sentence. In our experiments, we set the batch-size to 1000-10000 sentences, depending on the task size. The larger the batches become, the more important the order in which the training sentences are processed is. If the training data consists of documents from several smaller sources or if all documents of a topic end up in a single batch, all the topic-specific phrases might be removed from the phrase table. This is not the intention of the cross-validation procedure. To avoid complications like that we need to make sure that batch sizes do not get too big and that the training documents are broken up and distributed randomly across the entire training data.

4.9 Phrase Matching

Before entering the alignment algorithm, and to be able to compute leaving-one-out phrase probabilities, we first have to determine, which phrases can be used in the upcoming alignment.

In phrase matching, we take all the phrases from the phrase table that can be applied to align the current sentence pair and determine the positions, in which they can be used. The phrase pairs are annotated with their source and target positions. Phrase matching is done before the alignment starts. During the alignment, we then only have to check the positional information. This greatly simplifies the alignment process as only a few integer number comparisons have to be done to extend alignment hypotheses.

Phrase matching is separated into two phases:

- **Source phrase matching** determines the source phrases matching parts of the source sentence. The algorithm is identical to the one used for translation. For each matching source phrase we retrieve the translation candidates.
- **Target phrase matching** discards all translation candidates that cannot be applied in the sentence and annotates all remaining translation candidates with target sentence position information.

Note that if phrases appear multiple times in the source or the target sentence, they are replicated with differing position annotations. Efficient target phrase matching can be done by constructing a hash map or otherwise constant access time data structure for each sentence. The structure maps target words to their positions in the target sentence. If first word of a candidate target phrase is not in the map, the process can be aborted. Otherwise, we go to each position of that target word in the sentence and check if the rest of the phrase matches. If a word in the phrase does not match the word in the sentence, the process can be aborted. Only if a candidate target phrase passes all the tests, it is then used for alignment.

In practice, target phrase matching is only a very small portion of the total runtime and therefore we can use a linear search.

In rare cases it might happen that there are no phrase translations available for some of the words in the source or the target sentence. This can occur depending on how the phrase table was generated. In the case of missing phrases, new phrases of the currently uncovered words are formed with all potential target or source words. This would allow

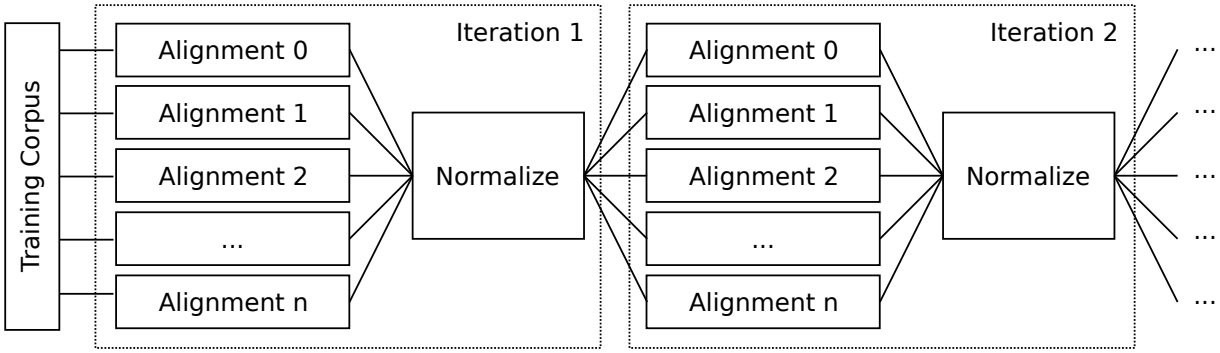


Figure 4.4: Parallelization of alignment training.

these words to be aligned to any word in the paired sentence. Since we do not have phrase translation probability estimates for these phrases, a fixed penalty is used. These additional phrases are needed for good reference reachability.

4.10 Parallelization

To cope with the runtime and memory requirements of phrase model training that was pointed out by previous work [Marcu & Wong 02, Birch & Callison-Burch$^+$ 06], we parallelize the forced alignment by splitting the training corpus into parts of 1k to 10k sentence pairs depending on the task size. Each of these blocks is processed separately on the next available machine. During the alignment process, the phrase counts are collected and cached internally. At the end of the block, these partial phrase count files are sorted and written to disk. To balance the load between the different blocks, the training corpus is randomized on the sentence level. This evens out run-time and memory requirements of individual jobs by distributing the sentence lengths evenly. To conserve memory, each of these blocks only loads the phrases that are required for alignment.

When all blocks have completed, the results are merged and normalized for the next iteration. Process scheduling is handled by the Oracle Grid Engine[1]. The process is illustrated in Figure 4.4.

4.11 Phrase Model Training

The produced phrase alignment can be given as a single best alignment, as the n-best alignments, or as an alignment graph representing all alignments considered by the decoder. We have developed two different methods for phrase translation probabilities which make use of the force-aligned training data. Additionally, we consider smoothing

[1] `http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html`

by interpolation of the generative model with the state-of-the-art heuristics proposed by [DeNero & Gillick$^+$ 06].

4.11.1 Viterbi

The first of our generative phrase training variants estimates phrase translation probabilities by their relative frequencies in the Viterbi alignment of the data. This procedure is similar to the heuristic model but with counts from the phrase-aligned data produced in training rather than computed on the basis of a word alignment. The translation probability of a phrase pair $(\tilde{f}, \tilde{e})$ is estimated as:

$$p_{\mathrm{FA}}(\tilde{f}|\tilde{e}) = \frac{N_{\mathrm{FA}}(\tilde{f}, \tilde{e})}{\sum_{\tilde{f}'} N_{\mathrm{FA}}(\tilde{f}', \tilde{e})} \tag{4.7}$$

where $N_{\mathrm{FA}}(\tilde{f}, \tilde{e})$ is the count of the phrase pair $(\tilde{f}, \tilde{e})$ in the phrase-aligned training data. This can be applied to either the Viterbi phrase alignment or an n-best list. For the simplest model, each hypothesis in the n-best list is weighted equally. We will refer to this model as the *count* model as we simply count the number of occurrences of a phrase pair. We also experimented with weighting the counts with the estimated phrase posterior of the corresponding entry in the n-best list. The sum of the likelihoods of all entries in an n-best list is normalized to 1. We will refer to this model as the *weighted count* model.

In the general case, we define the forced alignment counts as

$$N_{\mathrm{FA}}(\tilde{f}, \tilde{e}) := \sum_{(f_1^J, e_1^I)} \sum_{s_1^K} \sum_{k=1}^{K} \left[\frac{\sum_{m=1}^{M} \lambda_m h_m(e_1^I, ß, f_1^J)}{\sum_{s'^K_1} \sum_{m'=1}^{M} \lambda_{m'} h_{m'}(e_1^I, s'^{K'}_1, f_1^J)} \cdot \delta(\tilde{f}_k, \tilde{f}) \cdot \delta(\tilde{e}_k, \tilde{e}) \right]. \tag{4.8}$$

4.11.2 Phrase Expectations

Ideally, the training procedure would consider all possible alignment and segmentation hypotheses. Alternatives are weighted by their posterior probability. As discussed earlier, the run-time requirements for computing all possible alignments is prohibitive for anything but toy tasks. However, we can approximate the space of all possible hypotheses by the search space that was used for the alignment. While this might not cover all phrase translation probabilities, it allows the search space and translation times to be feasible and still contains the most probable alignments. This search space can be represented as a graph of partial hypotheses [Ueffing & Och$^+$ 02] on which we can compute expectations using the forward-backward algorithm.

The forward-backward algorithm is an efficient way to compute normalized posterior probabilities on the hidden markov model. Dynamic programming is used to efficiently sum over the hidden variables to obtain marginal distributions for normalization.

In our case, the hidden variable is the phrase alignment and we want to obtain the normalized probability for each bilingual phrase used in all alignments. As mentioned in Section 4.7 the key aspect that is different for phrase alignment is the fact that the alignment not only contains the correspondence of the source and the target words but also simultaneously contains a segmentation of the source and the target sentence into phrases.

The tutorial in [Rabiner 90] gives a general description of the algorithm by using a graph, where the vertices are *states* and the edges are *transitions*. In our case, the states follow directly from the different values in the tuples (C, i, j) in Algorithm 2. Each tuple defines a different state in the graph. The transitions correspond to phrases in an alignment. That correspondence allows the original forward-backward algorithm to be applied directly.

We will refer to this alignment as the *full* alignment. In contrast to the method described in Section 4.11.1, phrases are weighted by their posterior probability in the word graph. As suggested in work on minimum Bayes-risk decoding for SMT [Ehling & Zens$^+$ 07], [Tromble & Kumar$^+$ 08], we tried using a global factor to scale the posterior probabilities. However, we did not observe this factor to have a strong impact on the translation quality achieved by the learned models. This might be partially due to the fact that we always normalize the model scaling factors to sum up to 1.

4.11.3 Phrase Table Interpolation

[DeNero & Gillick$^+$ 06] reported improvements in translation quality by interpolation of phrase tables produced by the generative and the heuristic model. We adopt this method and also report results using log-linear interpolation of the estimated model with the heuristic model.

The log-linear interpolations $p_{int}(\tilde{f}|\tilde{e})$ of the phrase translation probabilities are estimated as

$$p_{int}(\tilde{f}|\tilde{e}) = \Big(p_{\mathrm{H}}(\tilde{f}|\tilde{e})\Big)^{1-\psi} \cdot \Big(p_{\mathrm{FA}}(\tilde{f}|\tilde{e})\Big)^{\psi} \tag{4.9}$$

where ψ is the interpolation weight, p_{H} the heuristically estimated phrase model and p_{FA} the count model. The interpolation weight ψ is adjusted on the development corpus. When interpolating phrase tables containing different sets of phrase pairs, we retain the intersection of the two.

4.12 Results

In this section, we will experimentally explore the phrase model training presented in Chapter 3.2.6. We will explore the effect of different models settings on translation quality

and phrase table size. The experiments will be mostly done on the WMT 2008 German-English test set, with some additional experiments to show the effect of the method on other tasks.

4.12.1 Experimental Setup

We are given the three data sets WMT 2008 German-English, WMT 2008 Dev and WMT 2008 Test. For the heuristic phrase model, we first use GIZA++ [Och & Ney 03] to compute the word alignment on WMT 2008 German-English. Next we obtain a phrase table by extracting phrases from the word alignment. For detailed corpus statistics, refer to Section 9.2.1. For the WMT 2008 German-English experiments, we perform minimum error rate training with the downhill simplex algorithm [Nelder & Mead 65] on the development data to obtain a set of scaling factors that achieve a good BLEU score. We then use these models and scaling factors to do a forced alignment, where we compute a phrase alignment for the training data. For the experiments on other languages, we used uniform scaling factors for all models. We found no major or consistent difference between the optimized and uniform scaling factors.

The phrase table obtained by the heuristic extraction is also used to initialize the training. The forced alignment is run on the training data from which we obtain the phrase alignments. Those are used to build a phrase table according to the proposed generative phrase models. Afterward, the scaling factors are trained on WMT 2008 Dev for the trained phrase table. By feeding back the new phrase table into the forced alignment we can reiterate the training procedure. When training is finished, the resulting phrase model is evaluated on the WMT 2008 Dev and the WMT 2008 Test. Additionally, we interpolate the new phrase table with the heuristically estimated phrase table, retrain the scaling factors and evaluate afterwards.

The baseline system is a standard phrase-based SMT system with eight features: phrase translation and word lexicon probabilities in both translation directions, phrase penalty, word penalty, language model score, and a simple distance-based reordering model. To evaluate the performance of the learned phrase table, we replace the two phrase translation probabilities and keep the other features and models identical to the baseline. For the feature-wise combination, the two generative phrase probabilities are added to the features, resulting in a total of 10 features. We used a 4-gram language model with modified Kneser-Ney discounting for all experiments, which is trained on the target language text of the bilingual training data.

In this section, we investigate the different aspects of the models and methods presented in this chapter. We will focus on the proposed leaving-one-out technique and show that it helps in finding good phrasal alignments on the training data that lead to improved translation models. Our final results show an improvement of 1.4 BLEU over the heuristically extracted phrase model on the test data set.

In Section 4.8.1 we discussed several methods which aim to overcome the over-fitting problems described in [DeNero & Gillick$^+$ 06]. Table 4.4 shows the translation scores of the count model on the development data after the first training iteration for both of the leaving-one-out strategies we introduced, and for training without leaving-one-out with

Table 4.4: Comparison of different training setups for the count model on WMT 2008 Dev.

Type of training	max phr.len.	BLEU[%]	TER[%]
heuristic	6	25.7	61.1
no leaving-one-out	2	25.2	61.3
	3	25.7	61.3
	4	25.5	61.4
	5	25.5	61.4
	6	25.4	61.7
standard leaving-one-out	6	26.4	60.9
length-based leaving-one-out	6	26.5	60.6

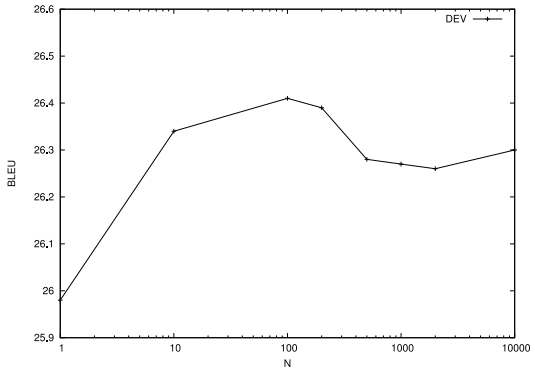


Figure 4.5: Performance on WMT 2008 Dev in Bleu of the count model plotted against size n of n-best list on a logarithmic scale.

different restrictions on phrase length. We can see that by restricting the source phrase length to a maximum of 3 words, the trained model is close to the performance of the heuristic phrase model. With the application of leaving-one-out, the trained model is superior to the baseline, the length-based strategy performing slightly better than the standard leaving-one-out. For these experiments the count model was estimated with a 100-best list.

The count model we described in Section 4.11.1 estimates phrase translation probabilities using counts from the n-best phrase alignments. For smaller n, the resulting phrase table contains fewer phrases and is more deterministic. For higher values of n, more competing alignments are taken into account, resulting in a bigger phrase table and a smoother distribution. We can see in Figure 4.5 that translation performance improves by moving from the Viterbi alignment to the n-best alignments. The variations in performance with sizes between $n = 10$ and $n = 10000$ are less than 0.2 Bleu. The maximum is reached for $n = 100$, which we used in all subsequent experiments. An additional benefit of the count model is the smaller phrase table size compared to the heuristic phrase extraction. This is consistent with the findings of [Birch & Callison-Burch$^+$ 06]. Table 4.5 shows

Table 4.5: Phrase table size of the count model for different n-best list sizes, the full model and for heuristic phrase extraction.

N	# phrases	% of full table
1	4.9M	5.3
10	8.4M	9.1
100	15.9M	17.2
1000	27.1M	29.2
10000	40.1M	43.2
full	59.6M	64.2
heuristic	92.7M	100.0

the phrase table sizes for different values of n. With $n = 100$, we retain only 17% of the original phrases. Even for the full model, we do not retain all phrase table entries. Due to pruning in the forced alignment step, not all translation options are considered. As a result, experiments can be done more rapidly and with less resources than with the heuristically extracted phrase table. Also, our experiments show that the increased performance of the count model is partly derived from the smaller phrase table size. In Table 4.6 we can see that the performance of the heuristic phrase model can be increased by 0.6 BLEU on WMT 2008 Test by filtering the phrase table to contain the same phrases as the count model and reoptimizing the log-linear model weights. The experiments on the number of different alignments taken into account were done with standard leaving-one-out.

The final results are given in Table 4.6. We can see that the count model outperforms the baseline by 0.8 BLEU on WMT 2008 Dev and 0.9 BLEU on WMT 2008 Test after the first training iteration. The performance of the filtered baseline phrase table shows that part of that improvement derives from the smaller phrase table size. Application of cross-validation (cv) in the first iteration yields a performance close to training with leaving-one-out (l1o), which indicates that cross-validation can be safely applied to higher training iterations as an alternative to leaving-one-out. The weighted count model underperforms the simpler count model. A second iteration of the training algorithm shows nearly no changes in BLEU score, but a small improvement in TER. Here, we used the phrase table trained with leaving-one-out in the first iteration and applied cross-validation in the second iteration. Log-linear interpolation of the count model with the heuristic yields a further increase, showing an improvement of 1.3 BLEU on WMT 2008 Dev and 1.4 BLEU on WMT 2008 Test over the baseline. The interpolation weight is adjusted on the development set and was set to $\psi = 0.6$. Integrating both models into the log-linear framework (feat. comb.) yields a BLEU score slightly lower than with the fixed interpolation on both WMT 2008 Dev and WMT 2008 Test. This might be attributed to deficiencies in the tuning procedure. The full model, where we extract all phrases from the search graph, weighted with their posterior probability, performs comparable to the count model with a slightly worse BLEU and a slightly better TER.

Table 4.6: Final results for the heuristic phrase table filtered to contain the same phrases as the forced alignment phrase model (heuristic (filtered)), the forced alignment model trained with leaving-one-out and cross-validation, and the full model. Further, scores for fixed log-linear interpolation of the forced alignment model trained with leaving-one-out with the heuristic as well as a feature-wise combination are shown.

Setup	WMT 2008 Dev		WMT 2008 Test	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]
heuristic	25.7	61.1	26.3	60.9
heuristic (filtered)	26.0	61.6	26.9	61.2
forced alignment (l1o)	26.5	60.6	27.2	60.5
forced alignment (cv)	26.4	60.7	27.0	60.7
forced alignment (full)	26.3	60.0	27.0	60.2
interpolation	27.0	59.4	27.7	59.2

4.12.2 Results on other language pairs

In addition to the experiments on the WMT 2008 German-English data, where we analyzed the aspects of our training procedure in detail, we were also interested in the performance of the phrase model training on other language pairs. We chose to evaluate on the Arabic-English 300k NIST and Chinese-English FBIS corpora to have additional results on other languages. Tables 4.7 and 4.8 show that the improvements reported in this work are not limited to the German-English language pair.

Table 4.7: Results of the phrase model training for the Arabic-English 300k NIST data.

Setup	NIST'06		NIST'08		NIST'09	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	42.7	50.2	40.3	52.3	42.7	49.2
learned phrases, no fallback	42.8	50.3	40.1	52.4	42.7	49.2
learned phrases	43.0	49.9	40.9	51.6	43.5	48.4

Table 4.8: Results of the phrase model training for the Chinese-English FBIS data.

Setup	NIST'06		NIST'08	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	21.5	69.8	16.4	73.7
learned phrases	21.9	69.5	17.0	73.4

With improvements of up to 0.8 BLEU points for Arabic-English and 0.6 BLEU points for Chinese-English, the overall improvements are smaller than for German-English (and not significant at $p = 0.10$). This can be attributed to two main causes.

1. The match between training and test data is much higher for WMT 2008 German-English, where the training and test data are taken from exactly the same sources. Due to the standards given to the editors for the European parliamentary proceedings, there is a higher degree of consistency throughout the dataset. Better learning of the training data is likely to lead to improvements on the test data. For Arabic-English 300k NIST and Chinese-English FBIS corpora, there is a higher mismatch between the train and the test data, as they come from completely different sources, with each translation done according to its own specific guidelines.
2. Alignment, especially for Chinese-English is more difficult than for the WMT 2008 German-English data. For the first iteration of the alignment, 4.6% of the sentences in the Chinese-English FBIS data cannot be aligned without fallback. This is an indication of a more difficult alignment problem, resulting in a smaller translation quality increase. Even for the Arabic-English 300k NIST data presented in Table 4.7, using fallback clearly improves the training result.

4.12.3 Conclusion

We have shown that training phrase models can improve translation performance on a state-of-the-art phrase-based translation model. This is achieved by training phrase translation probabilities in a way that they are consistent with their use in translation. A crucial aspect here is the use of leaving-one-out to avoid over-fitting. We have shown that the technique is superior to limiting phrase lengths and smoothing with lexical probabilities alone.

While models trained from Viterbi alignments already lead to good results, we have demonstrated that considering the 100-best alignments allows to better model the ambiguities in phrase segmentation.

The proposed techniques are shown to be superior to previous approaches that only used lexical probabilities to smooth phrase tables or imposed limits on the phrase lengths. On the WMT08 Europarl task, we show improvements of 0.9 Bleu points with the trained phrase table and 1.4 Bleu points when interpolating the newly trained model with the original, heuristically extracted phrase table. In Ter, improvements are 0.4 and 1.7 points. For Arabic-English, improvements in translation quality range from 0.3 to 0.8 Bleu points (0.3 – 0.8 Ter points), for Chinese-English Bleu improves by 0.4 to 0.6 points (0.3 Ter points).

In addition to the improved performance, the trained models are smaller, leading to faster and smaller translation systems.

5 Smoothing for Phrase Models

In this chapter we investigate the effect of existing and new phrase-table smoothing techniques. Smoothing in general is a technique to cope with sparsity problems in data. Even though large amounts of texts are used to train translation systems, there are still many events that are only seen once. It is difficult to correctly estimate the probability of these events. For phrase models described in Equation 3.6, the conditional probabilities for the source or the target phrase pairs that have been only seen once are 1, probably largely overestimating their true probability. Figure 5.1 show a cumulative histogram of the phrase counts for the Arabic-English 300k NIST training corpus for phrases of a maximum source phrase length of 6 words and a maximum target phrase length of 12. Already 81.2% of all phrase pairs extracted have a count of one or less[1]. A similar trend can be seen for the source phrase counts where 55% of the source phrases have been seen only once.

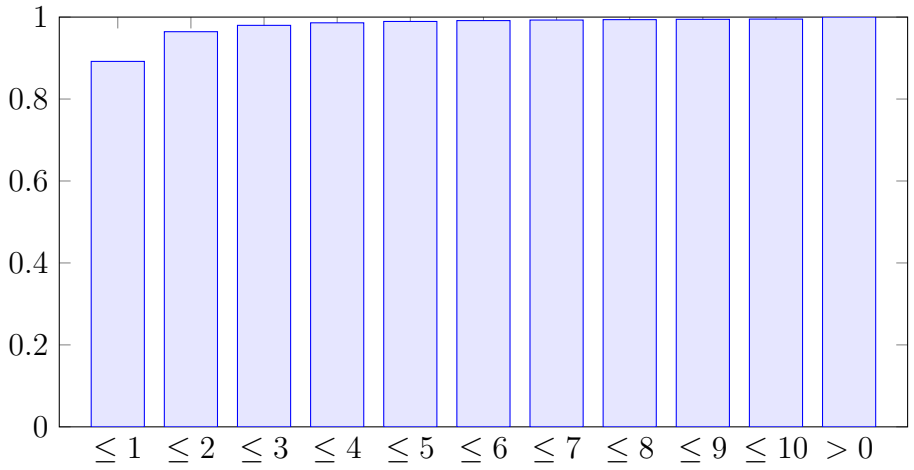


Figure 5.1: Distribution of bilingual phrase counts for the Arabic-English 300k NIST data.

Several approaches have been taken to tackle this problem. The first method that can be interpreted as a smoothing technique is the use of forward and backward conditional probabilities $p(\tilde{e}|\tilde{f})$ and $p(\tilde{f}|\tilde{e})$ [Koehn & Och$^+$ 03]. Using the relative frequency estimates for phrase translation probabilities, a rare source language phrase $\tilde{f}$ aligned to a more frequent target phrase $\tilde{e}$ will have a high probability $p(\tilde{e}|\tilde{f})$ but a low probability $p(\tilde{f}|\tilde{e})$. This smoothing technique will only work for phrases that are rare only in either the source or target data. It does not work in cases where both the source and target phrase are infrequent.

[1]See Section 3.2.1 for an explanation on how phrases are counted and how fractional counts can occur.

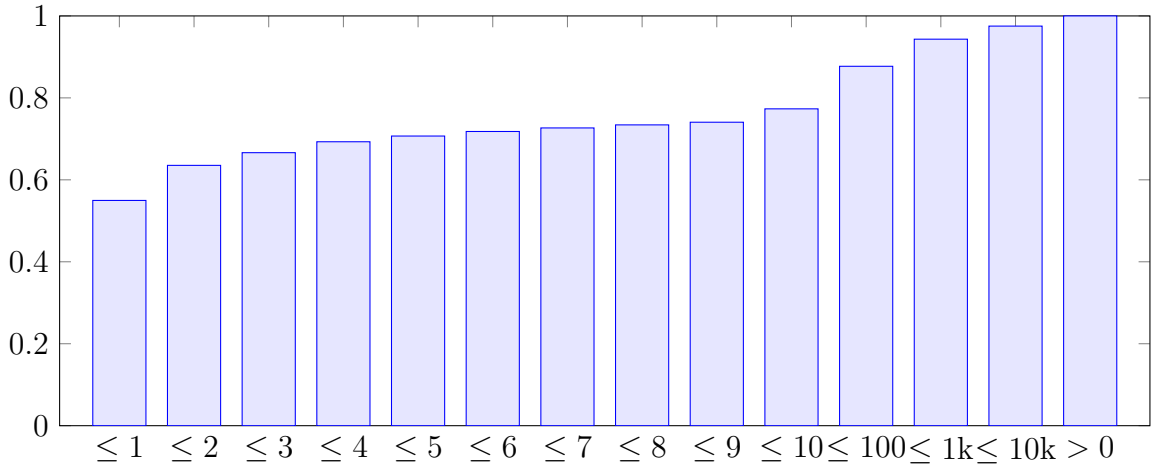


Figure 5.2: Distribution of source phrase counts for the Arabic-English 300k NIST data.

To also address cases of infrequent phrase pairs where thge source and target phrase are also infrequent, additional smoothing techniques have been considered. A comparison of the most common ones can be found in [Foster & Kuhn$^+$ 06]. The standard technique in the phrase-based and related translation models is smoothing based on within-phrase word lexica. Used already with the first phrase-based translation models [Och 03] [Koehn & Och$^+$ 03], the training of these models has not been thoroughly investigated the in literature.

This chapter, will describe and compare existing smoothing methods and propose additional, improved smoothing techniques that show improved translation model performance. We will distinguish between independently trained smoothing models and heuristics (Section 5.2) and our newly proposed models which have been trained in the forced alignment process (Section 5.3). And show that integrated training of smoothing models improves translation performance. Finally, we will also show ł1o based estimation of smoothing parameters in Section 5.4.

5.1 Publications and Team Work

In this chapter there are sections that contain work that was published without contributions from the author of this thesis, sections that have been published by the author of this thesis, and previously unpublished work.

The models described in Section 5.2.1, only contains work that was published without contributions from the author of this thesis.

The method describe in Section 5.2.2 has been published as part of [Mauser & Zens$^+$ 06] and is augmented here with additional experiments. The idea for this method was developed with Richard Zens who also contributed the implementation. The design and execution of the experimental evaluation and the analysis of the results was done by the author of this thesis, both for the original publication and for the additional work presented in this thesis.

Sections 5.3 and 5.4 describe new methods that have not been published prior to this thesis. The idea for these models was developed by the author of this thesis who also did the implementation, empirical evaluation and analysis.

5.2 Independently Trained Smoothing Methods

This section describes phrase table smoothing techniques that rely on models or heuristics that do not depend on phrase-level forced alignment training. The models used here are trained independently from the phrase model by using word-based models or simple heuristics depending on the phrase counts.

5.2.1 Within-Phrase Lexical Models

A standard setup for a phrase-based statistical machine translation system contains lexical smoothing by computing IBM1-like scores on all words within the phrase. Recall that, in phrase-based translation, we have a segmentation of the source sentence and the produced translation into K phrases with segmentation

$$k \mapsto s_k := (i_k, b_k, j_k) \qquad \text{(cf. 3.3)}$$

Following Equations (4.10) and (4.11) in [Zens 08], the formulas for computing these lexical probabilities are for a single phrase:

$$p_{\text{Lex}}(\tilde{f}_k|\tilde{e}_k) = \prod_{j=b_k}^{j_k} \left[p(f_j|e_0) + \sum_{i=i_{k-1}+1}^{i_k} p(f_j|e_i) \right]. \tag{5.1}$$

For the entire sentence in the "standard" direction:

$$p_{\text{Lex}}(f_1^J|e_1^I; s_1^K) = \prod_{k=1}^{K} p_{\text{Lex}}(\tilde{f}_k|\tilde{e}_k). \tag{5.2}$$

For the inverse direction we get:

$$p_{\text{iLex}}(\tilde{e}_k|\tilde{f}_k) = \prod_{i=i_{k-1}+1}^{i_k} \left[p(e_i|f_0) + \sum_{j=b_k}^{j_k} p_{Lex}(e_i|f_j) \right] \tag{5.3}$$

and

$$p_{\text{iLex}}(e_1^I|f_1^J; s_1^K) = \prod_{k=1}^{K} p_{\text{iLex}}(\tilde{e}_k|\tilde{f}_k). \tag{5.4}$$

Note that we ignore the uniform alignment probabilities that would be part of the original IBM1.

In its formulation as used in [Och & Ney 04] and [Koehn & Hoang$^+$ 07], the within-phrase word alignment is used for lexical scoring. The within-phrase word alignment is obtained from the word-aligned training data in phrase extraction. In this case, the lexical probability for a phrase $(\tilde{f}_1^J, \tilde{e}_1^I, \tilde{A})$ is computed as

$$p_{\text{aLex}}(\tilde{e}|\tilde{f}, \tilde{A}) = \prod_{i=1}^{I} \frac{1}{|\{j|(j,i) \in \tilde{A}\}|} \sum_{j:(j,i)\in\tilde{A}} p(e_i|f_j) \tag{5.5}$$

While [Zens 08] state that they use IBM4 lexical probabilities trained with GIZA++, others (for example [Koehn & Hoang$^+$ 07]) learn word translation probabilities by counting aligned word pairs in the word-aligned training data that is also used for phrase extraction. This method is also described in Section 5.3.3 in [Koehn 10]. When using IBM models to estimate lexicon probabilities in training, it is important to note that this estimation is done on the sentence level. This results in a mismatch between model training and model use as describe in Equations (5.2) and (5.4) where only the within-phrase context is used. We will show in Section 5.3 how the lexical probabilities can be learned using consistent training. As our baseline, we will consider both variants using lexicons learned as part of the IBM models and using the word-aligned data.

5.2.1.1 Word Lexicon Estimation from Word-Aligned Data

Given a training corpus $((f_n)_1^{J_n}, (e_n)_1^{I_n}, A_n)$, $n = 1, \ldots, N$, which is a list of triples consisting of the source sentence $(f_n)_1^{J_n}$, the target sentence $(e_n)_1^{I_n}$, and a word alignment $A_n = \{(j,i) : j, i \quad \text{are aligned}\}$, we estimate the lexical probabilities for a specific source word f given a target word e as

$$p(f|e) = \frac{N_{lex}(f,e)}{N(f)} \tag{5.6}$$

with

$$N_{lex}(f,e) = \sum_{n=1}^{N} \sum_{(j,i)\in A_n} w_n \cdot \delta(e_i, e) \cdot \delta(f_j, f) \cdot |(j,i') : (j,i') \in A_n|^{-1} \tag{5.7}$$

with $\delta(a,b)$ being the Kronecker delta. Note that we collect fractional counts for 1-to-many alignments. The counting and normalization is also done for the inverse direction. Each observation n also has a weight w_n which can be used to give different weights to each training observation. In the default case, w_n will be 1 for all n.

5.2.1.2 Word Lexicon Estimation using IBM1 Training

When using the IBM1 model for training the lexical smoothing, lexical probabilities are estimated using the EM algorithm as derived in [Brown & Della Pietra$^+$ 93]. The E-step at iteration k consists of computing

$$\gamma^{\{k\}}(f,e) = \sum_{n=1}^{N} \sum_{i=1}^{I} \sum_{j=1}^{J} w_n \cdot \delta(e_i, e) \cdot \delta(f_j, f) \cdot \frac{p^{\{k\}}(f_j|e_i)}{\sum_{i'=1}^{I} p^{\{k\}}(f_{j'}|e_i)} \tag{5.8}$$

and the M-step results in renormalizing the intermediate quantities $\gamma^{\{k\}}(f,e)$

$$p^{\{k+1\}}(f|e) = \frac{\gamma(f,e)}{\sum_f \gamma^{\{k\}}(f,e)}. \tag{5.9}$$

5.2.2 Phrase Count Features

Whether relative frequency phrase probability estimates are reliable, depends on the amount and quality of training data and the frequency of a phrase in the data. The probability of rare phrases tends to be over-estimated, if they occur, and under-estimated, if they don't. If they do not occur often, it might be due to errors originating from the translation or alignment errors. The log-linear framework we use in translation gives us an easy way to integrate this knowledge about the sparsity of our observations in the translation process by adding additional features. Following [Zens 08], We use binary features to indicate, if a phrase count $N(\tilde{f}_k, \tilde{e}_k)$ is above a given threshold τ:

$$h_{\mathrm{C},\tau}(f_1^J, e_1^I, s_1^K) = \sum_{k=1}^{K} [N(\tilde{f}_k, \tilde{e}_k) \leq \tau]$$

We use a square bracket operator $[\cdot]$ to convert a true or false statement into an integer [Graham & Knuth$^+$ 94]. The result is 1 if the statement is true, and 0 otherwise. In general, we use the following convention:

$$[\,\mathcal{C}\,] = \begin{cases} 1, & \text{if condition } \mathcal{C} \text{ is true} \\ 0, & \text{if condition } \mathcal{C} \text{ is false} \end{cases} \tag{5.10}$$

The value τ determines the threshold for the phrase count feature. In the evaluation system, we used three phrase count features with τ manually chosen and ranging as 1.0, 2.0, and 3.0. As actual phrase count values are fractional, fractional thresholds can also be used.

5.3 Word Lexicon Models with Consistent Training

In this section, we will describe how the within-phrase lexical translation models can be learned in a consistent training framework. We are considering two dimensions of the problem. First, we have the estimation which can be done using IBM1 or by counting the aligned word pairs in our training data as in [Koehn & Hoang$^+$ 07]. As a second dimension, we have two ways of weighting our training examples: (1) a uniform weighting among all unique phrase pairs or: (2) using the posterior probability of the phrase. We will formulate the equations for the latter case and treat the uniform weighting as a special case.

Starting from the counting in Equation (5.6) we change the way of accumulating our word lexicon statistics. Instead of collecting counts over all aligned words in the sentence, we

use the within-phrase alignment obtained during the heuristic phrase extraction:

$$N_{pLex}(f,e) = \sum_{(\tilde{f},\tilde{e},\tilde{A})\in\mathcal{P}} \sum_{(j,i)\in\tilde{A}} w(\tilde{f},\tilde{e}) \cdot \delta(e_i,e) \cdot \delta(f_j,f) \cdot |(j,i') : (j,i') \in \tilde{A}|^{-1}. \quad (5.11)$$

We will set the instance weight $w(\tilde{f},\tilde{e}) = N(\tilde{f},\tilde{e})$ for the weighted learning and $w(\tilde{f},\tilde{e}) = 1$ for the uniform weighting. We obtain normalized probabilities by marginalizing over all possible translations:

$$p_{pLex}(f|e) = \frac{N_{pLex}(f,e)}{\sum_{f'} N_{pLex}(f,e)} \quad (5.12)$$

In a similar way, we adapt the E-step from the IBM1 training Equation (5.8) by changing the summation and the weighting:

$$\gamma(f,e) = \sum_{(\tilde{f},\tilde{e})\in\mathcal{P}} \sum_{i=1}^{I} \sum_{j=1}^{J} w_n(\tilde{f},\tilde{e}) \cdot \delta(e_i,e) \cdot \delta(f_j,f) \cdot \frac{p^{\{k\}}(f_j|e_i)}{\sum_{i'=1}^{I} p^{\{k\}}(f_{j'}|e_i)} \quad (5.13)$$

The M-step remains unchanged.

5.4 Absolute Discounting with Interpolation

Discounting methods have been highly successful in language modeling [Chen & Goodman 98]. One of the most widely used smoothing methods is absolute discounting where a fixed value b is subtracted from every observed count while leaving the marginal counts intact [Kneser & Ney 95]. The probability mass that is freed in this way is then redistributed to a generalized distribution $\beta(\cdot,\cdot)$. When using interpolation to combine the two distributions, this is typically called Kneser-Ney smoothing [Chen & Goodman 98]. Following [Foster & Kuhn$^+$ 06], we adapt this smoothing method to the phrase translation probabilities. Instead of using a uniform phrase model as the generalized distribution, we use the lexical probability (Equation 5.1).

$$p_s(\tilde{e}|\tilde{f}) = \frac{N(\tilde{e},\tilde{f}) - b}{N(\tilde{f})} + b \cdot \frac{N_{>0}(\tilde{e},\tilde{f})}{N(\tilde{f})} \cdot p_{Lex}(\tilde{e}|\tilde{f}) \quad (5.14)$$

We need the counts $N_{>0}(\tilde{f})$ to determine how much count mass we have removed from the original distribution $\frac{N(\tilde{e},\tilde{f})}{N(\tilde{f})}$. This is computed as

$$N_{>0}(\tilde{f}) = \sum_{\tilde{e}:N(\tilde{e},\tilde{f})>0} \min\left\{b, N(\tilde{e},\tilde{f})\right\} \quad (5.15)$$

In phrase extraction, fractional phrase counts can be generated by the heuristic dealing with empty words as described in Section 3.2.1.

5.5 Results

This section compares the various smoothing methods that were described in this chapter. The analysis is divided into two main subsections: Section 5.5.1 discusses the impact of the independently obtained smoothing methods presented in Sections 5.2 and 5.4. Section 5.5.2 analyzes the impact of the consistently trained smoothing methods described in Section 5.3.

For all results in this section, we run 5 iterations of lattice-based MERT with 20 random restarts each.

5.5.1 Independently Trained Models and Heuristics

Table 5.1: Independently trained word lexica for the Arabic-English 300k NIST data.

	NIST'06		NIST'08		NIST'09	
Setup	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
no smoothing	39.1	53.7	37.2	54.9	39.4	52.5
count	40.5	51.8	38.7	53.3	41.5	50.1
Moses lexicon	39.2	52.8	37.4	54.1	40.5	51.1
word lexicon	41.5	51.5	38.8	53.5	41.6	50.7
IBM1 lexicon	42.3	50.4	39.8	52.3	42.4	49.4
IBM1 lexicon+count	42.7	50.2	40.3	52.3	42.7	49.2

Table 5.1 shows an overview of the results of the independently trained models. Starting with a translation system without any smoothing we see that both, count features and word lexica, improve the translation quality. We use three phrase count features with thresholds $\tau = 1, 2, 3$.

Interestingly, the "Moses" variant of the lexicon model, here labeled "Moses lexicon", where the scoring is only done along the alignment points within the phrase, performs worse than the other lexicon models and also worse than the count features. This fact seems to be in part due to the scoring in search and in part due to the training of the model itself. When comparing to the intermediate result, "word lexicon", where we use the same lexicon as for "Moses lexicon", but perform IBM1-type scoring as in Equation (5.1), we get a largely improved result. Significance tests using bootstrap sampling show that all smoothing methods are significantly better (with $p = 0.05$) than no smoothing. Among the different smoothing setups, IBM1 smoothing is significantly better ($p = 0.05$) than other methods.

The best word lexicon setup in this comparison is to use a sentence-level trained IBM1 model as obtained from GIZA++ and score the phrases using all words, disregarding the alignment. This result is labeled "IBM1 lexicon" in Table 5.1. A combination of the best lexicon model setup and the phrase count features gives an additional small improvement.

There are several possible reasons for this. First, the reason for the better scoring is that the word alignment is not always correct. There might be wrongly aligned words and

unaligned words, that are not taken into account by the "Moses" type scoring. For the lexicon model itself, relying only on the aligned words from the training data results in a very peaked distribution that only assigns probability mass to a few translation options for each lexicon entry. For the Arabic-English 300k NIST task, the alignment-based lexicon model contains only 679k entries (approximately 6 translation options per source word on average) compared to 25.5 million entries of the IBM1 lexicon (approximately 227 translation options per source word on average). Intuitively this means that the IBM1 lexicon can provide a distinction for much more source/target word pairs, where the alignment-based lexicon only provides probability 0 in many cases.

In the above experiments, the word lexica were included in the log-linear model combination. This means that phrase translation probabilities were log-linearly interpolated with the word lexicon probabilities. In other natural language applications such as language modeling, it has been beneficial to perform linear interpolation and discounting as smoothing. The procedure we applied here for translation is described in Section 5.4. When doing the experiments, we tried several discounting parameters. The results of the experiments are shown in Table 5.2.

Although the absolute discounting approach with interpolation proves successful as a smoothing technique, and improves translation quality over an unsmoothed baseline, the performance is behind the conventional, log-linear interpolation of the lexicon models. One possible reason for this is that the log-linear interpolation has the advantage that the interpolation parameters are optimized for best error rates. The linear interpolation with absolute discounting cannot be easily optimized in the same way.

All improvements over the baseline are significant at $p = 0.05$. The IBM1 model without discounting is only significantly better than the baseline and the discounting with 0.7.

Table 5.2: Absolute discounting for the Arabic-English 300k NIST data.

Setup	NIST'06		NIST'08		NIST'09	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
no smoothing	39.1	53.7	37.2	54.9	39.4	52.5
IBM1 lexicon, abs.disc. = 0.7	40.2	52.5	37.6	54.3	40.0	51.8
IBM1 lexicon, abs.disc. = 0.89	41.7	51.5	39.0	53.3	42.0	50.6
IBM1 lexicon, abs.disc. = 1.0	41.9	51.2	38.9	53.4	41.9	50.3
IBM1 lexicon+count	42.7	50.2	40.3	52.3	42.7	49.2

5.5.2 Word lexicon models with consistent training

The consistent training of the smoothing models presented in this section was described in Section 5.3. In this experimental exploration, we keep the scoring function fixed to be the IBM1 like within-phrase scoring as in Equation 5.1. In the first experiment, (Table 5.3, we attempt to learning the alignment-based lexicon where only aligned words are consitered for the lexicon score. The final word alignment from GIZA++ training, which is normally used for this score, is replaced by the word alignment that is obtained during

the forced alignment training. We have two variants of the estimation: (1) we weight each observation by the posterior probability of the forced alignment, labeled "learned lexicon (weighted)" in Table 5.3; (2) each observation is counted only at its first occurrence. The second variant is labeled "learned lexicon" in Table 5.3. The first results show the effect of solely training the word lexicon model, without changing the phrase table. When doing a combined learning of the word lexicon and phrase table, we obtain the result labeled "learned (phrases+lexicon)".

Both variants show slightly higher BLEU scores than the baseline, but the difference between the two variants is very small. Using bootstrap sampling we find that none of the results is significantly better than the baseline ($p = 0.1$)

Table 5.3: Smoothing comparison for the Arabic-English 300k NIST data

	NIST'06		NIST'08		NIST'09	
Setup	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
alignment-based lexicon	42.1	51.1	39.5	53.1	42.3	50.0
learned lexicon (weighted)	42.9	50.4	40.0	52.4	42.8	49.3
learned lexicon	42.7	50.5	39.8	52.5	42.9	49.3
learned (phrases+lexicon)	42.8	50.5	40.1	52.4	42.7	49.2

In the second set of experiments, we learn the lexicon models by using the forced alignment to estimate the within-phrase IBM1 models. Table 5.4 shows the results of these experiments. In contrast to the alignment-based training, there seems to be no added benefits in using the forced alignment to learn the within-phrase lexicon models. The increased modeling consistency does not seem to result in increased translation quality.

Table 5.4: Smoothing comparison for the Arabic-English 300k NIST data

	NIST'06		NIST'08		NIST'09	
Setup	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
IBM1 lexicon+count	42.7	50.2	40.3	52.3	42.7	49.2
learned IBM1 (weighted)	42.7	50.6	40.1	52.5	42.8	49.4
learned IBM1	42.7	50.6	39.9	52.6	42.8	49.4
learned (phrases+IBM1)	42.6	50.6	40.4	52.4	42.7	49.4

6 Reordering Models

6.1 Introduction

One of the problems in phrase-based statistical machine translation is dealing with divergence in word order of the source and target language. Most state-of-the-art systems use a simple distortion penalty to penalize non-monotonicity. Additionally, several lexicalized approaches have been proposed, which provide a more fine-grained way of modeling the word order [Zens & Ney 06], [Zhang & Zens$^+$ 07], [Al-Onaizan & Papineni 06], [Koehn & Hoang$^+$ 07], [Costa-Jussà & Fonollosa 06].

Previous approaches have mostly relied on word-aligned data or linguistic annotation for estimating reordering probabilities. The word alignments, however, were obtained using models that were mostly unaware of the phrases that are used in translation. Our approach differs from the existing attempts to learn reordering models in that we use a full phrase-based training approach to train the reordering model probabilities. In this way, the model training is more similar to the decoding process and also benefits from the other models used in translation.

Lexicalization of reordering, looking at the words involved in the reordering sequence on the source and target side can be motivated by observing reordering patterns between languages. A simple example would be translating a question from German to English where the main verb in English moves to the end.

In this work, we extend the previously proposed lexicalized reordering model provided by the Moses open-source decoder [Koehn & Hoang$^+$ 07] and we propose a novel reordering model which is a lexicalized, learned variant of the jump distance model. Rather than relying on word alignments for training, we make use of the forced alignment procedure described in Chapter 4 to align the training data and compute model expectations.

6.2 Related Work

Standard reordering models in most phrase-based decoders are just based on the distance of the final source position of the last translated phrase to the first covered source position in the next phrase. This penalizes deviations from the monotone decoding path. A maximum distortion distance is given as a parameter to constrain the search space. Jumps larger than this distance are either completely forbidden or more heavily penalized. Whenever jumps over a certain length are completely forbidden and pruning is used in translation, it is possible that given the current beam, no complete translation can be found, as the last untranslated source positions are beyond the permitted jump distance.

Therefore, a hard limit on jump distances is not advisable.

Typically, jumps below the limit are penalized linearly, jumps beyond the limit are penalized quadratically. The distance between two source positions j and j' is defined as $d(j, j') = j - j' - 1$ with $j \neq j'$. For simplicity, we will abbreviate the jump distance by d. The distortion penalty for distance d is then computed as

$$q_{dist}(d, D) = \begin{cases} |d| & \text{if } |d| < D \\ |d| + |d|^2 & \text{else} \end{cases} \tag{6.1}$$

Using an unlexicalized, distance-based penalty in reordering is only a rough approximation of the natural translation process. As illustrated with the examples in Section 6.1 the reordering does depend on the involved words. Another example is translating between English and German where verbs can be moved to the very end of the sentence. In these cases, long jumps would be penalized very strongly while being perfectly reasonable.

The Moses phrase-based decoder [Koehn & Hoang$^+$ 07] uses a lexicalized phrase orientation model. In the most general variant, the location of the previous and the following phrase is modelled with three classes: `monotone`, `swap`, and `discontinuous`. We refer to this set as MSD. We will describe this model in detail in Section 6.6.

Our approach differs from the existing attempts to learn reordering models in that we use full phrase-based training to train the model probabilities. Previous approaches have mostly relied on word-aligned data for estimating reordering probabilities. The word alignments, however were obtained using models that were unaware of the phrases that are used in translation.

Other phrase orientation models have been proposed in the literature. Similar to the MSD model, [Tillmann & Zhang 05] propose three classes "Left", "Right" and "Neutral", which correspond to the classes `swap`, `monotone` and `discontinuous`. The models are trained using discriminative classifiers. The extensions in [Tillmann & Zhang 06] are similar to this work, where translation models and reordering models are trained jointly. However in [Tillmann & Zhang 06], a discriminative framework is used and the overall performance of the system is not compared to a standard phrase-based baseline.

6.3 Publications and Team Work

The novel work presented in this chapter was not published prior to this thesis. Unless specifically marked, ideas, concepts, implementation, experimentation, evaluation, and analysis has been performed by the author of this thesis.

Novel models are presented in Sections 6.5 and 6.6, with some additional explanations in Section 6.4.

While some of the ideas had been explored before in the literature (see Section 6.2), the idea of learning the model parameters in the way described in this chapter is new.

All implementation of training and the integration into the statistical machine translation system were done by the author of this thesis. The same holds for the design, execution, verification and analysis of all the experimental evaluation.

6.4 Reordering Model Estimation

For all the following models, we use a similar procedure for learning. We perform an initial word alignment using GIZA++, do standard phrase extraction, and automatically tune the model scaling factors for optimal performance on the development set.

We then employ a forced alignment training procedure to produce a phrase alignment of the training data using optimized scaling factors. Even though the search space in the alignment is constrained by the source and the target sentence and the existing phrase model used in initialization, a full search over all possible alignments would be computationally to expensive. Therefore, pruning is applied also in training. Furthermore, we restrict the reordering in alignment in the same way as it is restricted in search. For all experiments, we use a soft jump distance limit of $D = 10$ source positions. Soft means, that these jumps are not completely impossible during decoding, but receive a very high penalty. This is in rare cases neccessary to allow the decoder to find a translation.

The resulting alignment can be represented as the phrase sequence of the single-best alignment or as an alignment graph containing all hypotheses considered in the search space. This alignment information is used in different ways, depending on the model. All of them have in common that they collect counts from the phrase alignment. We collect these counts by summing the phrase alignment probabilities over all sentences (f_1^J, e_1^J) in the training data. Using the function from Equation 3.1, we define the reordering event count for source phrase $\tilde{f}$, target phrase $\tilde{e}$, and reordering r as

$$N_{\mathrm{FA}}(r, \tilde{f}, \tilde{e}) := \sum_{(f_1^J, e_1^I)} \sum_{s_1^k} \sum_{k=1}^{K} \frac{\sum_{m=1}^{M} \lambda_m h_m(e_1^I, s_1^K, f_1^J)}{\sum_{s'^K_1} \sum_{m'=1}^{M} \lambda_{m'} h_{m'}(e_1^I, s'^K_1, f_1^J)} \cdot \delta(\tilde{f}_k, \tilde{f}) \cdot \delta(\tilde{e}_k, \tilde{e}) \cdot \delta(r, R(k, s_1^K)). \quad (6.2)$$

For notational simplicity, we define $\tilde{f}_k$ and $\tilde{e}_k$ as the source and target phrase aligned in segment k.

We use the general reordering event r that can be computed using the function $R(k, s_1^K)$ with the segmentation and alignment information. The type of reordering events can vary according to the reordering model used. For example in the case of a conventional distance-based distortion model, r would be the jump distance. We will describe the training of the individual reordering models in the following sections.

6.5 Learned Jump Distance Model

The jump distance model defined in Equation 6.1 is neither normalized, nor conditioned on any lexical information. No information is retained from the training data, not even the general distribution of the reorderings. To overcome this loss of information, we learn a phrase-conditioned jump model that makes the jump direction and distance dependent

on the phrase. Using the maximum distortion distance D, our model is defined as

$$p_{dist}(d|\tilde{f},\tilde{e}) = \frac{N_{\mathrm{FA}}(d,\tilde{e},\tilde{f})}{N_{\mathrm{FA}}(\tilde{f},\tilde{e})}. \tag{6.3}$$

We use a special value for cases where $d \geq D$, to account for out-of-bounds jumps that are permitted by the baseline model. The counts $N_{\mathrm{FA}}(\cdot)$ are collected from our phrase alignment. Here, $N_{\mathrm{FA}}(d,\tilde{f},\tilde{e})$ is the count of the particular combination of jump distance d and phrase pair $(\tilde{f},\tilde{e})$ in our phrase alignment and $N_{\mathrm{FA}}(\tilde{f},\tilde{e})$ is the count of the phrase pair.

Our observations tend to be quite sparse. A phrase pair might have only been seen in a few configurations and thus, the estimation of the jump parameters are unreliable. Therefore we linearly interpolate the distortion probability from Equation 6.1 with an unconditioned jump distribution

$$p_{dist}(d) = \sum_{(\tilde{e},\tilde{f})} \frac{N_{\mathrm{FA}}(d,\tilde{e},\tilde{f})}{N_{\mathrm{FA}}(\tilde{f},\tilde{e})}. \tag{6.4}$$

The resulting interpolated jump model which is used in the translation system then is

$$p_{sdist}(d|\tilde{f},\tilde{e}) = (1-\alpha)\cdot p_{dist}(d|\tilde{f},\tilde{e}) + \alpha \cdot p_{dist}(d) \tag{6.5}$$

with interpolation weight α, $0 \leq \alpha \leq 1$. Experimentally, we found that a value of $\alpha = 0.6$ resulted in the best translation performance on the development set.

To compute the jump distance from the previous phrase to the next, we only need the last covered source position of the previous phrase. This makes this model rather memory efficient in search.

6.6 Lexicalized Orientation Model

The most general variant of the lexicalized phrase orientation model used by Moses [Koehn & Hoang$^+$ 07], the location of the previous and the following phrase is modelled with three classes: `monotone`, `swap`, and `discontinuous`. The different classes are visualized in Figure 6.1. We will now give a detailed description of the Moses orientation model, as the available literature is very brief on this topic.

Formally, the three classes can be described by the start and end positions of the source phrases used in decoding. Given a bilingual phrase segmentation $k \mapsto s_k := (b_k, j_k)$, $k = 1,\ldots,K$, of a sentence in search with b_k being the first position of the source phrase, and j_k being the end of the source phrase, we can define the orientation of a phrase s_k with respect to the previous phrase s_{k-1}

$$\mathcal{O}_{\mathrm{MSD}}(s_k, s_{k-1}) = \begin{cases} \texttt{monotone} & \text{if } b_k = j_{k-1}+1 \\ \texttt{swap} & \text{if } j_k + 1 = b_{k-1} \\ \texttt{discontinuous} & \text{else} \end{cases} \tag{6.6}$$

For finding orientations of the first and the last segment, we assume $s_0 = (0, 0)$ and $s_{K+1} = (J + 1, I + 1)$.

Note that in the actual implementation in Moses, the `discontinuous` class is labelled `other`, as there is no explicit definition of `discontinuous`. To be consistent with the other literature, we will continue to use `discontinuous` in this work, whenever possible.

Moses allows for several variants of the orientation model. It can be one-sided, modelling only the orientation of the current with respect to the previous phrase $\mathcal{O}(s_k, s_{k-1})$ or the following phrase $\mathcal{O}(s_{k+1}, s_k)$, or two-sided using both, the orientation towards the preceding phrase and towards the following phrase. There are also variations in the number of reordering classes used by the model. Instead of the three classes described above, only `monotone` and `other` can be used. The context used for the reordering decision can be the full phrase pair $(\tilde{f}_k, \tilde{e}_k)$ or only on the source phrase $\tilde{f}_k$. In this work, we will only investigate the default setting in Moses which is bidirectional and conditioned on the phrase pair. We will refer to this orientation model as MSD model.

In the standard approach, the MSD model is extracted from the same word alignment that is used to extract the phrases. While the application of the MSD model in search is straightforward, the extraction from word-aligned data requires the handling of some special cases for unaligned words.

To describe, how phrase orientation counts can be extracted from word alignments, we first revisit the definition of a phrase. Given a source sentence f_1^J, a target sentence e_1^I, and an alignment

$$A = \{(j, i) : j \in \{1, \ldots, J\}, i \in \{1, \ldots, I\}\} \tag{6.7}$$

we follow [Och & Tillmann$^+$ 99] and [Koehn & Och$^+$ 03], and define a phrase as consistent with the alignment if there are no alignment points outside of the phrase in the same columns and rows as the phrase and there is at least one alignment within the phrase. Formally, we have defined a phrase $(f_{j_1}^{j_2}, e_{i_1}^{i_2})$ as being consistent with the alignment in Section 3.2.1:

$$\begin{aligned} &\forall (j, i) \in A : j_1 \leq j \leq j_2 \leftrightarrow i_1 \leq i \leq i_2 \\ \wedge &\exists (j, i) \in A : j_1 \leq j \leq j_2 \wedge i_1 \leq i \leq i_2. \end{aligned}$$

Given a phrase $(f_{j_1}^{j_2}, e_{i_1}^{i_2})$ that is consistent with the alignment, we can define the orientation with the preceding phrase in the MSD model as

$$\mathcal{O}_{\text{MSD}}(j_1, j_2, i_1, i_2, A) = \begin{cases} \texttt{monotone} \\ \quad \text{if } (j_1 - 1, i_1 - 1) \in A \wedge \\ \quad\quad \neg\exists (j, i_1 - 1) \in A : j > j_1 \\ \texttt{swap} \\ \quad \text{if } (j_2 + 1, i_1 - 1) \in A \wedge \\ \quad\quad \neg\exists (j, i_1 - 1) \in A : j < j_2 \\ \texttt{discontinuous} \\ \quad \text{else} \end{cases} \tag{6.8}$$

For the first and last phrase, we assume alignments at $(0, 0)$ and $(J+1, I+1)$, encouraging a monotone orientation at the beginning and end of the sentence.

The orientation counts are then accumulated in the same way as the phrase translation counts. In the case of the bidirectional phrase-pair-conditioned MSD model, we can compute the probability of an orientation of a phrase pair as

$$p(o|\tilde{f}, \tilde{e}) = \frac{N_{\text{MSD}}(o, \tilde{f}, \tilde{e}) + N_0}{\sum_{(\tilde{f}', \tilde{e}')} N_{\text{MSD}}(\tilde{f}', \tilde{e}') + |\mathcal{O}_{\text{MSD}}| \cdot N_0} \tag{6.9}$$

with the phrase orientation o being one of $\{\texttt{monotone}, \texttt{swap}, \texttt{discontinuous}\}$. Furthermore, we use $N_{\text{MSD}}(o, \tilde{f}, \tilde{e})$ as the joint count of the orientation o and the phrase pair $(\tilde{f}, \tilde{e})$, and $N_{\text{MSD}}(\tilde{f}, \tilde{e})$ as the count of the phrase pair in the aligned data. Even with the small number of classes that are used in the MSD model, the observations can be relatively sparse. Therefore the standard Moses training procedure smooth the counts towards a uniform distribution by adding a constant $N_0 = 0.5$. This is the model that we use as our baseline.

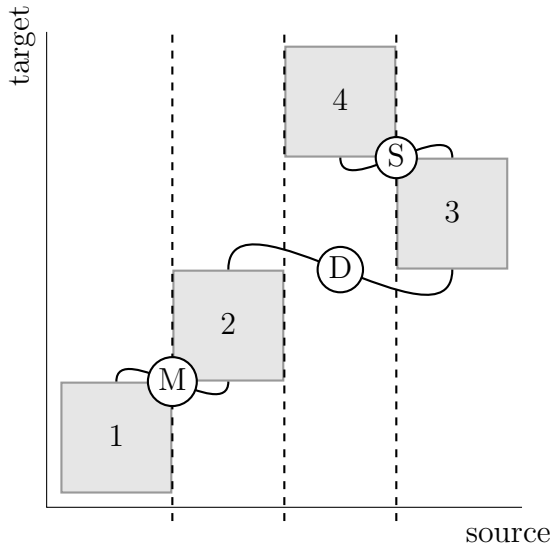


Figure 6.1: Illustration of phrase orientation for the MSD model using orientations `monotone` (M), `swap` (S) and `discontinuous` (D).

Note that full MSD modelling, with all three classes or modelling the orientation with the following phrase and not only the previous, requires a change in the recombination of phrase-based hypotheses compared to the simple, distance-based modelling or an orientation model with only `monotone` and `other` classes. In the latter case, the state information only needs to preserve the last position of the previous phrase to be able to compute all necessary information. In order to be able to detect a `swap` orientation, as defined in Equation 6.6, we also need the start position of the previous phrase, which therefore needs to be preserved in the state. Hypotheses with identical last positions of the previous phrase, but different first positions, cannot be recombined in search. The orientation with the

following phrase can only be computed in search, when the following phrase is actually considered. Therefore, when modelling the outbound orientation on the phrase identity, we also need to preserve this phrase identity in the state. Partial search hypotheses that differ in the last phrase used, but are otherwise identical, cannot be recombined.

Both implementation issues mentioned above result in an artificially large search space that potentially contains much less different alternative translations. When comparing to reordering methods that require less state information, either the same states have to be used in both cases or pruning parameters have to be adjusted in order to compensate for the difference. Here, we use the second approach and make sure that the pruning parameters were sufficiently large for all experiments.

Note that in Moses, the reordering model is conditioned on the "outgoing" phrase. During decoding, we model the reordering behavior to the right of the last translated phrase. This is consistent with the implementation of the decoding in Moses. The reordering could also be conditioned on the next phrase to be translated, or both variants could be used.

For example when translating from Arabic to English, the subject and verb are swapped from verb-subject-object to subject-verb-object. When translating the subject in an Arabic sentence, the decoder has to decide to leave a gap containing the verb which is then translated later. This decision is probably easier when looking at the subject than when looking at words before that.

6.6.1 Rest Cost Estimation

When doing reordering in beam-search for phrase-based machine translation, we have to pay close attention to rest cost estimation or future cost estimation. One reason for this is that every distortion move in the decoding implies an additional move that has to be done in the future to return to the monotone path. In all our experiments, we used rest cost estimates based on the conventional, distance-based distortion model as described in [Zens 08].

6.7 Reordering Results

This section describes the results from experiments using the reordering models described in this chapter. Starting from the baseline, distance-based reordering model, the MSD phrase orientation model clearly improves translation quality on Arabic-English (Table 6.1) and Chinese-English (Table 6.2). While the orientation models learned with forced alignment are better than using no orientation model when looking at translation quality, they do not reach the performance of the orientation models learned from word alignments. None of the results shows significant improvements over the baseline using bootstrap sampling at a level of $p = 0.1$.

For the learned distance-based reordering model, we see a similar picture. While the translation quality as measured in Bleu or Ter improves slightly over the heuristic distance-based model, the results presented in Table 6.3, the increase in quality is hardly

Table 6.1: Comparison of the reordering models learned from phrase alignment for the Arabic-English 300k NIST dataset.

Setup	NIST'06		NIST'08		NIST'09	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	42.7	50.2	40.3	52.3	42.7	49.2
MSD orientation	44.0	49.2	41.6	51.4	44.4	48.0
learned MSD orientation	43.0	50.4	40.4	51.9	43.2	48.9
learned (phrases+orientation)	43.6	49.5	41.1	51.5	43.9	48.4

Table 6.2: Comparison of the MSD orientation models learned from phrase alignment for the Chinese-English FBIS dataset.

Setup	NIST'06		NIST'08	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	21.5	69.8	16.4	73.7
MSD orientation	22.8	69.7	17.5	73.6
learned MSD orientation	22.3	68.9	16.9	73.2
learned (phrases+orientation)	22.2	69.2	17.0	73.2

worth the additional effort. Significance tests using bootstrap sampling do not show significant improvements for p=0.1.

Table 6.3: Comparison of the distance-based reordering models learned from phrase alignment for the Arabic-English 300k NIST dataset.

Setup	NIST'06		NIST'08		NIST'09	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	42.7	50.2	40.3	52.3	42.7	49.2
learned distance	42.8	50.2	40.3	52.3	42.8	49.2

7 Extended Lexicon Models

7.1 Introduction

Lexical dependencies modeled in standard phrase-based statistical machine translation are rather local. Even though the decision about the best translation is made on the sentence level, phrase models and word lexicons usually do not take context beyond the phrase boundaries into account. This is especially problematic since the average source phrase length used during decoding is small. When translating Chinese to English, it is typically close to only two words.

The target language model is the only model that uses lexical context across phrase boundaries. It is a very important feature in the log-linear setup of today's phrase-based decoders. However, its context is typically limited to three to six words and it only considers the target words produced in translations. In this chapter, we present a model that explicitly takes advantage of sentence-level dependencies in the source sentence and makes predictions for the target words. This is an important aspect when translating from languages like German and Chinese where long-distance dependencies are common. In Chinese, for example, tenses are often encoded by indicator words and particles whose position is relatively free in the sentence. In German, prefixes of verbs can be moved over long distances towards the end of the sentence. For written Arabic, the diacritics are often not written. This creates ambiguities for many words that can only be resolved by larger contexts.

Figure 7.1 shows the context dependency for translation from Chinese to English in a real-world example. The English sentence is the translation produced by the baseline phrase-based translation system. The word "恢复" marked in orange can be a noun meaning "recovery", "restoration", "retrieval", etc.or a verb meaning "recover", "restore", "resume", etc.Whether the word is to be read as a verb or a noun is indicated by the context. In this example, we have the word "正在", marked in red in Figure 7.1 which indicates a progressive form. However, "正在", is outside the phrase context used by the translation model, resulting in an incorrect translation of "恢 复" as a noun. The target language model context marked in green in Figure 7.1 is too short to take the time information "at present" from the beginning of the sentence into account.

Our model can be categorized as extensions of standard word lexicons: a maximum entropy word lexicon that uses global, i.e. sentence-level source information to predict the target words using a statistical classifier, allowing for a more fine-grained lexical choice of the target words. The log-linear framework of the discriminative word lexicon offers a high degree of flexibility in the selection of features. Other sources of information such as syntax or morphology can be easily integrated.

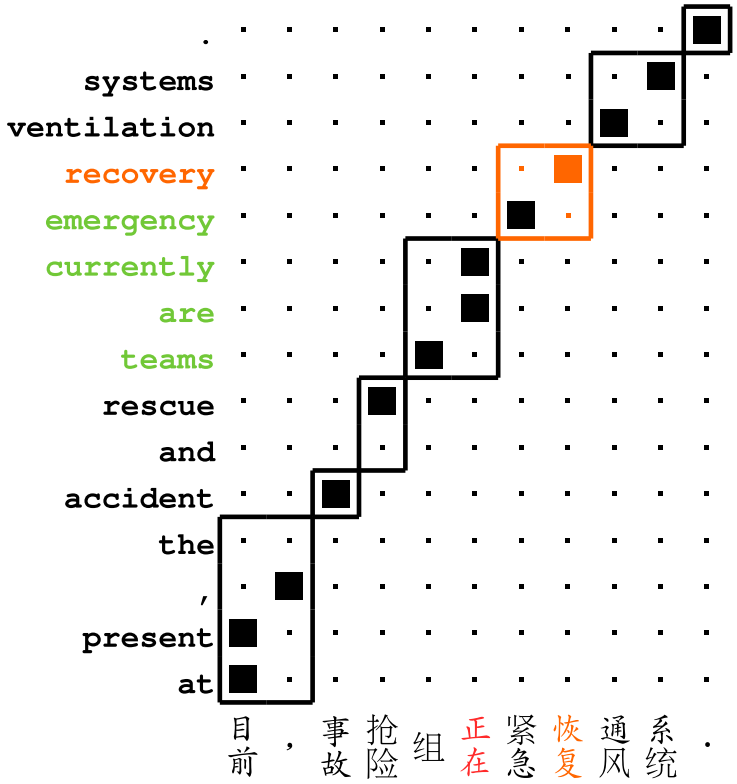


Figure 7.1: Visualization of non-local context dependencies when translating from Chinese to English. The English sentence is the translation given by the baseline phrase-based translation system. The lexical context used in the translation of "恢复" is marked in orange. The target language model context is marked in green. The out-of-context word "恢复" that is needed to determine the correct tense for the translation of "恢复" is marked in red. As a result, the system incorrectly translates "恢复" as a noun, instead of a progressive tense verb.

As will be shown later, the experiments indicate that these models help to ensure translation of content words that are often omitted by the baseline system. This is a common problem in Chinese-English translation.

In practice, the models can get extremely large. The standard approach for classifiers of this type would be to learn parameters for all combinations of the target words and source features. We will refer to this formulation as the full model. With a typical vocabulary of 100k words (10^5) on source and target side, the resulting models would be unmanageable in terms of size with 10^{10} or 10 billion parameters. We will focus our attention here on the efficient approximation and representation of the models in training and in translation.

The structure of this chapter is as follows: In Section 4.2, we will address related work and briefly pin down how our models differentiate from previous work. Section 7.4 will describe the discriminative lexical selection model and the features used in more detail. Section 7.5 will explain the sparse representation of our models, and Sections 7.6 and 7.7 will describe the training procedures and show how the models are integrated into the decoder.

7.2 Related Work

Several word lexicon models have emerged in the context of multilingual natural language processing. Some of them were used as a machine translation system or as a part of such a system. There are three major types of models: Heuristic models as in [Melamed 00], generative models as the IBM models [Brown & Della Pietra$^+$ 93] and discriminative models [Garcia-Varea & Och$^+$ 01, Bangalore & Haffner$^+$ 06, Mauser & Hasan$^+$ 09]

Similar to this work, the authors of [Garcia-Varea & Och$^+$ 01] try to incorporate a maximum entropy lexicon model into an SMT system. They use the words and word classes from the local context as features and show improvements with n-best rescoring.

The models in this chapter are also related to word sense disambiguation. For example, [Chan & Ng$^+$ 07] trained a discriminative model for word sense disambiguation using local but also across-sentence unigram collocations of words in order to refine phrase pair selection dynamically by incorporating scores from the word sense disambiguation classifier. They showed improvements in translation quality in a hierarchical phrase-based translation system. Another words sense disambiguation approach incorporating context-dependent phrasal translation lexicons is given in [Carpuat & Wu 07] and has been evaluated on several translation tasks. Instead of disambiguating phrase senses as in [Carpuat & Wu 07], we model word selection independently of the phrases used in the MT models. Finally, the training is done in a different way as will be presented in Sections 7.6.

Recently, full translation models using discriminative training criteria have emerged as well. They are designed to generate a translation for a given source sentence and not only score or disambiguate hypotheses given by a translation system. In [Ittycheriah & Roukos 07], the model can predict 1-to-many translations with gaps and uses words, morphologic, and syntactic features from the local context.

In [Venkatapathy & Bangalore 07] three different models are proposed: The first one is a global lexical selection model which includes all words of the source sentence as features, regardless of their position. Using these features, the system predicts the words that should be included in the target sentence. Sentence structure is then reconstructed using permutations of the generated bag of target words.

We also compare our models to the Trigger-based lexicon models [Hasan & Ganitkevitch$^+$ 08], [Hasan & Ney 09], where a generative lexicon model with extended context is used. They model the probability of a target word e by conditioning on two source words f and f' and achieve improvements on a large-scale Chinese-English translation task. This chapter is closely related to [Mauser & Hasan$^+$ 09], which will be described in more detail in the following sections.

7.3 Publications and Team Work

This chapter is based on [Mauser & Hasan$^+$ 09], a joint paper with Saša Hasan and Hermann Ney.

The author of this thesis conceived the idea of the discriminative lexicon model for phrase-

based translation as described in Section 7.4.

Similar approaches have been used in the literature. See Section 7.2 for a review of the related work.

The implementation of the training (see Section 7.6) and the integration into the statistical machine translation system (see Section 7.7) was done by the author of this thesis. While initially a third party tool was used to train models, a new training was implemented for this thesis.

Experiments presented in [Mauser & Hasan$^+$ 09] involving the discriminative lexicon model were designed and performed by the author of this theses. The design of the experiments was done in cooperation with Saša Hasan to ensure comparability of experimental results for different models.

All other experiments were designed and performed solely by the author of this thesis.

The analysis of the experimental results presented in [Mauser & Hasan$^+$ 09] was done in collaboration with Saša Hasan. The author of this thesis performed the detailed analysis of the discriminative lexicon model and contributed to the analysis and comparison of the joint results.

7.4 Discriminative Lexicon Model

In this section, we present the extended lexicon models, how they are trained and integrated into the phrase-based decoder.

Discriminative models have been shown to outperform generative models on many natural language processing tasks. For machine translation, however, the adaptation of these methods is difficult due to the large space of possible translations and the size of the training data that has to be used to achieve significant improvements.

In this section, we propose a discriminative word lexicon model that follows [Bangalore & Haffner$^+$ 07] and integrate it into the standard phrase-based machine translation approach.

The core of our model is a classifier that predicts target words, given the words of the source sentence. The structure of the source as well as the target sentence is neglected in this model. We do not make any assumtions about the location of the words in the sentence. This is useful in many cases, as words and morphology can depend on information given at other positions in the sentence. An example would be the character 了 in Chinese that indicates a completed or past action and does not need to appear close to the verb.

We model the probability of the set of target words in a sentence, $\mathbf{e}$, given the source sentence f_1^J. For each word in the target vocabulary, we can calculate a probability for being or not being included in the set. We use a helper function $\delta(e, \mathbf{e})$ that transforms set membership of $e \in \mathbf{e}$ into a binary number to simplify the notation:

$$\delta(e, \mathbf{e}) = \begin{cases} 1 & \text{if } e \in \mathbf{e} \\ 0 & \text{if } e \notin \mathbf{e} \end{cases} \tag{7.1}$$

The probability of the whole set then is the product over the entire target vocabulary $\mathbf{V_E}$:

$$P(\mathbf{e}|f_1^J) = \prod_{e \in \mathbf{V_E}} P(\delta(e,\mathbf{e})|e, f_1^J) = \prod_{e \in \mathbf{e}} \underbrace{P(\delta(e,\mathbf{e})|e, f_1^J)}_{\delta(e,\mathbf{e})=1} \cdot \prod_{e \in \mathbf{V_E} \setminus \mathbf{e}} \underbrace{P(\delta(e,\mathbf{e})|e, f_1^J)}_{\delta(e,\mathbf{e})=0} \tag{7.2}$$

We model the individual factors $p(\delta(e,\mathbf{e})|e, f_1^J)$ of the probability in (7.2) as a log-linear model using the source words from f_1^J as binary features $\phi(f, f_1^J)$ and feature weights $\lambda_{f,\cdot}$:

$$P(\delta(e,\mathbf{e})|e, f_1^J) = \frac{\exp\left(\sum_{j=1}^J \lambda_{f_j,\delta(e,\mathbf{e})}\right)}{\sum_{\delta' \in \{0,1\}} \exp\left(\sum_{j'=1}^J \lambda_{f_{j'},\delta'}\right)} \tag{7.3}$$

Modeling the lexicon on sets and not on sequences has two benefits. Phrase-based statistical machine translation, along with n-gram language models, is strong at predicting sequences but only use information from a local context. By using global features and predicting words in a non-local fashion, we can augment the strong local decisions from the phrase-based systems with sentence-level information.

For practical reasons, translating to a set simplifies the parallelization of the training procedure. The classifiers for the target words can be trained separately as explained in the following section.

7.5 Sparse Model Representation

Current machine translation tasks have a vocabulary size in the order of 10^5 to 10^7. For example the "10^9 French-English" corpus provided for the Workshop for Machine Translation[1] has a vocabulary of over two million words. Training full discriminative models in a standard fashion using only simple parameters, such as input words usually leads to a model size that is quadratic in the vocabulary, ranging from 10^{10} to 10^{14} parameters. While this might still be manageable for high-end computer systems today, it is certainly inconvenient. In this section, we are looking at various ways to reduce the model size.

When looking at the statistics of parallel data, especially the cooccurrence of a specific source word f and a target word e, a high degree of sparsity can be observed. In the German-English WMT 2010 training data, every target word has on average only cooccurred with 262 source words. The majority of all source words and thus also features that we obtain from these source words has only been observed with a few target words or classes and vice versa.

In Figure 7.2, the target vocabulary is shown on the vertical axis and the source vocabulary on the horizontal axis. Each dot in the image represents a cooccurrence of a source-target word pair in the training data. Target and source words are sorted by their occurrence in the corpus. Frequent words are expected to occur earlier in the corpus than infrequent

[1] `http://www.statmt.org/wmt10/training-giga-fren.tar`

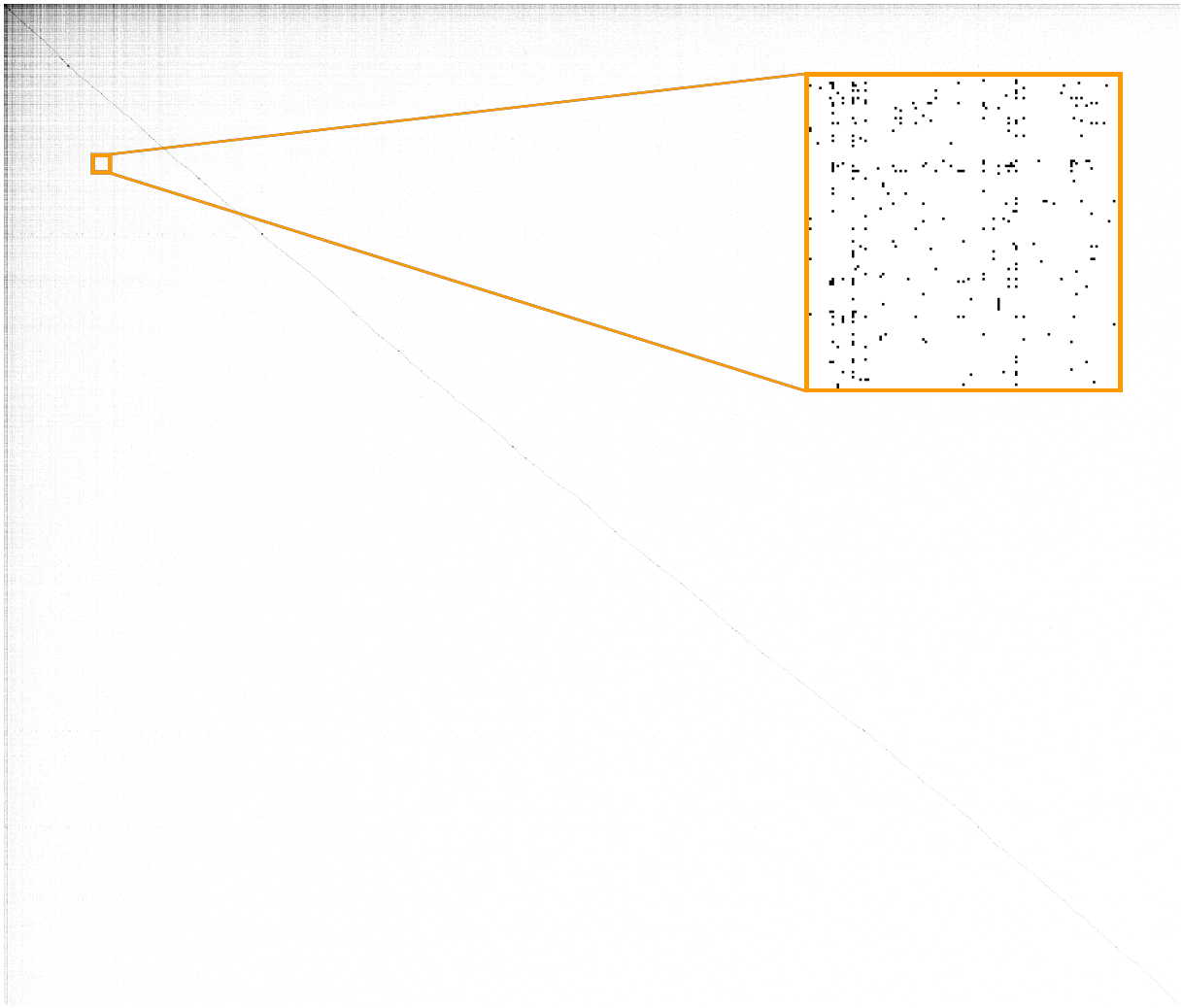


Figure 7.2: Sparseness of word cooccurrence for the IWSLT Chinese-English training data. The vertical axis lists the English vocabulary in order of appearence in the corpus. The horizontal axis lists the Chinese words in the order of appearence. Every dot in the image is a word-cooccurrence between a source and a target word.

words. Few words of the source and the target occur with many words on the other side. In order to occur with many other words, these words have to be frequent and thus show up early in the corpus, leading to the dense parts in the top and left of the figure. What can also be seen in Figure 7.2, is the sparseness of the observations that we use to train our model. Each of the pixels in the plot corresponds to one parameter that is learned in a standard discriminative model. White pixels correspond to parameters that are never directly observed on the data. In order to verify, if we need to train these parameters, we performed an experiment to compare the full model with the sparse model.

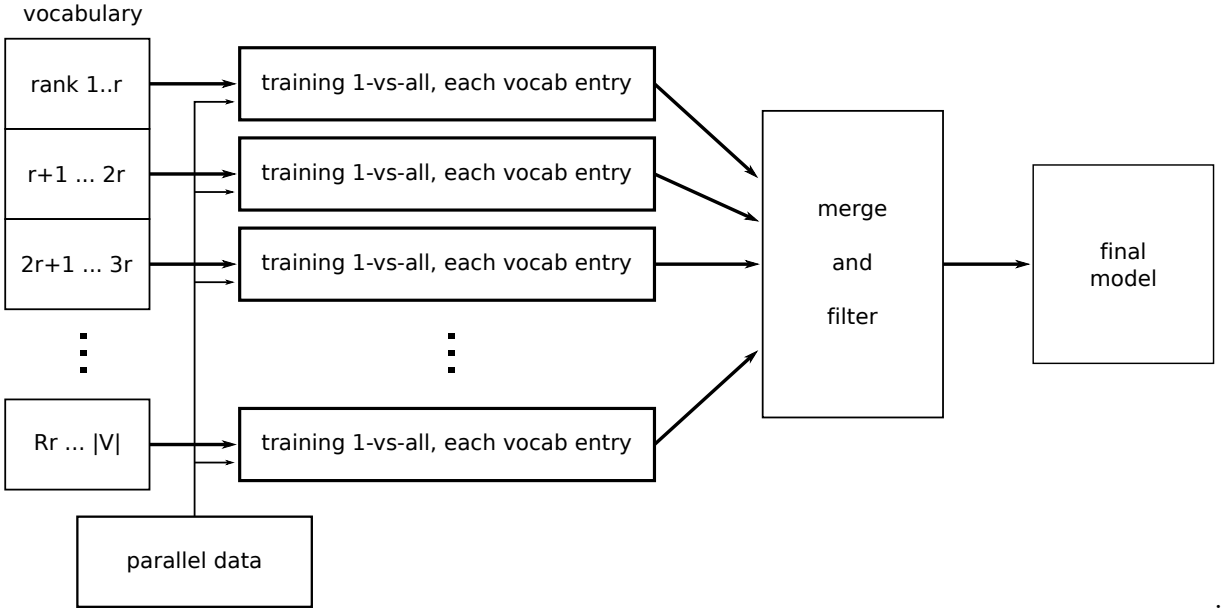


Figure 7.3: Parallelization of training for the discriminative word lexicon model. The vocabulary is split in R groups of r entries that are processed in parallel. The partial results are then merged and filtered into the final model.

7.6 Training

Common classification tasks have a relatively small number of classes. In our case, the number of classes is the size of the target vocabulary. For large translation tasks, this is in the range of a 100k classes. It is far from what conventional out-of-the-box classifiers can handle.

The discriminative word lexicon model has the convenient property that we can train a separate model for each target word making parallelization straight forward. Discussions about possible classifiers and the choice of regularization can be found in [Bangalore & Haffner$^+$ 07].

We implemented the training using a selectable sparse/dense vector and matrix implementation to achieve optimal run-time and memory utilization. Parameter updates are estimated using RPROP [Riedmiller & Braun 93] with L2 regularization. The parallelization is used on two levels: first we separately train classifiers for each target word. Target words are bundles in sets of 100 to 10 000, depending on the corpus size and feature set. The training of each individual classifier is parallelized using multiple threads on the same machine. The strategy is depicted in Figure 7.3. Then, the individual models parameters are merged together into one file.

7.7 Decoding

In search, we compute the model probabilities as an additional model in the log-linear model combination of the phrase-based translation approach. To reduce the memory footprint and startup time of the decoding process, we reduced the number of parameters by keeping only large values $\lambda_{f,e}$. The reasoning behind this is that smaller values tend to have a smaller effect on the overall probability. In experiments we determined that we could safely reduce the size of the final model by a factor of 10 without losing predictive power. When scoring the hypothesis proposed by the phrase-based system, we see the translation hypothesis as the set of target words that are predicted. Words from the target vocabulary which are not included in the hypothesis are not part of the set. During the search process, however, we also have to score incomplete hypotheses, where we do not know which words will not be included. This problem is circumvented by rewriting Equation 7.2 as

$$P(\mathbf{e}|f_1^J) = \prod_{e \in \mathbf{V_E}} P(0|e, f_1^J) \cdot \prod_{e \in \mathbf{e}} \frac{P(1|e, f_1^J)}{P(0|e, f_1^J)}.$$

The first product is constant given a source sentence and therefore does not affect the search. Using the model from Equation 7.3, we can further simplify the computation and compute the model score entirely in log-space which is numerically stable even for large vocabularies.

In comparison with the translation model from [Bangalore & Haffner$^+$ 07] where a threshold on the probability is used to determine which words are included in the target sentence, our approach relies on the phrase model to generate translation candidates. This has several advantages: the length of the translation is determined by the phrase model. In contrast to [Bangalore & Haffner$^+$ 07] where repeated target words are treated as distinct classes, we do not explicitly model words occurring multiple times in the translation.

The main advantage of the integration being done here is that the phrase model and the discriminative word lexicon model are complementary in the way they model the translation. While the phrase model is good in predicting translations in a local context, the discriminative word lexicon model is able to predict global aspects of the sentence, like tense or vocabulary changes in questions. While the phrase model is closely tied to the structure of word and phrase alignments, the discriminative word lexicon model completely disregards the structure in the source and target sentences.

7.8 Results for Extended Lexicon Models

7.8.1 Experimental Results

In this section we evaluate the lexicon model described in this chapter. As our main commaprison, we use the Triplet lexicon model presented in [Mauser & Hasan$^+$ 09].

We evaluate both models separately and in combination on the GALE Chinese-English task for newswire and web text translation and additionally on the official NIST 2008 task for both Chinese-English and Arabic-English. The baseline system was built using a

Table 7.1: Results on the GALE Chinese-English test set for the newswire and web text setting.

GALE	newswire		web text	
test08	BLEU[%]	TER[%]	BLEU[%]	TER[%]
Baseline	32.3	59.38	25.3	64.40
DWL	33.1	58.90	26.2	63.75
Triplet	32.9	58.59	26.2	64.20
DWL+Triplet	33.3	58.23	26.3	63.87

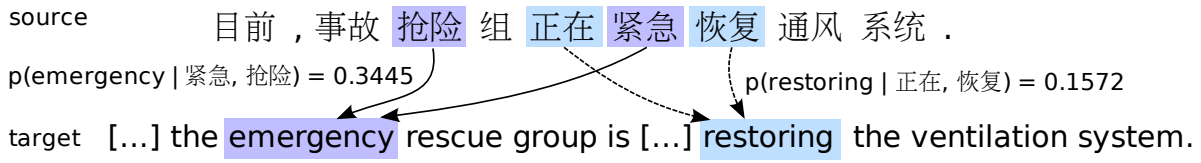


Figure 7.4: Triggering effect for the example sentence using the Triplet lexicon model. The Chinese source sentence is shown in its segmented form. Two triplets are highlighted that have high probability and favor the target words *emergency* and *restoring*.

state-of-the art phrase-based machine translation system as described in Chapter 3. We use the standard set of models with phrase translation probabilities for source-to-target and target-to-source direction, smoothing with lexical weights, a word and phrase penalty, distance-based and lexicalized reordering, and a 5-gram (GALE) or 6-gram (NIST) target language model.

For the translation system, we used training data provided by the Linguistic Data Consortium (LDC) consisting of 10M parallel Chinese-English sentence pairs of various domains for GALE (cf. Table 9.7) and less amounts of data for the NIST systems (cf. Tables 9.4 and 9.6). The lexicon model was integrated into the decoder.

For the GALE development and test set, we separated the newswire and web text parts and did separate parameter tuning for each genre using the corresponding development set which consists of 485 sentences for newswire texts and 533 sentences of web text. The test set has 480 sentences for newswire and 490 sentences for web text. For NIST, we tuned on the official 2006 eval set and used the 2008 evaluation set as a blind test set.

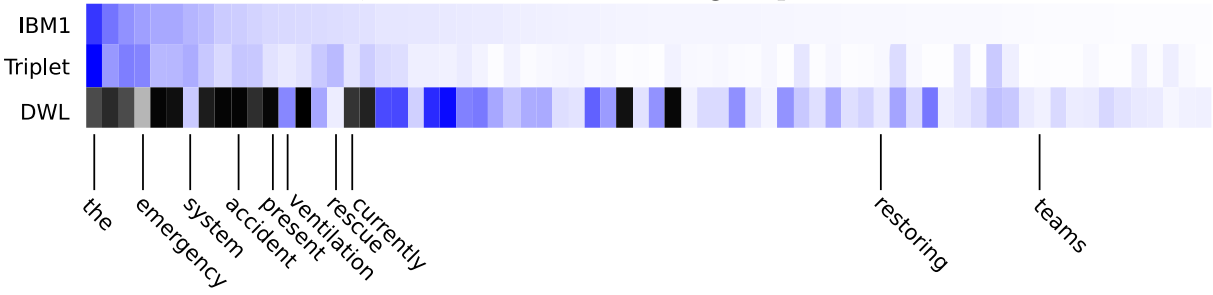
The translation results on the two GALE test sets are shown in Table 7.1. Both the triplet lexicon and the discriminative word lexicon can individually improve the baseline by approximately +0.6–0.9% BLEU and -0.5–0.8% TER. For the combination of both lexicons on the newswire setting, we observe only a slight improvement on BLEU and an additional reduction in TER, arriving at +1% BLEU and -1.2% TER. For web text, the findings are similar: the combination of the trigger-based and discriminative lexicons yields +1% BLEU and decreases TER by -0.5%.

We compared these results against an inverse IBM1 but the results were inconclusive

Table 7.2: Results on the test sets for the NIST 2008 Chinese-English and Arabic-English task.

NIST	Chinese-English		Arabic-English	
nist08	BLEU[%]	TER[%]	BLEU[%]	TER[%]
Baseline	26.8	65.11	42.0	50.55
DWL	27.6	63.56	42.4	50.01
Triplet	27.7	63.60	42.9	49.76
DWL+Triplet	27.9	63.56	43.0	49.15

Figure 7.5: Ranking of words for the example sentence for IBM1, Triplet and DWL model. Ranks are sorted at IBM1, darker colors indicate higher probabilities within the model.



which is consistent with the results presented in [Och & Gildea$^+$ 04] where no improvements were achieved using $p(e|f)$. In our case, inverse IBM1 improves results by 0.2–0.4% BLEU on the development set but does not show the same trend on the test sets. Furthermore, combining IBM1 with the discriminative word lexicon or Triplets, often degraded the translation results, for example only 32.8% BLEU was achieved on newswire for a combination of the IBM1, discriminative word lexicon and Triplet model. In contrast, combinations of the DWL and Triplet model did not degrade performance and could benefit from each other.

We tested the presented lexicon models also on another large-scale system that is NIST, for Chinese-English and Arabic-English. The results for this tasks are in line with the previous findings as can be seen in Table 7.2. The overall improvements for this language pair are +1% BLEU and -1.4% TER. In contrast to the GALE Chinese-English task, the Triplet lexicon model for the Arabic-English language pair performs slightly better than the discriminative word lexicon.

7.8.2 Discussion

Automatic evaluation measures indicate that it is helpful to integrate the extended lexicon models into the translation system. In this section, we will analyze some more details of the models and take a look at the lexical choices they make and what differentiates them from the baseline models. In Table 7.3, we selected an example sentence from the GALE newswire test set and show the translation outputs produced by our system under

Table 7.3: Translation example from the GALE newswire test set, comparing the baseline and the presented lexicon models given a reference translation.

Source	目前 , 事故 抢险 组 正在 紧急 恢复 通风 系统 .
Baseline	at present, the accident and rescue teams are currently emergency recovery ventilation systems.
DWL	at present, the emergency rescue teams are currently restoring the ventilation system.
Triplet	at present, the emergency rescue group is in the process of restoring the ventilation system.
DWL +Triplet	at present, the accident emergency rescue teams are currently restoring the ventilation system.
Reference	right now, the accident emergency rescue team is making emergency repair on the ventilation system.

Table 7.4: The top 10 content words predicted by each model for the newswire example sentence. Ranks for the related IBM1 are given as subscripts for the Triplet model.

DWL		Triplet	
emergency	0.894	emergency_{1}	0.048
currently	0.330	system_{2}	0.032
current	0.175	rescue_{8}	0.027
emergencies	0.133	accident_{3}	0.022
present	0.133	ventilation_{7}	0.021
accident	0.119	work_{33}	0.021
recovery	0.053	present_{5}	0.011
group	0.046	currently_{9}	0.010
dealing	0.042	rush_{60}	0.010
ventilation	0.034	restoration_{31}	0.009

different conditions. The baseline does not produce the present participle of the verb *restore* which makes the sentence somewhat hard to understand. Both, the discriminative and the trigger-based lexicon approaches, are capable of generating the correct use of *restoring*. Figure 7.4 gives an example of how discontinuous triggers affect the word choice in the translation. Two cases are depicted where high probabilities of triplets including *emergency* and *restoring* on the target side influence the overall hypothesis selection. The non-local modeling advantages of our model can be observed as well: the source sentence features do not need to be located next to each other or within a given phrase pair. They move across the whole source sentence, thus allowing for capturing of long-range dependencies.

Table 7.4 shows the 10 hightest ranked content words that are predicted by the two models, discriminative word lexicon and triplet lexicon model. IBM1 ranks are indicated by subscripts in the column of the Triplet model. Although the Triplet model is similar to IBM1, we observe differences in the word lists. Comparing this to the visualization of the probability distribution for the example sentence, cf. Figure 7.5, we argue that, although the IBM1 and Triplet distributions look similar, the Triplet model is sharper

Table 7.5: Translation example from the GALE web text test set. In this case, the baseline has a better TER but we can observe a corrected content word (*remark*) for the extended lexicon models.

Source	我 听 了 莹 的话 , 乐 得 哈哈 大笑 .
Baseline	i have listened to anna, happy and laugh.
DWL	i have listened to the remarks, happy and laugh.
Triplet	i have listened to the music, a roar of laughter.
DWL +Triplet	i have listened to the remarks, happy and laugh.
Reference	hearing ying's remark, i laughed aloud happily.

Table 7.6: Translation example from the NIST Arabic-English test set.

Source	و كانت بعض الصحف السعودية قد نشرت عددا من الحالات التي تعرضت ل السجن دون مبرر و كذلك بعض حوادث القتل داخل الشقق و غير ها .
Baseline	some saudi newspapers have published a number of cases that had been subjected to imprisonment without justification, as well as some killings inside the flats and others.
DWL +Triplet	some of the saudi newspapers have published a number of cases which were subjected to imprisonment without justification, as well as some incidents of murder in apartments and others.
Reference	some saudi newspapers have published a number of cases in which people were unjustifiably imprisoned, as well as some incidents of murder in apartments and elsewhere.

and favors words such as the ones in Table 7.4, resulting in different word choice in the translation process. In contrast, the DWL approach gives more distinct probabilities, selecting content words that are not chosen by the other models.

Table 7.5 shows an example from the web text test set. Here, the baseline hypothesis contains an incorrect word, *anna*, which might have been mistaken for the name *ying*. Interestingly, the hypotheses of the DWL lexicon and the combination of DWL and Triplet contain the correct content word *remarks*. The Triplet model makes an error by selecting *music*, an artifact that might come from words that co-occur frequently with the corresponding Chinese verb *to listen*, i.e. 听 , in the data. Although the TER score of the baseline is better than the one for the alternative models for this particular example, we still think that the observed effects show how our models help producing different hypotheses that might lead to subjectively better translations.

An Arabic-English translation example is shown in Table 7.6. Here, the term *murder in apartments* was chosen over the baseline's *killings inside the flats* which has a better match with the reference translation.

7.8.3 Conclusion

In this chapter, we have presented a log-linear model that uses global source sentence context and is capable of predicting context-specific target words. The models have been directly integrated into the decoder and have shown to improve the translation quality of a state-of-the-art phrase-based machine translation system. The model uses sentence-level features to predict if a word from the target vocabulary should be included in the translation or not.

Overall improvements are up to +1% in BLEU and -1.4% in TER. Compared to the inverse IBM1 which did not yield consistent improvements, the presented models are a valuable additional feature in a phrase-based statistical machine translation system.

8 Linear-Chain Conditional Random Fields for Statistical Machine Translation

Log-linear models offer a convenient way to combine many features that are helpful for translation into a consistent modelling framework. The number of features that can be used in this approach goes far beyond the typical 15-30 models used in conventional phrase-based systems. While it has shown to be beneficial to translation quality to include more and more features [Chiang & Knight$^+$ 08], there have always been difficulties when including millions or billions of features [Liang & Buchard-Côté$^+$ 06], [Blunsom & Cohn$^+$ 08], [Lavergne & Allauzen$^+$ 11]. While some success has been reported for word alignment problems [Blunsom & Cohn 06], [Dyer & Clark$^+$ 11], so far, only [Ittycheriah & Roukos 07] seems to have been successful in outperforming a strong phrase-based baseline system with a maximum entropy translation system.

The earliest attempts in this direction were done very early in the history of statistical machine translation by [Berger & Brown$^+$ 94] and [Foster 00] who used locally-normalized models which are sometimes referred to as maximum entropy Markov models. Later, in other natural language processing tasks such as tagging, the locally normalized models were consistently outperformed by Conditional Random Field models (CRFs) [Lafferty & McCallum$^+$ 01]. Building on the improvements achieved with our maximum entropy lexicon models from Chapter 7, we now extend our modelling to a full globally normalized phrase-based maximum entropy model for statistical machine translation.

This chapter is structured as follows. First, we revisit linear-chain CRFs in Section 8.2. This model is extended in Section 8.3 to include the phrase-based translation model. Sections 8.4 and 8.5 describe the training and inference with the models.

The features used are described in Section 8.6 and the evaluation of the approach is done in Section 8.7. We conclude this chapter with a discussion of the results in Section 8.8.

8.1 Publications and Team Work

The novel work presented in this chapter was not puplished prior to this thesis. Unless specifically marked, ideas, concetps, implementation, experimentation, evaluation, and analysis has been performed by the author of this thesis.

A novel model is presented in Section 8.3, with some additional explanation in Sections 8.4, 8.5, and 8.6.

While some of the ideas had been explored before in the literature (see the beginning of

this chapter and Section 8.2), the specific modelling idea of learning the model parameters in the way described in this chapter has not been presented.

All implementation of training and the integration into the statistical machine translation system were done by the author of this thesis. The same holds for the design, execution, verification and analysis of all the experimental evaluation.

8.2 CRFs

In this section we formally introduce CRFs with a focus on linear-chain CRFs that are commonly used for natural language processing tasks and describe efficient training methods. CRFs are typically used for natural language processing tasks, where a sequence of symbols as the output of the procedure is modelled. A general property of these sequence labelling tasks is that the input and output sequences have the same length.

Popularized by [Lafferty & McCallum$^+$ 01], CRFs have been successfully applied to a number of tasks in natural language processing and beyond.

Named Entity Recognition (NER) The task of named entity recognition is to label all names of persons, organizations, locations and others in text. In [McCallum & Li 03] CRFs are used to incorporate external information gathered from the web.

Parts-of-speech tagging Each word in the input sequences is labelled with its corresponding parts-of-speech tag. The set of parts-of-speech tags is limited in size, ranging from 40 to about 200 tags, depending on granularity. [Lafferty & McCallum$^+$ 01] report improvements in tagging error rate over both HMMs and MEMMs.

Concept Tagging Being a subtask of natural language understanding, the task of concept tagging is to extract a sequence of concepts from an input sentence. Concepts are semantic units in a sentence and are often used to model information items in dialog systems. Concepts can span more than one word, this concept tagging includes an implicit segmentation. This discrepancy between the input and output sequence length is solved by introducing "begin" and "continue" tags for each concept. The first word is always labeled with the "begin" tag. If the next word belongs to the same concept, the "continue" label is used. [Hahn & Lehnen$^+$ 08] compares various methods for concept tagging. The authors find, that CRFs leads to better labelling quality compared to other models. Experiments are carried out on the MEDIA [Bonneau-Maynard & Rosset$^+$ 05] corpus which is annotated with 74 different concept tags.

Non-NLP applications Beyond natural language processing, CRFs have been successfully applied to problems such as image labelling [He & Zemel$^+$ 04], speech recognition [Zweig & Nguyen 09], and intrusion detection [Gupta & Nath$^+$ 10] to name just a few.

Traditionally, HMMs have been popular for these tasks but as generative models, they have the disadvantage of modelling the joint probability of the input and output sequences and thus also need to model the (known) input sequence.

8.2.1 From Tagging to Translation

Above, we have given examples, where CRFs have been successfully applied in sequence labelling tasks. While the translation task can be seen as similar to the sequence labelling task in a sense that we have to label every source input word with its target translation, translation is different from tagging in three important aspects:

- The length of the output sequence may be longer or shorter than the length of the input sequence. While the latter is also true for the concept tagging tasks, longer sequences typically do not occur.
- The order of the words in the translation may be different from the words in the source sequence. Depending on the language pair, the amount of reordering may be large or small.
- The size of the output label set is orders of magnitude larger than in typical labeling tasks. As the run-time complexity of a linear-chain CRF, as used in common sequence labelling tasks, depends at least quadratically on the label vocabulary size. This is a very important factor in the use of CRFs for statistical machine translation.

This last point will be addressed in Section 8.4.2. The first two problems originate from the structure of the translation problem itself. In the common setup for training phrase-based statistical machine translation systems, this is solved by using an automatically obtained word alignment. By giving an alignment between words in the input sequence and words in the output sequence of the training data, specific relations such as 1-to-many or many-to-one word translations are implicitly encoded. The same is true for the reordering of the words.

To develop our CRF model for statistical machine translation, we start with the modelling done for a normal tagging task, assuming that both, the source and target sentences have the same length and have a monotone alignment. For notational consistency with the rest of this thesis, we use f_1^J as the input sentence and e_1^I as the output sequence. Monotonicity and equal length means here that $I = J$ and we also assume that e_i is the label for the source word f_j if $i = j$. In the first equation, we will therefore just use i and I as indices. The conditional log-linear model for the translation probability is then

$$p(e_1^I|f_1^I, I) = \frac{\prod_i \exp\left(\sum_k \lambda_{e_i,k} h_k(e_i, e_{i-1}, f_1^I, i)\right)}{\sum_{\tilde{e}_1^I} \prod_i \exp\left(\sum_k \lambda_{\tilde{e}_i,k} h_k(\tilde{e}_i, \tilde{e}_{i-1}, f_1^I, i)\right)}. \tag{8.1}$$

The feature functions $h_k(e_i, e_{i-1}, f_1^I, i)$ can use a bigram label context on the target side, the current position and the full source sentence. Although the target side context can be extended to longer n-grams, we keep the bigram for notational simplicity. Equation 8.1 assumes a monotonous 1-to-1 alignment between source words f and target words e. If we allow for a more general alignment A and assume that this alignment is given, we can extend this approach to a translation problem, where we merely have to predict the correct words, but not the structure. We require the alignment to be a function of source positions such that

$$A : i \mapsto\subseteq \{1, \ldots, J\} \tag{8.2}$$

$$p(e_1^I|f_1^J, I, J, A) = \frac{\prod_i \exp\left(\sum_k \lambda_{e_i,k} h_k(e_i, e_{i-1}, f_1^J, i, A)\right)}{\sum_{\tilde{e}_1^I} \prod_i \exp\left(\sum_k \lambda_{\tilde{e}_i,k} h_k(\tilde{e}_i, \tilde{e}_{i-1}, f_1^J, i, A)\right)}. \tag{8.3}$$

In Equation 8.3, we have a CRF model that can deal with length differences in input and output sequences, and with one-to-many and many-to-one alignments. However, we still rely on an externally provided alignment. As stated in Equation 8.2, we have a set of aligned source words for each target position that is seen as input to be labelled. Thus, we still remain within the sequence labelling framework. As additional information, we can also include which words are within the same phrase as used by a phrase-based translation model. The phrase alignment is generated using the forced alignment procedure described in Section 4.5.

However, this approach restricts the resulting model in several ways. First, the static alignment that we use is typically obtained automatically and can contain errors. The CRF model however takes theses as given and it is not able to change the alignment or consider alternate hypotheses. Second, the integration of alignment alternatives is difficult as that would mean a departure from the plain sequence labelling task.

Therefore, [Dyer & Clark$^+$ 11] introduced a hidden CRF where the word alignment is treated as a hidden variable. The formulation of the translation probability then reads as:

$$p(e_1^I|f_1^J) = \sum_{s_1^K} p(e_1^I, s_1^K|f_1^J) = \frac{\sum_{s_1^K} \prod_{i=1}^{I} \exp\left(\sum_m^M \lambda_{e_i,m} h_m(e_i, e_{i-1}, f_1^J, i, S(i))\right)}{\sum_{\tilde{I}, \tilde{e}_1^{\tilde{I}}, \tilde{s}_1^{\tilde{K}}} \prod_{\tilde{i}=1}^{\tilde{I}} \exp\left(\sum_m^M \lambda_{\tilde{e}_{\tilde{i}},m} h_m(\tilde{e}_{\tilde{i}}, \tilde{e}_{\tilde{i}-1}, f_1^J, \tilde{i}, S(\tilde{i}))\right)}. \tag{8.4}$$

Note that as [Dyer & Clark$^+$ 11], we are not explicitly modelling the target length I here. While this might be useful during full decoding, we will only apply this model to re-score existing phrase lattices and therefore are not primarily concerned about length modelling.

The model itself can now consider alternative alignment hypotheses and score them independently, while still efficiently computing expectations. The flexibility however comes at a price. First, we now have to sum over all possible alignments in the model, making the computation of expectations computationally more demanding. Second, the optimization problem of hidden CRFs is not convex anymore. This means that the initialization of parameters matters.

A similar model has been proposed by [Lehnen & Peter$^+$ 13], where word-level hidden CRFs are successfully used to re-score a translation system.

To reduce the computational complexity resulting from a summation over all alignment paths in Equation 8.4, we will use the phrase alignments generated using the forced alignment procedure described in Section 4.5. The procedure and resulting model is described in the following section.

8.3 Phrase-based hidden CRF

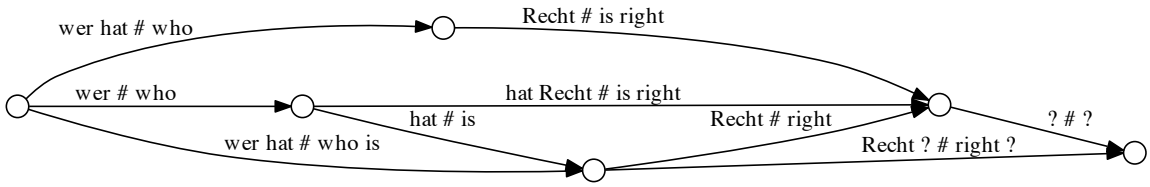


Figure 8.1: Example alignment graph with alternative phrase alignment hypotheses. Edge labels include the original source words to the left of the # symbol and the translation of the phrase to the right of the # symbol. Each arc has a set of model weights that are not shown here.

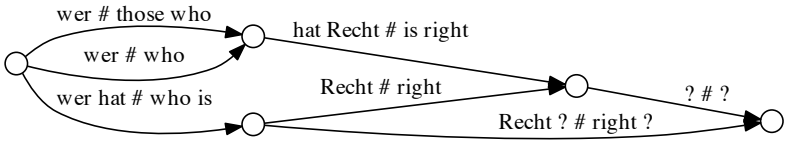


Figure 8.2: Example translation phrase graph with alternative translation hypotheses. Edge labels include the original source words to the left of the # symbol and the translation of the phrase to the right of the # symbol. Each arc has a set of model weights that are not shown here.

Alternate alignments will be represented as alternative paths in a graph. An example phrase alignment graph for a sentence is shown in Figure 8.1. The alignment graph represents the numerator of Equation 8.4 and contains segmentation and reordering hypotheses for the correct translation of the source sentence given in our training data. In order to train the discriminative model, we also need to consider competing hypotheses for computing the denominator in Equation 8.4. These are obtained by doing a regular translation of the source sentence. The alternative translations hypotheses are also encoded in a hypothesis graph. The example is shown in Figure 8.2. To ensure the correctness of the training procedure and to ensure numerical stability, we have to guarantee that the correct translation hypotheses from the numerator are also contained in the denominator. This is accomplished by inserting the numerator graph into the translation graph. This merged graph shown in Figure 8.3 is then used as the denominator.

Graph-based discriminative training has been successfully used in automatic speech recognition [Macherey 10] and for n-gram-based statistical machine translation in [Lavergne & Allauzen$^+$ 11].

We are using phrase alignments for a number of reasons:

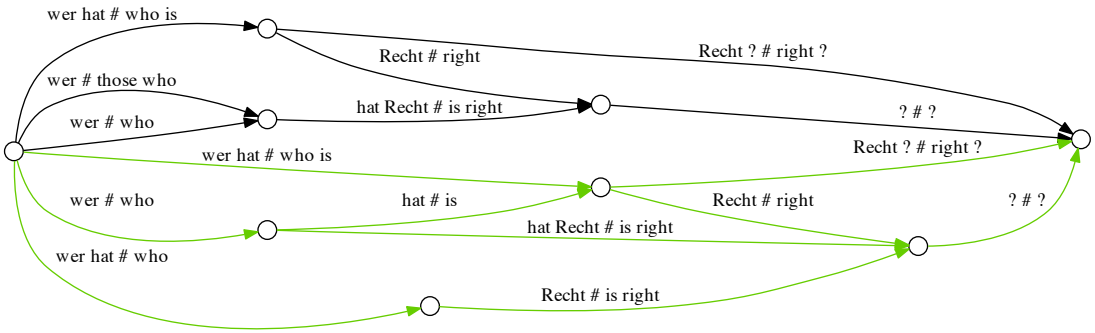


Figure 8.3: Combined alignment and translation graph. The graph was constructed by inserting the alignment graph for Figure 8.1, into the translation graph from Figure 8.2.

- The phrase-based translation system provides a state-of the art baseline to improve upon.
- The new model can incorporate features from the phrase-based models if needed and can rely on more fine-grained features to improve on that.
- We know that the pre-selection of permitted alignments and possible translations provided by the phrase-based system is good enough to allow for improvement. Given sufficiently large translation graphs, the oracle BLEU scores are significantly better than the baseline.
- Run-time and memory requirements can be conveniently scaled using graph pruning.
- As initialization matters for the non-convex hidden CRF model, we can use the phrase-based baseline system as initialization for the hidden CRF model.

Although we use the phrase-based alignments, the translation model still is a word-based CRF model, using the phrasal alignment information in its features. The reason for this is that we do not want translation hypotheses that yield the same target string but differ in the alignment to compete with each other. For example in the WMT 2010 German-English NewsCommentary data, there is on average 4 alignment hypotheses per translation hypothesis. For other, more ambiguous language pairs and tasks, this number may be higher. When using source phrases as our input events and target phrases as our output labels, we would treat these identical translations as different. We are not interested in producing the correct phrase sequence, but in the correct translations.

8.4 Training

In this section, we will describe the inference procedure for our hidden CRF model. Starting with the mathematical formulation and the description of the forward-backward algorithm for training, we then describe the practical implementation issues such as word-graph generation and then comment on our efficient implementation.

For the description of the training we assume that we have a training corpus of N sentence pairs (f_n, e_n). We use the feature function template $h_k(e_i, e_{i-1}, f_1^I)$ that corresponds to a set of feature functions for specific vocabulary entries for the e and f. Each feature function template h_m has weights $\lambda_{e,m}$, we use M templates.

For handling the alignment alternatives we follow [Lehnen & Hahn$^+$ 11a] and sum over the set of correct alignments.

8.4.1 Word Graph Construction

To give the training procedure full access to the search space of the underlying phrase-based translation model, we generate a graph-based representation of the search space as described in [Ueffing & Och$^+$ 02]. Instead of word graphs where each arc in the graph corresponds to one word, we use phrase graphs, as described in Section 4.7. To represent our training data, we have to generate a word graph for each training sentence. Since the unconstrained search space would lead to very large graphs and result in memory and run-time problems, we used pruned graphs. As a consequence, even on the training data, we cannot always ensure that the correct reference translation is contained in the hypotheses. Therefore we ensure reference reachability by inserting the correct hypotheses into the translation word graph.

8.4.2 Efficient Implementation

The problems of large maximum entropy models for machine translation has already been illustrated in Section 7.5. For the hidden CRF implementation of this chapter, we use the same underlying sparse model representation as for the lexicon models in Chapter 6.7. The main factor in reducing resource usage for the training is: we only train parameters that have been seen at least once in the training data. This drastically reduces memory requirements. The idea of the approach has been outlined in [Lehnen & Hahn$^+$ 11b].

Evaluating many feature functions can be computationally costly. Computing time is saved by factoring out the computation of features. Some features require only the source sentence and do not depend on the target sentence or position information. These features can be pre-computed for each sentence and thus have to be retrieved only once. Similarly, features that use only within-phrase context can be pre-computed for every phrase in the current sentence. This way we ensure that feature functions are evaluated a minimum number of times.

For the efficient storage and loading of word graphs, we use a custom file container that holds all graphs for the training or test set. The container allows thread-safe random-access to the individual word graphs. The container format provides an efficient way of accessing a large number of files. This is especially important for file systems with large block-sizes that are inefficient for storing many small files.

The word graphs themselves are stored in a binary format that can be efficiently read into memory. Individual graphs are Gzip-compressed to reduce the storage space requirement for the graphs.

Modern computer architectures have multiple computing cores in one processor with shared main memory. Our implementation uses multiple cores for the computation of the derivative statistics. Parallelization is done at the sentence level. A different thread is used for every training instance that is processed. In order to reduce overhead, we reuse the threads so that each processes a part of the corpus. Models, features and other read-only data structures are shared among threads, so only one instance exists in memory. The derivation statistics, however are kept separately for each thread, to avoid run-time overhead originating from thread access control. The derivation statistics are then merged after each iteration to estimate the models for the next iteration.

8.5 Search

For the search, we rely on the same approach as for the training. We generate word graphs for our test sets and apply the models learned on the training data. In addition to using only the hidden CRF scores for finding the best translation, we can also add the normalized hidden CRF model score to the translation word graph and then incorporate other additional features for minimum error rate training.

8.6 Features

When using phrase translation graphs as the core of our model, two sets of feature functions come to mind. First, we have the mostly real-valued models from the baseline phrase-based statistical machine translation system described in Section 3.2. Additionally, we can define feature functions that could help the translation. In [Chiang & Knight$^+$ 08], [Ittycheriah & Roukos 07] or [Dyer & Clark$^+$ 11], there is a wide variety of examples of feature functions for translation. Additionally, we add helpful features from our lexicon models described in Section 6.7. Most of these additional features are binary and typically very sparse. They are 1 for a few constellations and 0 otherwise. In addition to the word-based feature templates, we can also use the phrase alignment information that we get from the word graphs. For this work, we use the following features:

- **Phrase model score.** The phrase model score is used in both directions $p(\tilde{f}|\tilde{e})$ and $p(\tilde{e}|\tilde{f})$.
- **Within-phrase lexicon models.** We use lexical smoothing based on the within-phrase lexicon models described in both directions.
- **Language model.** The language model score is used as provided by the underlying translation system.
- **Aligned word context.** Stored with the phrase alignment, there is also word alignment information within the phrase. Using this information, we use indicator features for the words within a window around the source word aligned to the current target word at position i, including the aligned word itself.
- **Target word context.** Although the construction of the word graphs enables us

to use target word contexts as long as the language model context, we here restrict ourselves to the previous target word.

- **Source phrase identity.** We use the identity of the source phrase as individual indicator features. This allows us, to use a flexible context in the source side as provided by the phrase table.
- **Words in the source phrase.** All the words in the phrase serve as features similar to the within-phrase lexicon models.
- **Words outside of the source phrase.** Each item from the set of words from the source sentence that are not covered by the current phrase.
- **All words of the source sentence.** Identical to the features of the lexicon models from Section 6.7.
- **Pairs of source words.** These features are similar to the triplet lexicon model from [Hasan & Ganitkevitch$^+$ 08].

8.7 Experimental Evaluation

This section will report the results obtained with the hidden CRF model presented in Section 8.3. We will report two sets of results. First, we will evaluate our model on the Celex 40k grapheme to phoneme conversion task. Then we will evaluate our model on the WMT 2010 translation task using only the in-domain News Commentary corpus as training data.

8.7.1 Grapheme to Phoneme Conversion

Grapheme to phoneme conversion is the task of finding the correct pronunciation for a given written word. In practice, this is done by converting the letters, sometimes referred to by the more general term graphemes, into their phonetic transcription denoted by a sequence of phonemes. Graphemes may have different pronunciations depending on the immediate context or other factors.

The reasons for evaluating on the grapheme to phoneme conversion task are that the vocabulary is small, so that the experiments do not take too many resources, and that we have comparable results with CRFs and other methods on this task. Table 9.8 shows the statistics for the Celex 40k grapheme to phoneme conversion task.

The evaluation of the grapheme to phoneme conversion task is done using two error measures. PER is the phoneme error rate that is computed as the symbol edit distance between the predicted phoneme sequence and reference. SER is the sequence error rate, which is 0 only if the phoneme sequence hypothesis matches exactly the reference and 1 otherwise.

Table 8.1 shows the results of the experiments for the Celex 40k task. The phrase-based baseline system was trained in a standard way using GIZA++ for word alignment, standard phrase extraction and minimum error rate training for PER on Celex DEV. This

system provides a PER of 4.5% on the Celex TEST set. The Graph Oracle line shows that the graphs we produced with our phrase-based statistical machine translation system are large enough for the hidden CRF model to improve over the baseline, by having almost no oracle error. However, if we use only the phrase-based hidden CRF model, we do not quite reach the original error rate of the pure phrase-based model. One potential problem might have been the difference between using the over all alignments in training but only doing the best path search for finding the translation. However, we cannot completely recover by using the sum over all alignments and searching for the best translation. The remaining problem with the predictions produced by the phrase-based hidden CRF model is that they tend to be longer than the ones produced by the phrase-based system. To counteract this effect, we performed an additional minimum error rate training step using the phrase-based hidden CRF model in addition to the regular phrase-based translation models. Using this procedure, we can improve over the phrase-based baseline by about 10% relative in PER and SER.

	Celex DEV		Celex TEST	
Setup	PER	SER	PER	SER
[Jiampojamarn & Cherry$^+$ 10]				10.8
[Lehnen & Hahn$^+$ 11b]	2.7	12.3	2.6	12.3
Phrase-Based Baseline	4.3	19.5	4.5	20.3
Graph Oracle	0.2	0.8	0.1	0.6
phrase-based hidden CRF	8.0	32.2	7.8	31.6
+ sum	7.8	31.3	7.7	30.9
+ MERT	3.9	17.6	4.1	18.2

Table 8.1: Results for the Celex 40k grapheme to phoneme conversion task. Error rates are given in percent.

8.7.2 Statistical Machine Translation

When transitioning from the small vocabulary, monotonic grapheme to phoneme conversion task to the statistical machine translation task, we see an overall similar structure in the results in Table 8.2. The error rates for the statistical machine translation experiments here are the same BLEU and TER as what was used throughout the thesis. The details of these evaluation metrics are described in Section 9.1. For BLEU, which is an accuracy measure, higher numbers mean better results. For TER which is an error measure, lower scores are better.

The phrase-based model by itself achieves a BLEU score of 17.2 on the test data given the data used for the further hidden CRF experiments. While using additional data improves the translation performance, we used a smaller subset of the data to facilitate experiments. The full system score is posted for reference. Due to the nature of the translation task, where we do not typically have the ability to produce all translations needed to exactly reproduce the reference, the graph oracle scores are not quite as good as the first-best translation in the graph.

Similar as for the grapheme to phoneme conversion case presented in Table 8.1, the hidden CRF model by itself performs worse than the baseline phrase-based model. While using the sum increases performance as in the grapheme to phoneme conversion task, the combined model using sum and MERT over all does not improve on top of the performance of the pure phrase-based model. There are multiple potential reasons for that, which we will address in the following section.

	WMT NewsTest08		WMT NewsTest09	
Setup	Bleu	Ter	Bleu	Ter
Full system	21.1	63.4	19.9	63.3
Phrase-Based Baseline	18.4	66.2	17.2	66.3
Graph Oracle	30.8	57.2	25.0	57.5
Phrase-based HCRF	13.6	68.3	13.1	68.3
+ sum	14.1	68.4	13.3	68.3
+ MERT	18.2	66.3	17.0	66.1

Table 8.2: Results for the WMT2010 news commentary task. Error rates are given in percent.

8.8 Discussion

The results shown in Section 8.7 indicate that a phrase-based CRF model can improve translation quality in a scenario with small vocabularies, as in the grapheme to phoneme conversion case. While the quality of the statistical machine translation system does not quite reach that of methods designed specifically for the task, using the hidden CRF model improves over the plain phrase-based model by reducing the Per by 9% relative on the test set.

For the more general statistical machine translation case, the interpretation of the results is more difficult. While the graph oracle scores are not as good as for the grapheme to phoneme conversion task, there is still room for improvements in the graphs. However the models fail to realize this potential. On its own, the hidden CRF performs worse than the phrase-based baseline of both tasks. In the grapheme to phoneme conversion case, making sure that the conditions between training and translation are the same by using the sum and adjusting for the evaluation metric by running MERT. This summation and tuning does not seem to work for the translation case.

Potential reasons for that can be data sparsity, because with the larger vocabulary, we now train many more parameters in the hidden CRF model. An additional factor is the discrepancy between the objective function and the evaluation metric. For grapheme to phoneme conversion, Per and especially Ser are very close to the objective function of the hidden CRF. Bleu and Ter on the other hand are quite different to that. The experiments showed, that the current version of the models lacks explicit length modeling, resulting in too short hypotheses being produced.

Overall the results presented here show that with further research there is a potential of a hidden CRF-style model improving general statistical machine translation.

9 Corpora and Evaluation

9.1 Evaluation Criteria

Evaluation of machine translation output can be done manually and subjectively by humans or automatically. In this work, we will only use automatic evaluation metrics that compare the translation system output to one or more human-generated reference translation. The metrics used in this work have been chosen, because they are generally accepted in the research community and are also used in public evaluations do compare translation systems. Evaluation of translation quality is an area of research on its own [Callison-Burch & Koehn$^+$ 10] and it is beyond the scope of this thesis to discuss pro and counter arguments for the multitude of metrics available. We will describe the used evaluation metrics in the following paragraphs.

BLEU. Proposed by [Papineni & Roukos$^+$ 02], the Bleu criterion measures the precision of n-grams compared to the reference translation. In this work, n-grams of length $m = 1, \ldots, 4$ are used. Bleu is a accuracy measure, higher values indicate a better match with the reference translation. The criterion is normalized between 0 and 1. In this work, we will give all Bleu scores in percent. When multiple references are available, the n-grams of all reference translations are pooled. To avoid a precision bias, where systems that produce very short, probably incomplete hypotheses that have a high precision on the produced n-grams, a brevity penalty is used that penalizes sentences that are shorter than the reference translation. Several methods have been proposed to determine the reference length in the case of multiple references. The original method proposed by [Papineni & Roukos$^+$ 02] and today the most common variant is to choose the reference length that is closest to the hypothesis length. If there are two reference lengths that have an identical distance to the hypothesis length, the minimum of the two is chosen, resulting in a reduced brevity penalty. This variant is often called IBM-Bleu. In the past, especially in the NIST OpenMT evaluations prior to 2008, always the shortest available reference length has been used. This resulted in a preference of shorter hypotheses in these evaluations. In this work, we will use the original IBM-Bleu variant using the minimum nearest reference length.

TER. The Ter measure introduced by [Snover & Dorr$^+$ 06] and initially called "translation edit rate" measures the number of edits that are required to transform the translation hypothesis into the reference translation. Permitted edit operations are insertion, deletion, substitution of single words and the reordering of sequences of words. Reordered sequences have to match the reference exactly, additional substitutions and deletions are not permitted. All edit operations are weighted equally with 1 and the sum of all edit

operations is divided by the reference length to obtain an error rate. The minimum TER is 0, meaning that the translation hypothesis exactly matches the reference translation. There is no upper bound for the error, as the translation hypothesis can be longer than the reference, requiring more edits (deletions) than the reference has words. In the case of multiple references, the reference that produces the minimum number of errors is chosen. The reference length is the average of the length of all reference translations.

9.2 Task Descriptions and Corpus Statistics

We evaluate the methods presented in this thesis on machine translation tasks of different languages and corpora. Not all methods are evaluated on all tasks.

9.2.1 WMT 2008 German-English (German-English)

We conducted our experiments on the German-English data published for the *ACL 2008 Workshop on Statistical Machine Translation* (WMT08). The corpus statistics for the WMT 2008 German-English dataset are shown in Table 9.1

Table 9.1: Corpus statistics for the WMT 2008 German-English dataset.

		German	English
Train	Sentences	1 311 815	
	Running Words	34 398 651	36 090 085
	Vocabulary	336 347	118 112
	Singletons	168 686	47 507
WMT 2008 Dev	Sentences	2 000	
	Running Words	55 118	58 761
	Vocabulary	9 211	6 549
	OOVs (running words)	284 (0.6%)	77
	OOVs (in vocabulary)	279 (3.0%)	76
WMT 2008 Test	Sentences	2 000	
	Running Words	56 635	60 188
	Vocabulary	9 254	6 497
	OOVs (running words)	266 (0.5%)	89
	OOVs (in vocabulary)	264 (2.9%)	89

9.2.2 NIST OpenMT 2009 Constrained Task

The NIST OpenMT 2009 evaluation campaign is a roughly annual effort of the United States National Institute of Standards and Technology (NIST) with the aim of providing a comprehensive benchmark for translation systems. The main focus has been on

the translation directions Arabic and Chinese to English, although other languages and translation directions have been evaluated in the past.

While the training data comes from various sources such as documents released and translated by the United Nations or bilingual parliamentary proceedings and legislative texts from Hong Kong and news, the main focus of the evaluation lies on the translation of news wire texts, blogs and newsgroup posts.

To be able to conduct more experiments, we have selected a small subset that contains only a part of the training data. Additionally, for Chinese-English, we have selected a larger set to verify experimental findings using more data.

If not stated otherwise, all experiments have been separately optimized using 5 lattice-MERT runs, each with 20 random restarts. Evaluation is always done case-sensitive using IBM-Bleu.

9.2.2.1 Chinese-English FBIS

The corpus statistics for the Chinese-English FBIS dataset are shown in Table 9.2.

Table 9.2: Corpus statistics for the Chinese-English FBIS dataset.

		Chinese	English
Train	Sentences	224 216	
	Running Words	6 230 508	8 699 849
	Vocabulary	29 817	53 195
	Singletons	5 490	18 787
NIST'06	Sentences	1 664	
	Running Words	40 689	193 311
	Vocabulary	6 139	9 367
	OOVs (running words)	524 (1.3%)	9 691
	OOVs (in vocabulary)	280 (4.6%)	1 045
NIST'08	Sentences	1 357	
	Running Words	34 463	167 791
	Vocabulary	6 209	9 662
	OOVs (running words)	485 (1.4%)	8 242
	OOVs (in vocabulary)	334 (5.4%)	1 092

9.2.2.2 Chinese-English Medium Training Set

The corpus statistics for the Chinese-English Medium dataset are shown in Table 9.3.

9.2.2.3 Chinese-English Large NIST

The corpus statistics for the Chinese-English Large NIST dataset are shown in Table 9.6

Table 9.3: Corpus statistics for the Chinese-English Medium dataset.

		Chinese	English
Train	Sentences	1 586 525	
	Running Words	39 570 535	43 013 508
	Vocabulary	75 690	192 520
	Singletons	18 182	81 684
NIST'06	Sentences	1 664	
	Running Words	42 941	194 885
	Vocabulary	6 391	9 673
	OOVs (running words)	1 913 (4.5%)	9 454
	OOVs (in vocabulary)	257 (4.0%)	767
NIST'08	Sentences	1 357	
	Running Words	36 114	174 801
	Vocabulary	6 418	9 923
	OOVs (running words)	1 441 (4.0%)	11 367
	OOVs (in vocabulary)	197 (3.1%)	626

Table 9.4: Corpus statistics for the Arabic-English Large NIST dataset.

		Chinese	English
Train:	Sentences	7 278 005	
	Running Words	185 387 568	195 556 203
	Vocabulary	163 442	1 017 325
	Singletons	62 054	721 655
NIST'06	Sentences	1 664	
	Running Words	42 941	193 311
	Vocabulary	6 391	9 367
	OOVs (running words)	1 875 (4.4)	5 776
	OOVs (in voc.)	246 (3.8)	326
NIST'08	Sentences	1 357	
	Running Words	36 114	167 791
	Vocabulary	6 418	9 662
	OOVs (running words)	1 363 (3.8)	4 681
	OOVs (in voc.)	187 (2.9)	282

9.2.2.4 Arabic-English 300k NIST

The corpus statistics for the Arabic-English 300k NIST dataset are shown in Table 9.5.

9.2.2.5 Arabic-English Large NIST

The corpus statistics for the Arabic-English Large NIST dataset are shown in Table 9.6

Table 9.5: Corpus statistics for the Arabic-English 300k NIST dataset.

		Arabic	English
Train:	Sentences	294 046	
	Running Words	5 614 394	6 071 023
	Vocabulary	112 921	80 934
	Singletons	38 656	33 562
NIST'06	Sentences	1 797	
	Running Words	49 179	225 316
	Vocabulary	9 551	10 700
	OOVs (running words)	2 008 (4.1%)	7 867
	OOVs (in vocabulary)	768 (8.0%)	1 132
NIST'08	Sentences	1 357	
	Running Words	45 945	193 876
	Vocabulary	9 591	17 608
	OOVs (running words)	1 541 (3.7%)	48 435
	OOVs (in vocabulary)	624 (6.5%)	9 839
NIST'09	Sentences	1 313	
	Running Words	41 211	171 718
	Vocabulary	8 795	16 114
	OOVs (running words)	1 073 (2.6%)	42 100
	OOVs (in vocabulary)	505 (5.7%)	8 904

Table 9.6: Corpus statistics for the Arabic-English Large NIST dataset.

		Arabic	English
Train:	Sentences	4 561 210	
	Running Words	142 223 725	138 707 560
	Vocabulary	351 332	360 672
	Singletons	151 865	171 908
NIST'06	Sentences	1 797	
	Running Words	49 248	199 929
	Vocabulary	9 586	18 993
	OOVs (running words)	1 788 (3.6)	54 463
	OOVs (in voc.)	433 (4.5)	11 178
NIST'08	Sentences	1 357	
	Running Words	45 949	193 876
	Vocabulary	9 616	17 608
	OOVs (running words)	1 251 (2.7)	47 927
	OOVs (in voc.)	295 (3.1)	9 588

9.2.3 GALE Chinese-English Translation Task

The corpus statistics for the Chinese-English GALE dataset are shown in Table 9.7

Table 9.7: Corpus statistics for the Chinese-English GALE dataset.

		Chinese	English
Train	Sentences	10 004 219	
	Running Words	256 138 317	277 722 205
	Vocabulary	243 221	531 217
	Singletons	105 005	246 766
newswire DEV	Sentences	485	
	Running Words	14 747	66 278
	Vocabulary	3 539	5 826
	OOVs (running words)	7 (0.0)	3 576
	OOVs (in voc.)	6 (0.2)	345
newswire TEST	Sentences	480	
	Running Words	14 771	66 732
	Vocabulary	3 590	5 628
	OOVs (running words)	9 (0.1)	3 244
	OOVs (in voc.)	9 (0.3)	320
web text DEV	Sentences	533	
	Running Words	12 933	60 523
	Vocabulary	3 328	5 724
	OOVs (running words)	8 (0.1)	3 074
	OOVs (in voc.)	7 (0.2)	302
web text TEST	Sentences	490	
	Running Words	12 298	57 808
	Vocabulary	3 225	5 595
	OOVs (running words)	7 (0.1)	2 923
	OOVs (in voc.)	7 (0.2)	339

9.2.4 Celex 40k Grapheme to Phoneme Conversion

Grapheme to phoneme conversion is the task finding the correct pronunciation for a given written word. In practice, this is done by converting the letters, sometimes referred to by the more general term graphemes, into their phonetic transcription denoted by a sequence of phonemes. Graphemes may have different pronunciations depending on the immediate context or other factors.

The reasons for evaluating on the grapheme to phoneme conversion task are that the vocabulary is small, so that experiments do not take too many resources, and that we have comparable results with CRFs and other methods on this task. Table 9.8 shows the statistics for the Celex 40k grapheme to phoneme conversion task.

The evaluation of the grapheme to phoneme conversion task is done using two error measures. PER is the phoneme error rate that is computed as the symbol edit distance between predicted phoneme sequence and reference. SER is the sequence error rate, which is 0 only if the phoneme sequence hypothesis matches exactly the reference and 1 otherwise.

		Graphemes	Phonemes
Train:	Sentences	40 000	
	Running Words	334 583	282 732
	Vocabulary	26	52
	Singletons	0	1
Celex DEV	Sentences	5 000	
	Running Words	41 587	35 161
	Vocabulary	26	50
	OOVs (running words)	0 (0.0)	
	OOVs (in voc.)	0 (0.0)	
Celex TEST	Sentences	15 000	
	Running Words	125 696	106 143
	Vocabulary	26	50
	OOVs (running words)	0 (0.0)	
	OOVs (in voc.)	0 (0.0)	

Table 9.8: Corpus statistics for the Celex 40k grapheme to phoneme conversion task. The vocabulary on the input side are the graphemes or characters of the written English alphabet. On the output side we have the set of English phonemes.

9.3 Official Evaluations

Table 9.9: Comparison of the results of this work with the results from official NIST OpenMT Evaluations for the Arabic-English language pair

	Constrained			Unconstrained			This Work	
Evaluation	System	BLEU[%]	TER[%]	System	BLEU[%]	TER[%]	BLEU[%]	TER[%]
NIST'06	Google	42.8		Google	45.7		44.3	49.5
NIST'08	Google	45.3	48.5	Google	47.4	46.9	41.7	51.3
NIST'09	Cambridge	48.3	44.9	IBM	51.0	41.7	44.6	48.1

We have compared the performance of the translation models from this work with the results from the official NIST OpenMT Evaluations for Arabic-English (Table 9.9) and Chinese-English (Table 9.10). The results presented in the tables should be seen as a rough guideline to

Note that the comparison of the results of this work with results from official evaluations is difficult for various reasons: First, in this work, we present only single system results, while for evaluations, typically combinations of several systems are submitted. For efficiency

Table 9.10: Comparison of the results of this work with the results from official NIST OpenMT Evaluations for the Chinese-English language pair

	Constrained			Unconstrained			This work	
Evaluation	System	BLEU[%]	TER[%]	System	BLEU[%]	TER[%]	BLEU[%]	TER[%]
NIST'06	ISI	33.9		Google	36.2		0.0	0.0
NIST'08	BBN	29.6	57.1	Google	30.7	57.0	0.0	0.0

reasons, we also did not use all available data in all cases as this leads to very high memory and computing time requirements. In case of our "small" training sets Arabic-English 300k NIST or Chinese-English FBIS, we used just a rather small fraction of the training data. In addition to issues with the systems, there are also possible inconsistencies in scoring. Scoring scripts and reference length methods have changed over the past years introducing and removing specific side effects of scoring.

10 Conclusions

In this chapter, we will summarize the achievements of this work and point out directions for future work.

10.1 Summary

- We have described an algorithm and procedure for consistent training of phrase-based statistical translation models. The training involves a Forced Alignment part, where a phrase alignment is found between the source training instances and the given translation, using all models and components of the translation process. Improvements are shown in German-English, Chinese-English, and Arabic-English translation systems.
- We investigated different smoothing techniques that improve the generalization of our translation models when applied to previously unseen, out-of-domain test data. We introduced phrase count features and consistently-trained word-lexicon models. Both attempts result in improvements in translation quality. Our translation system using phrase count features was ranked first for Chinese-English and Japanese-English translation in the IWSLT 2006 Evaluation.
- Using the consistent training procedure, we were able to reduce the phrase table size to up to 33% without losing translation quality. The translation quality is improved for some of the language pairs.
- With our proposed extended lexicon models, we have shown that using machine-learning components can improve lexical selection in translation. Translation quality improves on large-scale Arabic-English and Chinese-English translation tasks.
- In order to train the extended lexicon models, we developed and implemented efficient algorithms and data structures for training large-scale sparse maximum entropy models. These methods reduce the time and memory requirement to train the extended lexicon models by several orders of magnitude.

10.2 Future Directions

- **More and Better Data.** The performance of data-driven approaches is closely linked to the quality and quantity of available training data. The more and the better the available data is, the better the systems will perform. For machine translation, promising steps have been taken in this direction by using comparable corpora or similarity-based search in large document collections [Munteanu & Fraser$^+$ 04],

[Uszkoreit & Ponte+ 10].

- **Adaptation and Unsupervised Training.** Statistical machine translation output has reached a level of quality that has already lead to a widespread use. Translation software and online services are used by private and commercial customers for a variety of tasks. Translating in instant messaging, social networking and other private communication is already used as well as the translation of customer reviews, product descriptions, newswire texts, patents, and other legal texts. The language for these applications varies from very colloquial to highly formalized. Not all these translations can be treated in the same way and adaptation techniques need to be developed and improved. The most common case of supervised adaptation in machine translation is using a translated in-domain development set to optimize system parameters. Beyond that, there have also been successful attempts in unsupervised adaptation (using only the source document) [Ueffing & Haffari+ 07] and unsupervised training (using larger amounts of monolingual in-domain data) [Schwenk & Senellart 09]. Also approaches of mixture model adaptation [Foster & Kuhn 07] have been successfully tested. Adaptation has a high practical relevance and has so far just been briefly studied.
- **Better Models.** Translation models at the moment tend to exploit only shallow dependencies in the source and target language. Phrase-based systems with n-gram language models are a good example for that. More and better data will surely help these methods. However there is also a large potential to exploit the information that is encoded implicitly in these large data collections. One of the most challenging and promising directions is the automatic acquisition of structural aspects of the language. This can go as for as full syntax of dependency structure at the sentence level or just plain reordering models. Even beyond the sentence level, there is potential information in the document context and structure. Successful attempts in this direction have been made by [Yamada & Knight 01], [Galley & Hopkins+ 04], [Zollmann & Venugopal 06], [Chiang 07], and [Golland & DeNero+ 12] to name just a few. Using more structure in the translation process can not only help to produce more accurate translations, but also to generate better, more fluent target language translations. There have also been successful and promising attempts to use richer featuresets for translations such as [Chiang & Knight+ 08], [Cherry 13] that allow to more accurately model details of the translation process and avoid common errors. Recently, also neural network models have been successfully applied to the machine translation problem [Schwenk & Dchelotte+ 06], [Schwenk 10], [Auli & Galley+ 13], [Mikolov & Le+ 13]. The continuous-space properties of these approaches offer interesting opportunities for future advances in the field.
- **Better Training.** Along with better models, better training procedures are also required. Discriminative training done in state-of-the-art systems resembles more a domain adaptation tasks, where few parameters (10-1000) are learned on a small, in-domain development set. Large-scale discriminative training with millions of features has been proposed for a long time [Berger & Brown+ 94], [Liang & Buchard-Côté+ 06], [Tillmann & Zhang 06], but in most cases they are not able to beat a simple baseline model. A notable exception is the Direct Translation Model 2 [Ittycheriah & Roukos 07] that has been shown to consistently achieve

good results for Arabic-English translation. The machine-learning world offers even more opportunities. Conditional Random Fields and Kernel methods for example are only starting to be used in the machine translation community. These methods are demanding computationally and pose new challenges in implementation and software design.

- **Better Evaluation.** Evaluation is probably the most challenging problem for machine translation in general. The challenge holds for both, human evaluation and automatic evaluation. It has been shown that even humans largely disagree on the quality of translations. Furthermore, human translations are difficult to obtain and not suited for rapid system development or tuning. Automatic measures on the other hand, suffer from the reliance on a single or very few reference translations that typically do not cover the full spectrum of possible translations. Both approaches, manual and common automatic evaluation measures, suffer from practical problems. Ideally a metric would be designed in a way that allows easy optimization of systems towards that metric. Current standard metrics such as Bleu and Ter however have complexity that makes them very hard to tune directly.

11 Publications and Team Work

11.1 Chapter Training

The work presented in this chapter is the result of a collaboration with Joern Wuebker and Hermann Ney published in [Wuebker & Mauser$^+$ 10]. The general, high-level idea was presented by Hermann Ney. The detailed idea and algorithm development was done by the author of this thesis (Section 4.4).

The implementation was shared between Joern Wuebker and the author of this thesis. The author of this thesis directly implemented the techniques described in 4.5, 4.7, 4.8.2, 4.9, 4.10, 4.11.1, and 4.11.2.

Systematic experimentation for the paper presented in [Wuebker & Mauser$^+$ 10] was designed by the author of this thesis and executed by Joern Wuebker. Analysis and verification of the results was done in collaboration of all authors of the paper.

All experimentation and analysis going beyond [Wuebker & Mauser$^+$ 10] was done solely by the author of this thesis.

11.2 Chapter Smoothing for Phrase Models

In this chapter there are sections that contain work that was published without contributions from the author of this thesis, sections that have been published by the author of this thesis, and previously unpublished work.

The models described in Section 5.2.1, only contains work that was published without contributions from the author of this thesis.

The method describe in Section 5.2.2 has been published as part of [Mauser & Zens$^+$ 06] and is augmented here with additional experiments. The idea for this method was developed with Richard Zens who also contributed the implementation. The design and execution of the experimental evaluation and the analysis of the results was done by the author of this thesis, both for the original publication and for the additional work presented in this thesis.

Sections 5.3 and 5.4 describe new methods that have not been published prior to this thesis. The idea for these models was developed by the author of this thesis who also did the implementation, empirical evaluation and analysis.

11.3 Chapter Reordering Models

The novel work presented in this chapter was not published prior to this thesis. Unless specifically marked, ideas, concepts, implementation, experimentation, evaluation, and analysis has been performed by the author of this thesis.

Novel models are presented in Sections 6.5 and 6.6, with some additional explanations in Section 6.4.

While some of the ideas had been explored before in the literature (see Section 6.2), the idea of learning the model parameters in the way described in this chapter is new.

All implementation of training and the integration into the statistical machine translation system were done by the author of this thesis. The same holds for the design, execution, verification and analysis of all the experimental evaluation.

11.4 Chapter Extended Lexicon Models

This chapter is based on [Mauser & Hasan$^+$ 09], a joint paper with Saša Hasan and Hermann Ney.

The author of this thesis conceived the idea of the discriminative lexicon model for phrase-based translation as described in Section 7.4.

Similar approaches have been used in the literature. See Section 7.2 for a review of the related work.

The implementation of the training (see Section 7.6) and the integration into the statistical machine translation system (see Section 7.7) was done by the author of this thesis. While initially a third party tool was used to train models, a new training was implemented for this thesis.

Experiments presented in [Mauser & Hasan$^+$ 09] involving the discriminative lexicon model were designed and performed by the author of this theses. The design of the experiments was done in cooperation with Saša Hasan to ensure comparability of experimental results for different models.

All other experiments were designed and performed solely by the author of this thesis.

The analysis of the experimental results presented in [Mauser & Hasan$^+$ 09] was done in collaboration with Saša Hasan. The author of this thesis performed the detailed analysis of the discriminative lexicon model and contributed to the analysis and comparison of the joint results.

List of Figures

List of Tables

Bibliography

[Al-Onaizan & Papineni 06] Y. Al-Onaizan, K. Papineni: Distortion models for statistical machine translation. Proc. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pp. 529–536, Stroudsburg, PA, USA, July 2006.

[Andrés Ferrer 10] J. Andrés Ferrer: *Statistical approaches for natural language modelling and monotone statistical machine translation.* Ph.D. thesis, UNIVERSIDAD POLITÉCNICADE VALENCIA, 2010.

[Andrés Ferrer & Juan 09] J. Andrés Ferrer, A. Juan: A phrase-based hidden semi-Markov approach to machine translation. Proc. *Procedings of European Association for Machine Translation (EAMT)*, pp. 168–175, Barcelona, Spain, May 2009. European Association for Machine Translation.

[Auli & Galley$^+$ 13] M. Auli, M. Galley, C. Quirk, G. Zweig: Joint Language and Translation Modeling with Recurrent Neural Networks. Proc. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1044–1054, Seattle, Washington, USA, Oct. 2013.

[Bangalore & Haffner$^+$ 06] S. Bangalore, P. Haffner, S. Kanthak: Sequence Classification for Machine Translation. Proc. *Ninth International Conf. on Spoken Language Processing, Interspeech 2006 — ICSLP*, pp. 1722–1725, Pitsburgh, PA, Sept. 2006.

[Bangalore & Haffner$^+$ 07] S. Bangalore, P. Haffner, S. Kanthak: Statistical Machine Translation through Global Lexical Selection and Sentence Reconstruction. Proc. *45th Annual Meeting of the Association of Computational Linguistics*, pp. 152–159, Prague, Czech Republic, June 2007.

[Bellman 57] R. Bellman: *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.

[Bender & Zens$^+$ 04] O. Bender, R. Zens, E. Matusov, H. Ney: {A}lignment {T}emplates: the {RWTH SMT S}ystem. Proc. *Int.˜Workshop on Spoken Language Translation (IWSLT)*, pp. 79–84, Kyoto, Japan, Sept. 2004.

[Berger & Brown$^+$ 94] A.L. Berger, P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, J.R. Gillett, J.D. Lafferty, H. Printz, L. Ures: The {C}andide System for Machine Translation. Proc. *ARPA Workshop on Human Language Technology*, pp. 157–162, Plainsboro, NJ, March 1994.

[Birch & Callison-Burch$^+$ 06] A. Birch, C. Callison-Burch, M. Osborne, P. Koehn: Constraining the Phrase-Based, Joint Probability Statistical Translation Model. Proc. *smt2006*, pp. 154–157, June 2006.

[Blunsom & Cohn 06] P. Blunsom, T. Cohn: Discriminative word alignment with condi-

tional random fields. Proc. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL - ACL '06*, pp. 65–72, Sydney, Australia, July 2006.

[Blunsom & Cohn$^+$ 08] P. Blunsom, T. Cohn, M. Osborne: A Discriminative Latent Variable Model for Statistical Machine Translation. Proc. *Proceedings of ACL-08: HLT*, pp. 200–208, Columbus, Ohio, June 2008.

[Bonneau-Maynard & Rosset$^+$ 05] H. Bonneau-Maynard, S. Rosset, C. Ayache, A. Kuhn, D. Mostefa: Semantic Annotation of the French Media Dialog Corpus. Proc. *ISCA Eurospeech*, pp. 3457–3460, Lisbon, Sept. 2005.

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

[Brown & Della Pietra$^+$ 93] P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, R.L. Mercer: The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, Vol. 19, No. 2, pp. 263–311, June 1993.

[Callison-Burch & Koehn$^+$ 10] C. Callison-Burch, P. Koehn, C. Monz, K. Peterson, M. Przybocki, O. Zaidan: Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. Proc. *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pp. 17–53, July 2010.

[Carpuat & Wu 07] M. Carpuat, D. Wu: Improving Statistical Machine Translation using Word Sense Disambiguation. Proc. *Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pp. 61–72, Prague, Czech Republic, June 2007.

[Cettolo & Federico 04] M. Cettolo, M. Federico: Minimum error training of log-linear translation models. Proc. *International Workshop on Spoken Language Translation (IWSLT)*, pp. 103–106, Kyoto, Japan, Sept. 2004.

[Chan & Ng$^+$ 07] Y.S. Chan, H.T. Ng, D. Chiang: Word Sense Disambiguation Improves Statistical Machine Translation. Proc. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 33–40, Prague, Czech Republic, June 2007.

[Chen & Goodman 98] S.F. Chen, J. Goodman: An Empirical Study of Smoothing Techniques for Language Modeling, 63 pages, Aug. 1998.

[Cherry 13] C. Cherry: Improved Reordering for Phrase-Based Translation using Sparse Features. Proc. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 22–31, Atlanta, Georgia, June 2013.

[Chiang 07] D. Chiang: Hierarchical Phrase-Based Translation. *Computational Linguistics*, Vol. 33, No. 2, pp. 201–228, June 2007.

[Chiang & Knight$^+$ 08] D. Chiang, K. Knight, W. Wang: 11,001 New Features for Statistical Machine Translation. Proc. *Machine Translation*, NAACL '09, pp. 218–226, Stroudsburg, PA, USA, May 2008.

[Costa-Jussà & Fonollosa 06] M.R. Costa-Jussà, J.A.R. Fonollosa: Statistical machine reordering. Proc. *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pp. 70–76, Stroudsburg, PA, USA, July 2006.

[Dempster & Laird+ 77] A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, Vol. 39, No. 1, pp. 1–22, 1977.

[DeNero & Buchard-Côté+ 08] J. DeNero, A. Buchard-Côté, D. Klein: Sampling Alignment Structure under a Bayesian Translation Model. Proc. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 314–323, Honolulu, Oct. 2008.

[DeNero & Gillick+ 06] J. DeNero, D. Gillick, J. Zhang, D. Klein: Why generative phrase models underperform surface heuristics. Proc. *Proceedings of the Workshop on Statistical Machine Translation - StatMT '06*, pp. 31–38, Morristown, NJ, USA, June 2006.

[DeNero & Klein 08] J. DeNero, D. Klein: The complexity of phrase alignment problems. Proc. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pp. 25–28, Morristown, NJ, USA, June 2008.

[Dyer & Clark+ 11] C. Dyer, J. Clark, A. Lavie, N.a. Smith: Unsupervised Word Alignment with Arbitrary Features. Proc. *49th Annual Meeting of the Association for Computational Linguistics*, HLT '11, pp. 409–419, Portland, OR, June 2011.

[Ehling & Zens+ 07] N. Ehling, R. Zens, H. Ney: Minimum Bayes Risk decoding for BLEU. Proc. *ACL '07: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pp. 101–104, Morristown, NJ, USA, June 2007.

[Foster 00] G. Foster: A maximum entropy/minimum divergence translation model. Proc. *38th Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pp. 37–44, Hong Kong, Oct. 2000.

[Foster & Kuhn+ 06] G. Foster, R. Kuhn, H. Johnson: Phrasetable Smoothing for Statistical Machine Translation. Proc. *Conf.\ on Empirical Methods for Natural Language Processing (EMNLP)*, pp. 53–61, Sydney, Australia, July 2006.

[Foster & Kuhn 07] G. Foster, R. Kuhn: Mixture-model adaptation for SMT. Proc. *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pp. 128–135, Stroudsburg, PA, USA, June 2007.

[Galley & Hopkins+ 04] M. Galley, M. Hopkins, K. Knight, D. Marcu: What's in a translation rule? Proc. *Human Language Technology Conf. / North American Chapter of the Assoc. ˜for Computational Linguistics Annual Meeting (HLT-NAACL)*, pp. 273–280, Boston, MA, May 2004.

[Garcia-Varea & Och+ 01] I. Garcia-Varea, F.J. Och, H. Ney, F. Casacuberta: Refined Lexicon Models for Statistical Machine Translation using a Maximum Entropy Approach. Proc. *39th Annual Meeting of the Assoc. ˜for Computational Linguistics (ACL)*, pp. 204–211, Toulouse, France, July 2001.

[Golland & DeNero+ 12] D. Golland, J. DeNero, J. Uszkoreit: A Feature-rich Constituent

Context Model for Grammar Induction. Proc. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pp. 17–22, Stroudsburg, PA, USA, Aug. 2012.

[Graham & Knuth$^+$ 94] R.L. Graham, D.E. Knuth, O. Patashnik: *Concrete Mathematics*. Addison-Wesley Publishing Company, Reading, MA, 2nd edition, 1994.

[Gupta & Nath$^+$ 10] K.K. Gupta, B. Nath, R. Kotagiri: Layered Approach Using Conditional Random Fields for Intrusion Detection. *IEEE Transactions on Dependable and Secure Computing*, Vol. 7, No. 1, pp. 35–49, 2010.

[Hahn & Lehnen$^+$ 08] S. Hahn, P. Lehnen, C. Raymond, H. Ney: A Comparison of Various Methods for Concept Tagging for Spoken Language Understanding. Proc. *International Conference on Language Resources and Evaluation*, pp. 2947–2950, Marrakech, Morocco, May 2008.

[Hasan & Ganitkevitch$^+$ 08] S. Hasan, J. Ganitkevitch, H. Ney, J. Andrés-Ferrer: Triplet Lexicon Models for Statistical Machine Translation. Proc. *EMNLP*, pp. 372–381, Honolulu, Hawaii, Oct. 2008.

[Hasan & Ney 09] S. Hasan, H. Ney: Comparison of Extended Lexicon Models in Search and Rescoring for SMT. Proc. *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies conference*, Vol. short pape, pp. 17–20, Boulder, Colorado, June 2009.

[He & Zemel$^+$ 04] X. He, R.S. Zemel, M.A. Carreira-Perpinan: Multiscale conditional random fields for image labeling. Proc. *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, Vol. 2, pp. II–695—-II–702, June 2004.

[Ittycheriah & Roukos 07] A. Ittycheriah, S. Roukos: Direct Translation Model 2. Proc. *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pp. 57–64, Rochester, New York, April 2007.

[Jelinek 98] F. Jelinek: *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA, 1998.

[Jiampojamarn & Cherry$^+$ 10] S. Jiampojamarn, C. Cherry, G. Kondrak: Integrating Joint n-gram Features into a Discriminative Training Framework. Proc. *In Proceeding of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pp. 697–700, June 2010.

[Kneser & Ney 95] R. Kneser, H. Ney: Improved Backing-Off for M-gram Language Modelling. Proc. *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 181–184, Detroit, MI, May 1995.

[Knight 99] K. Knight: Decoding complexity in word-replacement translation models. *Computational Linguistics*, Vol. 25, No. 4, pp. 607–615, 1999.

[Koehn 03] P. Koehn: *Noun Phrase Translation,*. Phd thesis, University of Southern California, 2003.

[Koehn 10] P. Koehn: *Statistical Machine Translation*. Cambridge University Press,

Cambridge, UK, 2010.

[Koehn & Hoang+ 07] 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. Constantine, E. Herbst: Moses: Open Source Toolkit for Statistical Machine Translation. Proc. *45th Annual Meeting of the Assoc. for Computational Linguistics (ACL): Poster Session*, pp. 177–180, Prague, Czech Republic, June 2007.

[Koehn & Och+ 03] P. Koehn, F.J. Och, D. Marcu: Statistical phrase-based translation. Proc. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pp. 48–54, Edmonton, Canada, May 2003.

[Lafferty & McCallum+ 01] J. Lafferty, A. McCallum, F. Pereira: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Proc. *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289, San Fransisco, June 2001. Morgan Kaufmann.

[Lavergne & Allauzen+ 11] T. Lavergne, A. Allauzen, F. Yvon, J.M. Crego: From n-gram-based to crf-based translation models. Proc. *Proceedings of the 6th Workshop on Statistical Machine Translation (WMT)*, pp. 542–553, July 2011.

[Lehnen & Hahn+ 11a] P. Lehnen, S. Hahn, A. Guta, H. Ney: Incorporating Alignments into Conditional Random Fields for Grapheme to Phoneme Conversion. Proc. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 4916–4919, Prague, Czech Republic, May 2011.

[Lehnen & Hahn+ 11b] P. Lehnen, S. Hahn, H. Ney: N-grams for Conditional Random Fields or a Failure-transition Posterior for Acyclic FSTs. Proc. *Interspeech*, pp. 1437–1440, Florence, Italy, Aug. 2011.

[Lehnen & Peter+ 13] P. Lehnen, J.T. Peter, J. Wuebker, S. Peitz, H. Ney: (Hidden) Conditional Random Fields Using Intermediate Classes for Statistical Machine Translation. Proc. *Machine Translation Summit*, pp. 151–158, Nice, France, Sept. 2013.

[Liang & Buchard-Côté+ 06] P. Liang, A. Buchard-Côté, D. Klein, B. Taskar: An End-to-End Discriminative Approach to Machine Translation. Proc. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pp. 761–768, Sydney, Australia, July 2006.

[Macherey 10] W. Macherey: *Discriminative Training and Acoustic Modeling for Automatic Speech Recognition*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, March 2010.

[Macherey & Och+ 08] W. Macherey, F.J. Och, I. Thayer, J. Uszkoreit: Lattice-based Minimum Error Rate Training for Statistical Machine Translation. Proc. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 725–734, Honolulu, Hawaii, Oct. 2008.

[Marcu & Wong 02] D. Marcu, W. Wong: A Phrase-Based, Joint Probability Model for Statistical Machine Translation. Proc. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pp. 133–139, July 2002.

[Mauser & Hasan+ 09] A. Mauser, S. Hasan, H. Ney: Extending statistical machine translation with discriminative and trigger-based lexicon models. Proc. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pp. 210–218, Stroudsburg, PA, USA, Aug. 2009.

[Mauser & Zens+ 06] A. Mauser, R. Zens, E. Matusov, S. Hasan, H. Ney: The RWTH Statistical Machine Translation System for the IWSLT 2006 Evaluation. Proc. *International Workshop on Spoken Language Translation (IWSLT)*, pp. 103–110, Kyoto, Japan, Nov. 2006.

[McCallum & Li 03] A. McCallum, W. Li: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. Proc. *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pp. 188–191, Stroudsburg, PA, USA, May 2003.

[Melamed 97] I.D. Melamed: Automatic Discovery of Non-Compositional Compounds in Parallel Data. Proc. *Empirical Methods in Natural Language Processing (EMNLP)*, Vol. 1542, pp. 33–36, Providence, RI, Aug. 1997.

[Melamed 00] I.D. Melamed: Models of translational equivalence among words. *Computational Linguistics*, Vol. 26, No. 2, pp. 221–249, 2000.

[Mikolov & Le+ 13] T. Mikolov, Q.V. Le, I. Sutskever: Exploiting Similarities among Languages for Machine Translation. *CoRR*, Vol. abs/1309.4, 2013.

[Moore & Quirk 07] R.C. Moore, C. Quirk: An iteratively-trained segmentation-free phrase translation model for statistical machine translation. Proc. *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pp. 112–119, Stroudsburg, PA, USA, June 2007.

[Munteanu & Fraser+ 04] D.S. Munteanu, A. Fraser, D. Marcu: Improved Machine Translation Performance via Parallel Sentence Extraction from Comparable Corpora. Proc. *Human Language Technology Conf. / North American Chapter of the Assoc. ˜for Computational Linguistics Annual Meeting (HLT-NAACL)*, pp. 265–272, Boston, MA, May 2004.

[Nelder & Mead 65] J.A. Nelder, R. Mead: A Simplex Method for Function Minimization. *The Computer Journal)*, Vol. 7, pp. 308–313, 1965.

[Ney 01] H. Ney: Stochastic Modelling: From Pattern Classification to Language Translation. Proc. *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL): Workshop on Data-Driven Machine Translation*, pp. 1–5, Morristown, NJ, July 2001.

[Och 03] F.J. Och: *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Phd thesis, 2003.

[Och & Gildea+ 04] 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, D. Radev: A Smorgasbord of Features for Statistical Machine Translation. Proc. *Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics Annual Meeting (HLT-NAACL)*, pp. 161–168, Boston, MA, May 2004.

[Och & Ney 01] F.J. Och, H. Ney: Discriminative training and maximum entropy models

for statistical machine translation. Proc. *Workshop: MT 2010 - Towards a Road Map for MT*, pp. 295–302, Morristown, NJ, USA, July 2001.

[Och & Ney 03] F.J. Och, H. Ney: A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, Vol. 29, No. 1, pp. 19–51, March 2003.

[Och & Ney 04] F.J. Och, H. Ney: The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, Vol. 30, No. 4, pp. 417–449, 2004.

[Och & Tillmann$^+$ 99] F.J. Och, C. Tillmann, H. Ney: Improved Alignment models for Statistical Machine Translation. Proc. *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP)*, pp. 20–28, College Park, MD, USA, June 1999.

[Papineni & Roukos$^+$ 02] K. Papineni, S. Roukos, T. Ward, W.J. Zhu: BLEU: a method for automatic evaluation of machine translation. Proc. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 311–318, Morristown, NJ, USA, July 2002.

[Peitz & Mauser$^+$ 12] S. Peitz, A. Mauser, J. Wuebker, H. Ney: Forced Derivations for Hierarchical Machine Translation. Proc. *International Conference on Computational Linguistics*, pp. 933–942, Mumbai, India, Dec. 2012.

[Rabiner 90] L.R. Rabiner: A Tutorial on Hidden Markov Models and Selected Aplications in Speech Recognition. In A. Waibel, K.F. Lee, editors, *Readings in Speech Recognition*, pp. 267–296. Kaufmann, San Mateo, CA, 1990.

[Riedmiller & Braun 93] M. Riedmiller, H. Braun: A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. Proc. *IEEE INTERNATIONAL CONFERENCE ON NEURAL NETWORKS*, pp. 586–591, March 1993.

[Schwenk 10] H. Schwenk: Continuous-space language models for statistical machine translation. *The Prague Bulletin of Mathematical Linguistics*, Vol. 93, No. 1, pp. 137–146, 2010.

[Schwenk & Dchelotte$^+$ 06] H. Schwenk, D. Dchelotte, J.L. Gauvain: Continuous Space Language Models for Statistical Machine Translation. Proc. *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, COLING-ACL '06, pp. 723–730, Stroudsburg, PA, USA, July 2006.

[Schwenk & Senellart 09] H. Schwenk, J. Senellart: Translation Model Adaptation for an Arabic/French News Translation System by Lightly-Supervised Training. Proc. *MT Summit XII*, Ottawa, Ontario, Canada, Aug. 2009.

[Searle 80] J.R. Searle: Minds, brains, and programs. *Behavioral and Brain Sciences*, Vol. 3, No. 03, pp. 417–424, Feb. 1980.

[Shen & Delaney$^+$ 08] W. Shen, B. Delaney, T. Anderson, R. Slyh: The MIT-LL/AFRL IWSLT-2008 MT System. Proc. *Proceedings of IWSLT 2008*, pp. 69–76, Hawaii, U.S.A., Oct. 2008.

[Snover & Dorr$^+$ 06] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, J. Makhoul: A Study of Translation Edit Rate with Targeted Human Annotation. Proc. *Conf. of the Assoc. for Machine Translation in the Americas (AMTA)*, pp. 223–231, Cambridge, MA, Aug.

2006.

[Steinbiss & Tran+ 94] V. Steinbiss, B.H. Tran, H. Ney: Improvements in Beam Search. Proc. *Int. Conf. on Spoken Language Processing (ICSLP)*, pp. 2143–2146, Sept. 1994.

[Stolcke 02] A. Stolcke: {SRILM} – An Extensible Language Modeling Toolkit. Proc. *Proc. Int. Conf. on Spoken Language Processing*, Vol. 2, pp. 901–904, Denver, CO, Sept. 2002.

[Tillmann & Ney 03] C. Tillmann, H. Ney: Word Reordering and a Dynamic Programming Beam Search Algorithm for Statistical Machine Translation. *Comput. Linguist.*, Vol. 29, No. 1, pp. 97–133, 2003.

[Tillmann & Zhang 05] C. Tillmann, T. Zhang: A Localized Prediction Model for Statistical Machine Translation. Proc. *43rd Annual Meeting of the Assoc. ˜for Computational Linguistics (ACL)*, pp. 557–564, Ann Arbor, MI, June 2005.

[Tillmann & Zhang 06] C. Tillmann, T. Zhang: A discriminative global training algorithm for statistical MT. Proc. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL - ACL '06*, pp. 721–728, Morristown, NJ, USA, July 2006.

[Tromble & Kumar+ 08] R.W. Tromble, S. Kumar, F. Och, W. Macherey: Lattice Minimum Bayes-Risk decoding for statistical machine translation. Proc. *Proceedings of the Conference on Empirical Methods in Natural Language Processing - EMNLP '08*, pp. 620–629, Honolulu, HI, Oct. 2008.

[Ueffing & Haffari+ 07] N. Ueffing, G. Haffari, A. Sarkar: Semi-supervised model adaptation for statistical machine translation. *Machine Translation*, Vol. 21, No. 2, pp. 77–94, June 2007.

[Ueffing & Och+ 02] N. Ueffing, F.J. Och, H. Ney: Generation of Word Graphs in Statistical Machine Translation. Proc. *Proc. of the Conference on Empirical Methods for Natural Language Processing*, pp. 156–163, Philadelphia, PA, USA, July 2002.

[Uszkoreit & Ponte+ 10] J. Uszkoreit, J. Ponte, A. Popat, M. Dubiner: Large Scale Parallel Document Mining for Machine Translation. Proc. *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pp. 1101–1109, Beijing, China, Aug. 2010. Coling 2010 Organizing Committee.

[Venkatapathy & Bangalore 07] S. Venkatapathy, S. Bangalore: Three models for discriminative machine translation using Global Lexical Selection and Sentence Reconstruction. Proc. *SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pp. 96–102, Rochester, New York, April 2007.

[Vogel & Ney+ 96] S. Vogel, H. Ney, C. Tillmann: HMM-based Word Alignment in Statistical Machine Translation. Proc. *Proceedings of the 16th conference on Computational linguistics*, pp. 836–841, Santa Cruz, CA, June 1996.

[Wu & Xia 94] D. Wu, X. Xia: Learning an English-Chinese lexicon from a parallel corpus. Proc. *In Proceedings of the First Conference of the Association for Machine Translation in the Americas*, pp. 206–213, Columbia, MD, Oct. 1994.

[Wuebker & Mauser+ 10] J. Wuebker, A. Mauser, H. Ney: Training Phrase Translation

Models with Leaving-One-Out. Proc. *Annual Meeting of the Assoc. for Computational Linguistics*, pp. 475–484, Uppsala, Sweden, July 2010.

[Yamada & Knight 01] K. Yamada, K. Knight: A Syntax-Based Statistical Translation Model. Proc. *39th Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pp. 523–530, Toulouse, France, July 2001.

[Zens 08] R. Zens: *Phrase-based Statistical Machine Translation: Models, Search, Training*. Ph.D. thesis, Computer Science Department, RWTH Aachen – University of Technology, Germany, Feb. 2008.

[Zens & Ney$^+$ 04] R. Zens, H. Ney, T. Watanabe, E. Sumita: Reordering Constraints for Phrase-Based Statistical Machine Translation. Proc. *20th Int. Conf. on Computational Linguistics (COLING)*, pp. 205–211, Geneva, Switzerland, Aug. 2004.

[Zens & Ney 05] R. Zens, H. Ney: Word Graphs for Statistical Machine Translation. Proc. *43rd Annual Meeting of the Assoc. for Computational Linguistics (ACL): Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, pp. 191–198, Ann Arbor, MI, June 2005.

[Zens & Ney 06] R. Zens, H. Ney: Discriminative Reordering Models for Statistical Machine Translation. Proc. *Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics Annual Meeting (HLT-NAACL): Workshop on Statistical Machine Translation*, pp. 55–63, New York City, NY, June 2006.

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

[Zhang & Zens$^+$ 07] Y. Zhang, R. Zens, H. Ney: Improved Chunk-level Reordering for Statistical Machine Translation. Proc. *Int. Workshop on Spoken Language Translation (IWSLT)*, pp. 21–28, Trento, Italy, Oct. 2007.

[Zollmann & Venugopal 06] A. Zollmann, A. Venugopal: Syntax Augmented Machine Translation via Chart Parsing. Proc. *Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics Annual Meeting (HLT-NAACL): Workshop on Statistical Machine Translation*, pp. 138–141, New York City, NY, June 2006.

[Zweig & Nguyen 09] G. Zweig, P. Nguyen: A Segmental CRF Approach to Large Vocabulary Continuous Speech Recognition. Proc. *IEEE Automatic Speech Recognition and Understanding Workshop*, pp. 152–157, Merano, Italy, Dec. 2009.