Adaptation in Machine Translation

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der Fakultät für Informatik

des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Jan Niehues

aus Borken

Tag der mündlichen Prüfung: 17.01.2014

Erster Gutachter: Prof. Dr. Alexander Waibel

Zweiter Gutachter: Prof. Dr. François Yvon

Abstract

Machine translation remains one of the grand challenge problems of natural language processing. Recent advances in the field have led to a number of applications demonstrating the potential and impact of the technology.

Statistical machine translation (SMT) has emerged as the currently most promising approach to tackle the translation problem. During the last decade, it advanced to solidly outperform rule-based methods over many tasks and evaluations. The main advantage of the statistical approach is that expert knowledge is no longer manually encoded rule by rule, but translation models can be learned automatically from parallel data consisting of translated texts. This leads to faster and cheaper development of new translation systems.

One limitation to date, however, is that the quality of SMT systems strongly depends on the similarity between the training data and its deployment. When a machine translation system is applied to a new task, performance typically drops significantly. And since huge amounts of data are needed for training, it is not possible to collect only matching data for every new application. This thesis is devoted to adapting MT systems in the scenario of mismatching training data. We develop different approaches to increase translation performance even though all or some of the data we used for training does not match the system's ultimate application.

In order to improve translation quality when applying a translation system to a new task, we explore four different approaches: integration of mismatching data, combining matching and mismatching data, adapting the system to very specific topics, and exploiting data matching in genre only. We present techniques and experimental systems that improve the translation quality for a particular type of given data.

We show that the context available during the translation process is shorter when using mismatching data. In order to address this problem, we develop a bilingual language model to increase the context that is available during decoding. Using this model we improve the ability of the system to exploit mismatching data and we show that this results in improved translation quality.

Our training data is typically derived from varied sources encompassing different topics and genres. Consequently, some parts of the data might match the task better than others. In response, we weight the different parts of the training data so that the influence of the matching data can be increased. By avoiding a binary decision whether the data matches or not, we can make better use of the corpora that match our target task to a certain degree. We show significant improvements by combining models trained on in-domain and out-of-domain data.

In order to enable the translation of topic-specific terms, data that matches the topic is needed. For most applications, however, it is difficult to obtain data that matches both in topic and in genre. Therefore, we present an approach to include data in the translation system that only matches the topic of the input data. We use the titles of Wikipedia articles to translate topic-specific terms. Since this type of data does not contain all possible word forms, we also develop techniques to find translations for morphological variations of the same word.

Another problem addressed in this thesis is the adaptation of a translation system to the genre of an application. In order to enable better exploitation of data that matches in genre, but not in topic, we present a continuous space language model. We show that this model generalizes better when topic-specific words occur than an n-gram language model.

We perform a detailed analysis of the impact of all these approaches on the task of translating different types of data and show their positive influence in systems submitted to international evaluation campaigns.

Zusammenfassung

Maschinelle Übersetzung ist eine der größten Herausforderungen im Bereich der Verarbeitung natürlicher Sprache. In den letzten Jahren haben Weiterentwicklungen in diesem Bereich neue Anwendungen der maschinellen Übersetzung eröffnet und das Potenzial dieser Technologie gezeigt.

Es gibt verschiedene Ansätze zur maschinellen Übersetzung. Zu Beginn wurden regelbasierte Ansätze und auf Interlingua basierende Systeme eingesetzt. In den letzten 20 Jahren hat sich der Ansatz der statistischen maschinellen Übersetzung zum vielversprechendsten entwickelt. Mittels dieser Methode konnte in vielen Anwendungen und Evaluationen eine Verbesserung der Übersetzungsqualität erreicht werden. Einer der Hauptvorteile der statistischen maschinellen Übersetzung ist, dass kein Expertenwissen bei der Entwicklung der Systeme benötigt wird, sondern Algorithmen benutzt werden, die aus parallelen Daten die Übersetzungssysteme automatisch erlernen. Dies führt zu kürzeren Entwicklungszeiten und verringert die Kosten für die Entwicklung von neuen Übersetzungssystemen.

Auch wenn die Verwendung von parallelen Daten zu einer signifikanten Verbesserung der Übersetzungsqualität geführt hat, ergeben sich durch ihre Verwendung neue Probleme. In heutigen Systemen hängt die Übersetzungsqualität von der Ähnlichkeit zwischen den Trainingsdaten und dem zu übersetzenden Text ab. Nur wenn diese sehr ähnlich sind, kann eine zufriedenstellende Qualität erreicht werden. Wenn hingegen das System für eine neue Aufgabe eingesetzt wird, kann es schnell zu einer deutlichen Verschlechterung der Übersetzungsqualität kommen. Da für das Training der Systeme sehr große Mengen an parallelen Daten benötigt werden, können allerdings nicht für jede neue Anwendung neue Daten erstellt werden. Daher beschäftigt sich diese Arbeit mit der Verbesserung der Übersetzungsqualität

in Anwendungsfällen, in denen keine oder nur teilweise passende Daten zur Verfügung stehen.

Um diese Aufgabe zu erfüllen, wurden Techniken für vier verschiedene Probleme entwickelt: die Integration von Daten, die sich in der Art sehr von den Eingabedaten der Anwendung unterscheiden, die Kombination von ähnlichen und weniger ähnlichen Daten, die Adaption eines Systems an sehr spezielle Themen und eine bessere Integration von Daten, die nur mit dem Genre der Anwendung übereinstimmen. Für alle diese Probleme wurden neue Ansätze entwickelt, die die Übersetzungsqualität in Anwendungsfällen verbessern, in denen die entsprechenden Daten zur Verfügung stehen.

Zunächst hat sich herausgestellt, dass der Kontext, der während des Übersetzungsprozesses zur Verfügung steht, kleiner ist, wenn die Daten nicht zur Anwendung passen. In der Arbeit wurde das "Bilingual Language Model" entwickelt, das den verfügbaren Kontext während des Übersetzungsprozesses vergrößert. Mittels dieses Modells ist es möglich, mehr Nutzen aus Daten zu ziehen, die den Eingabedaten der Anwendung weniger ähneln. Somit konnte die Übersetzungsqualität unter diesen Bedingungen verbessert werden.

In vielen Szenarien ist ein Teil der Daten ähnlich zu den Eingabedaten der Anwendung. Ein deutlich größerer Teil unterscheidet sich stärker von der Anwendung. Deshalb wurden im Rahmen der Arbeit Methoden entwickelt, um verschiedene Teile des Trainingscorpus unterschiedlich zu gewichten. Dadurch ist es möglich, den Einfluss der passenden Daten zu erhöhen. Ein Vorteil des Ansatzes ist, dass keine binären Entscheidungen getroffen werden, sondern den einzelnen Teilen des Corpus Gewichte zugeordnet werden. Dadurch ist es auch möglich, Daten die nur teilweise passen, optimal auszunutzen. In der Arbeit zeigen wir eine signifikante Verbesserung der Übersetzungsqualität durch die Kombination von Modellen, die aus passenden und nicht passenden Daten trainiert wurden.

Um das System auch zur Übersetzung von themenspezifischem Vokabular zu benutzen, benötigt man Daten, die zu dem jeweiligem Thema passen. Für die meisten Anwendungen ist es allerdings nicht möglich, Daten zu finden,

die sowohl im Thema als auch im Genre mit der Anwendung übereinstimmen. Daher präsentieren wir in dieser Arbeit Techniken, die es ermöglichen, auch Daten in das System einzubinden, die nur mit dem Thema der Anwendung übereinstimmen. Als einen möglichen Ansatz benutzen wir die Titel aus Wikipedia um themenspezifische Terminologie zu übersetzen. Da in diesen Daten nicht alle morphologischen Varianten der Wörter vorkommen, entwickeln wir einen Ansatz, um aus den Übersetzungen der Wikipedia-Titel Übersetzungen für unbekannte morphologische Varianten zu lernen.

Ein weiteres Problem ist die Adaptatierung eines Systems an das Genre der Anwendung. Um Daten, die nur in Bezug auf das Genre zu der Anwendung passen, besser zu integrieren, haben wir ein "Continuous Space Language Model" entwickelt. Mittels dieses Sprachmodells können Wordsequenzen, die themenspezifische Wörter enthalten, besser generalisiert werden.

In der Arbeit geben wir einen detaillierten Überblick über den Einfluss der entwickelten Modelle auf die Übersetzungsqualität bei der Übersetzung von verschiedenen Arten von Vorträgen. Darüber hinaus zeigen wir den Einfluss auf die Ergebnisse von Systemen, die bei internationalen Evaluationen benutzt wurden.

Acknowledgements

I would like to thank my first advisor Prof. Alex Waibel for giving me the opportunity to perform the research leading to this thesis and for the constructive discussions. Furthermore, I would like to thank Prof. François Yvon for co-advising this thesis. He showed great interest in the thesis and gave me valuable input.

Furthermore, all members of the MT team at KIT helped me during all phases of my research and only by the joint work in this team, it was possible to achieve the results presented in this thesis. My thanks go to: Eunah Cho, Thanh-Le Ha, Silja Hildebrand, Teresa Herrmann, Muntsin Kolss, Mohammed Mediani, Carsten Schnober, Isabel Slawik, and Yuqi Zhang. Furthermore, I would like to thank Stephan Vogel, who introduced me to the exciting field of machine translation.

Also all the other colleagues of the Interactive System Labs were very helpful with any problem I had and the exchange in the team led to very interesting ideas. Therefore, I would like to thank all of them: Silke Dannenmaier, Sarah Fünfer, Jonas Gehring, Michael Heck, Hartwig Holzapfel, Klaus Joas, Kevin Kilgour, Narine Kokhlikyan, Florian Kraft, Bastian Krüger, Patricia Lichtblau, Christian Mohr, Markus Müller, Huy Van Nguyen, Bao Quoc Nguyen, Matthias Paulik, Margit Rödder, Virginia Roth, Christian Saam, Rainer Saam, Maria Schmidt, Mirjam Simantzik, Matthias Sperber, Sebastian Stüker, Yury Titov, Joshua Winebarger, Matthias Wölfel and Liang Guo Zhang. The discussions with the research team from Mobile Technology (Christian Fügen, Thilo Köhler and Kay Rottmann), who shared their office with us, were very helpful.

Last but not least, I would like to thank my girlfriend Edith Steinle, who always supported me.

Contents

List of Figures **ix**

List of Tables **xi**

Glossary **xv**

1 Introduction **1**

1.1 Motivation 1

1.2 Domain Adaptation 4

1.3 Text Characteristics 5

1.4 Main Contributions 7

1.5 Outlook 8

2 Basic SMT System **11**

2.1 Preprocessing 11

2.2 Translation Model 12

2.3 Language Model 14

2.4 Reordering Model 14

2.5 Log-linear Model 16

2.6 Decoder 18

2.7 Optimization 18

3 Related Work **19**

3.1 Domain Adaptation in Machine Learning 19

3.2 Domain Adaptation in Statistical Machine Translation 21

3.3 Comparison of this Work to the Related Work 25

4 Application Scenarios **27**
4.1 Tasks . . . 27
4.2 Data . . . 28
4.2.1 Test Data . . . 28
4.2.2 Training Data . . . 28
4.2.3 Development Data . . . 30
4.3 Initial Experiments . . . 31
4.3.1 TED Translation Task . . . 31
4.3.1.1 Examples for the TED Translation Task . . . 33
4.3.2 Computer Science Lectures Translation Task . . . 33
4.3.2.1 Examples for the CS Translation Task . . . 36

5 Exploiting Mismatching Data **39**
5.1 Context Information in Phrase-based Machine Translation . . . 40
5.2 Bilingual Language Model . . . 43
5.2.1 Related Work . . . 45
5.2.2 Model . . . 46
5.2.3 Comparison to Tuples . . . 47
5.2.4 Comparison to Phrase Pairs . . . 48
5.2.5 POS-based Bilingual Language Models . . . 49
5.3 Results . . . 50
5.3.1 Speech Translation Tasks . . . 50
5.3.2 Additional Experiments . . . 55
5.3.2.1 German-English News Translation . . . 55
5.3.2.2 Arabic-English . . . 55

6 Adaptation by Combining In-Domain and Out-of-Domain Data **59**
6.1 Language Model Adaptation . . . 60
6.1.1 Linear Language Model Adaptation . . . 60
6.1.2 Log-linear Language Model Adaptation . . . 62
6.1.3 Results . . . 62
6.2 Translation Model Adaptation . . . 64
6.2.1 Problems Associated with Disregarding the Data Source . . . 64
6.2.2 Domain Adaptation using Back-off Probabilities . . . 65

6.2.2.1 Results . . . 67
6.2.3 Domain Adaptation using Factored Translations Models . . . 68
6.2.3.1 Domain Factors Translation Model . . . 70
6.2.3.2 Domain Factors Sequence Model . . . 71
6.2.3.3 Results . . . 72
6.3 Detailed Analysis and Combination of Different Translation Model Adaptation Strategies . . . 76
6.3.1 Additional Approaches to Domain Adaptation . . . 77
6.3.2 Candidate Selection . . . 78
6.3.3 Selecting Scores for the Phrase Table . . . 79
6.3.4 Results . . . 82
6.3.4.1 Candidate Selection . . . 82
6.3.4.2 Selecting Scores for the Phrase Table . . . 87
6.3.4.3 Combining In-Domain and Out-of-Domain Data . . . 93

7 Adapting towards Specific Topics 95
7.1 Motivation . . . 96
7.1.1 Out-of-Vocabulary Words . . . 97
7.2 Integration of Additional Knowledge Sources . . . 98
7.2.1 Wikipedia . . . 98
7.2.2 Related Work . . . 99
7.2.3 Lexicon Creation . . . 99
7.2.3.1 Disambiguation Pages . . . 100
7.2.3.2 Integration . . . 101
7.2.4 Article Disambiguation . . . 101
7.2.5 Results . . . 102
7.2.5.1 Integration of Wikipedia . . . 102
7.2.5.2 TF-IDF . . . 105
7.3 Quasi-Morphological Operations . . . 106
7.3.1 Motivation . . . 107
7.3.2 Related Work . . . 108
7.3.3 Approach . . . 108
7.3.4 Operations . . . 109

7.3.5 Features . . . 110
7.3.6 Training . . . 111
7.3.7 Integration of the Operations . . . 112
7.3.8 Results . . . 112
7.4 Summary . . . 114

8 Exploiting Genre Similarity of Training Data 117
8.1 Motivation for Genre Modeling . . . 118
8.2 Class-based Approaches . . . 120
8.2.1 Topic-dependent Words . . . 122
8.2.1.1 Class Definition based on OOV Words . . . 122
8.2.1.2 Class Definition based on TF-IDF . . . 123
8.2.2 Comparison to Class-based n-gram Models . . . 124
8.2.3 Results . . . 124
8.3 Restricted Boltzmann Machine-based Language Models . . . 127
8.3.1 Related Work . . . 128
8.3.2 Overview of Restricted Boltzmann Machines . . . 128
8.3.2.1 Layout . . . 129
8.3.2.2 Calculating Probabilities in RBMs . . . 130
8.3.2.3 Training . . . 131
8.3.3 Layout of the RBM for Language Modeling . . . 132
8.3.3.1 Word Factors . . . 132
8.3.4 Training of RBM-based Language Models . . . 133
8.3.5 Sentence Probability . . . 133
8.3.6 Results . . . 136
8.3.6.1 Network Layout . . . 140
8.3.6.2 Training Iterations . . . 141
8.3.6.3 Combination with Topic Adaptation . . . 141
8.3.6.4 RBMLM for English-French . . . 142
8.4 Summary . . . 143

9 Results Overview **145**
9.1 Speech Translation Task 145
9.1.1 Examples 150
9.2 Evaluation Campaigns 153
9.2.1 WMT 2012 153
9.2.2 Quaero 2012 154
9.2.3 IWSLT 2012 155

10 Conclusion and Outlook **157**

Appendices **161**

A Continuous Space Language Models using Restricted Boltzmann Machines **163**

B Evaluation Campaigns **165**
B.1 WMT 2012 165
B.2 Quaero 2012 167
B.3 IWSLT 2012 169

References **171**

List of Figures

5.1 Alternative segmentations into phrase pairs 41
5.2 Example of segmentation and bilingual tokens 45

6.1 Example of German-English translation with corpus ID factors 69
6.2 Adaptation approaches for candidate selection 79

7.1 Quasi-morphological operations 107

8.1 Layout of the Restricted Boltzman Machine-based language model . . . 132

List of Tables

4.1 Overview of the test data 29
4.2 Overview of the parallel training data 30
4.3 Overview of the development data 31
4.4 Initial experiments on the TED task 31
4.5 Initial experiments on the CS task 34

5.1 Average phrase pair length of the initial system 42
5.2 Different phrase pair lengths of the initial system 42
5.3 Average phrase pair length for the TED task 43
5.4 Average phrase pair length for the CS task 43
5.5 Overview of results for bilingual language model on the TED task . . . 51
5.6 Overview of results for bilingual language model on the CS task 51
5.7 Context used by the bilingual language model 53
5.8 Different n-gram lengths of the bilingual language model tested on the CS task 54
5.9 German-English news translation task using bilingual language model . 55
5.10 Arabic-English translation task using bilingual language model 56
5.11 Bilingual context used in the Arabic-English translation 57

6.1 Language model adaptation on the TED task 63
6.2 Language model adaptation on the CS task 63
6.3 Phrase table adaptation using back-off approach on the TED task . . . 67
6.4 Phrase table adaptation using back-off approach on the CS task 68
6.5 Domain Factor Sequence Models 74
6.6 Domain Factor Translation Models for the TED task 75

6.7 Domain Factor Translation Models for the CS task (Dev on TED) . . . 75
6.8 Domain Factor Translation Models for the CS task (Dev on CS) 76
6.9 Domain adaptation using Factored Translation Models on the NC task . 76
6.10 Characteristics of the different phrase table adaptation approaches . . . 82
6.11 Number of phrase pairs generated by different candidate selection methods 83
6.12 Candidate selection for the TED task (No optimization) 84
6.13 Candidate selection for the CS task (No optimization) 85
6.14 Candidate selection for the TED task 85
6.15 Candidate selection for the CS task (Dev on TED) 86
6.16 Candidate selection for the CS task (Dev on CS) 86
6.17 Feature selection on the TED task 88
6.18 Feature selection on the CS task (Dev on TED) 88
6.19 Feature selection on the CS task (Dev on CS) 89
6.20 Feature combination for the TED task 91
6.21 Feature combination for the CS task (Dev on TED) 92
6.22 Feature combination for the CS task (Dev on CS) 92
6.23 Overview of results using data weighting 93

7.1 Out-of-vocabulary words of the different tasks 97
7.2 Integration of Wikipedia data 103
7.3 Different integration strategies on the TED task 105
7.4 Different integration strategies on the CS task 105
7.5 Wikipedia integration using document similarity 106
7.6 Features for quasi-morphological operations 111
7.7 OOV words for quasi-morphological operations 113
7.8 Overview of results using quasi-morphological operations 113

8.1 Analysis of the different language models on the TED task 119
8.2 Analysis of the different language models on the CS task 119
8.3 Analysis of the class-based language models for the TED task 124
8.4 Analysis of the class-based language models for the CS task 125
8.5 Class-based language models on the TED task 125
8.6 Class-based language models on the CS task 126
8.7 Overview on RBM-based language models 137

8.8 Number of hidden units for the TED task 139
8.9 Number of hidden units for the CS task 139
8.10 Number of training iterations for the TED task 140
8.11 Combining RBM-based language models with topic adaptation 142
8.12 RBM-based language model on the English-French task 143

9.1 Overview on all scenarios without matching training data 146
9.2 Overview on all scenarios using TED training data 148
9.3 Contributions to WMT 2012 results 153
9.4 Contributions to Quaero 2012 results 155
9.5 Contributions to IWSLT 2012 results 156

A.1 Comparison of language models based on RBMs and feed-forward networks 163

B.1 German-English results for WMT 2012 165
B.2 English-German results for WMT 2012 165
B.3 English-French results for WMT 2012 166
B.4 French-English results for WMT 2012 166
B.5 German-French results for Quaero text evaluation 2012 167
B.6 French-German results for Quaero text evaluation 2012 167
B.7 German-French results for Quaero speech translation evaluation 2012 . . 168
B.8 French-German results for Quaero speech translation evaluation 2012 . . 168
B.9 English-French results for IWSLT 2012 MT task 169
B.10 English-French results for IWSLT 2013 SLT task 169

Glossary

ASR Automatic Speech Recognition

BiLM Bilingual Language Model, model in a phrase-based machine translation system to increase the bilingual context

BLEU Bilingual Evaluation Understudy, the most widely used evaluation metric for machine translation (Papineni et al., 2002)

BTEC Basic Travel Expression Corpus

CD Contrastive Divergence, training algorithm for RBMs

CS Computer Science, test set containing computer science lectures used as one of the main translation tasks in this thesis

Dev Development (Set), data that is used to train the log-linear model

EPPS European Parliament Plenary Sessions, the largest available parallel corpus for most European languages

HMM Hidden Markov Model

IBM1 IBM Model 1

IWSLT International Workshop on Spoken Language Translation

KIT Karlsruhe Institute of Technology

LM Language Model

LSA Latent Semantic Analysis

MERT Minimum Error Rate Training, most commonly used optimization method for phrase-based machine translation

MKCLS Word cluster algorithm

ML Machine learning

MLP Multi-Layer Perceptron, feed-forward neural network

MT Machine Translation

NC News Commentary, parallel corpus of several European languages

NER Named Entity Recognition, task of automatically marking names in a text

NLP Natural Language Processing, field in the area of computer science and computer linguistics that concentrates on processing natural language data

OOV Out-of-Vocabulary (word)

PL Phrase Length, length of the phrase pairs used to build the translation

POS Part-of-Speech, classes describing the function of the word in the sentence

PP Phrase Pair, basic translation unit in a phrase-based machine translation system

RBM Restricted Boltzman Machine, stochastic neural network

RBMLM RBM-based Language Model

SCL Structural Correspondence Learning, domain adaptation approach presented by (Blitzer, 2008)

SLT Spoken Language Translation

SMT Statistical Machine Translation

SRILM SRI Language Model, most commonly used language model toolkit

SVM Support Vector Machine, supervised model for machine learning

GLOSSARY

TED Technology, Entertainment, Design, global conference, whoes talks are transcribed and translated into many languages.

TER Translation Error Rate, evaluation metric for machine translation

TF-IDF Term Frequency - Inverse Document Frequency, common measure in information retrieval for the importance of a word

TM Translation Model

WMT Workshop on Machine Translation

1

Introduction

Due to progress in statistical machine translation (SMT), in recent years machine translation has gained increasing attention and has been used in more and more real world scenarios[1, 2]. While very good machine translation (MT) systems exist for a few domains, it is still difficult to adapt a machine translation system to a new domain.

In this thesis we developed different methods to improve the adaptation of a statistical machine translation system for a new application. The problems analyzed in this thesis were driven by the application of machine translation in the scenario of translating lectures, but the resulting approaches where tested in several conditions and helped to improve translation systems in several international evaluations (Callison-Burch et al., 2012; Federico et al., 2012). Before investigating the problem of adapting an MT system to a new task in the next chapters, we will first give a motivation of the problem. Afterwards, we will give a short introduction to domain adaptation. In Section 1.3, we introduce text characteristics that will be used to categorize the training and test data. Afterwards, we will summarize the main contributions of this thesis and give an outlook over the chapters of this thesis.

1.1 Motivation

We live in a more and more globalized world. A few decades ago most people had only few opportunities to communicate with people speaking different languages. In contrast, nowadays many people are able to travel to countries where they are not able

[1] http://translate.google.de/
[2] http://www.bing.com/translator

to speak the local languages. Furthermore, the internet provides access to information written in many different languages.

Due to this development the language barrier becomes a much more important problem for many people. Especially if we look at a university environment, we can find many examples of problems due to different languages. Many German students spend part of their studies abroad. Also many students from abroad come to Germany to continue their studies there. Although most of them are able to speak English and at least some words in the language of the country they are visiting, many problems exist in university and everyday life.

At the Karlsruhe Institute of Technology (KIT) most of the courses are held in German. As a result, in addition to the complex content of the lecture, foreign students have also problems following the lecture due to problems understanding the German language. To make the stay for foreign students more attractive it is therefore necessary to give them support in understanding the lectures. Therefore, we do not only need to translate the lectures themselves, but also the additional material available in class.

The internet provides even more resources that are only available in one or a few languages. Nowadays, videos of many university courses offer an easy access to the knowledge for many people around the world. But of course, only students who understand the teaching language are able to follow those lectures.

The TED website[1] sets an example of what is necessary to reach a broad audience. TED talks are available online, together with transcripts and translations generated by a large community. But with the enormous growth in resources available in the internet, it is not possible to do this for all data. In the European Parliament the speeches are even translated simultaneously. For university lectures we would also like to provide simultaneous translations. But this is even more expensive and therefore, universities cannot afford to pay interpreters for translating all lectures simultaneously. This problem can only be addressed by using automatic translation.

Advances in statistical machine translation in the last 20 years make it possible to build new translation systems quite fast while at the same time achieving a reasonable translation quality. The SMT approach no longer needs expert knowledge to create a new translation system, but the system is trained from huge amounts of existing

[1] http://www.ted.com/

translations using machine learning approaches. Currently, systems translating between more than 70 languages[1] are available.

The use of parallel data, instead of expert knowledge, is a great advantage when building a system. But the problem of getting appropriate parallel data still remains. Since it is not only necessary to have large amounts of parallel data, but this data also needs to match the application. In order to be able to build new systems for different applications, we need strategies to adapt existing systems to the task at hand.

Only for very few tasks, matching training data for the given application is available. This is, for example, the case for translating parliament proceedings or newspaper articles. But for many other tasks this is not the case. Regarding the task of translating university lectures, only a very small amount of matching data is available.

In order to build a translation system with a reasonable translation quality, we first need to be able to better learn from non-matching data. If we look at the task of translating TED lectures, there is, for example, the German phrase *vor 60000 Jahren* (engl. *60000 years ago*). This phrase seems not to be very domain specific, but it was wrongly translated by the baseline system to *before 60000 years*. In contrast, a very small system trained only on the in-domain data was able to translate the sentence correctly.

Additional problems arise in the scenarios where small amounts of matching data are available. In these cases we need to improve our ability to learn from a mixture of matching and non-matching training data. If this is not done correctly, we will often use translations learned from the largest training domain. In the test domain this can lead to wrong translations. An example can be found in the following sentence: *Wir wollen uns das bestmögliche Ergebnis vorstellen.* (engl. *We want to imagine the best case scenario outcome*). The baseline system translated *vorstellen* as *present* instead of *imagine*, since this is more common in speeches in the European Parliament. Another example can be found in the phrase *der Baupläne des Schiffs*, where the word *Schiffs* is translated as *vessel* while the better translation in this sentence would be *ship*.

Furthermore, the training data might be only partially suitable for the applications. If we build a translation system for computer science lectures, we might have data from lectures about mechanical engineering. This data is from the same genre and might

[1] http://translate.google.de/

therefore be very helpful. On the other hand, the data is about a completely different topic. Therefore, using this data might even harm the translation quality.

On the other hand, it might be useful to use research papers about computer science. This data is about the same topic, but the genre is completely different. Therefore, the wording used in the research paper is quite different from the ones in the lectures. But data about the same topic might help us to translate topic-specific words like *Perzeptron* (engl. *preceptron*), *Kegelschnitt* (engl. *cone cut*), *Rotations-Ellipsoid* (engl. *rotation ellipsoid*)and *Torus* (engl. *torus*) which could not be translated correctly by our baseline system. Therefore, we need to improve our ability to make the best use of this data.

1.2 Domain Adaptation

Adaptation techniques have been applied in speech recognition and language modeling for a long time. More recently, domain adaptation for different machine learning tasks in natural language processing has gained a lot of attention (Daumé III and Marcu, 2006).

In many natural language processing tasks, we have some input data $x \in \mathcal{X}$ and need to predict the corresponding output $y \in \mathcal{Y}$. This is the case for machine translation, where we need to predict the translation given the source sentence, but also for tagging, parsing and many other natural language processing tasks. In all these tasks statistical approaches, which try to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, have been proven to be a good approach. In order to train statistical models, some training data $\mathcal{D} : \{(x_n, y_n) \in \mathcal{X} \times \mathcal{Y}\}$ is used. These training examples are assumed to be independent and identically distributed, given some unknown distribution p. Then the model is trained to approximate y_n by $f(x_n)$ for all $(x_n, y_n) \in \mathcal{D}$. Afterwards the model is applied to some test data $\mathcal{D}^A$ drawn from the same distribution p.

The problem is that this assumption is not true for many real-world applications. In contrast, the test data $\mathcal{D}^A$ is typically selected from a different domain and therefore drawn from a different distribution p^A. While the model might approximate y_n well by $f(x_n)$ for $(x_n, y_n) \in \mathcal{D}$, this is often no longer true for samples from the test data $\mathcal{D}^A$. If the distribution or domain of the test data and training data differ too much,

the approximation will often no longer lead to satisfying results. In this case, we need to adapt the translation system to the domain and distribution of the test data.

A first direction of research, when adapting the models to a new system, is to develop models that better generalize in order also to match data drawn from other distributions. These models may be as good on predicting y_n for $(x_n, y_n) \in \mathcal{D}$ or even a little worse, but will perform better when predicting samples from $\mathcal{D}^A$.

A second research direction is to improve the modeling of the training data. The assumption that all the training data is drawn from the same distribution is quite a rough approximation. A better way to model it is to use a set of training data sets $\mathcal{D}_i$ with $\mathcal{D} = \cup\mathcal{D}_i$, where each one is drawn from a distribution p_i. The most common way to do this is to use two sets of training data, an in-domain and an out-of-domain set. In this case, we can use this additional information about the source of the data to build a model that better approximates the samples drawn from the distribution p^A.

The methods described above mostly rely on the fact that some distributions that are used to draw parts of the training data are more similar to the distribution of the test data than others. One problem in such a complex task as machine translation is that is not straight forward to define whether two domains or distributions are similar or not. For machine translation, the similarity of the distributions means the similarity of the input text.

When comparing texts, several dimensions of similarity can be distinguished. Two texts can be about the same topic, but from different genres like a lecture transcript and a web page about that topic. On the other hand we can have two university lectures, one about computer science and the other one about art. In both cases the domains of the two texts will be very different, but they share some characteristics.

Therefore, it might not only important to model how similar the source of the training data is to the test case, but also to model in which characteristic or dimension they are different and in which way they are similar.

1.3 Text Characteristics

A text can be characterized by several properties and these text characteristics can be used to categorize different texts. As mentioned before, we will try to use these charac-

teristics to adapt the system using data that matches only in some text characteristic. In this thesis we concentrated on two main characteristics.

First, texts can be characterized by the topic. The topic of the text describes what the text is about. So this characteristic focuses on the content of the text. A human is directly able to determine the topic given a text. But there can be topic switches in a text and the text can be about different topics. Furthermore, there is a hierarchy of topics. We can use broader topics like computer science or more specific ones like speech recognition.

The topic or theme of a text has gained much attention in the natural language processing community in the context of probabilistic topic models. These approaches try to determine the topic or a distribution of topics for a given text. In most of these approaches like, for example, in the most common one called Latent Dirichlet Allocation (LDA) (Blei et al., 2003), the topic is represented by a distribution over the words of the vocabulary. In recent years several extensions and additional approaches were presented (Blei et al., 2010; Shu et al., 2009). Most of them model the topic mainly by the distribution of words. Therefore, the topic seems to mainly influence the word choices in a given text.

The second dimension in which different texts can be compared is the genre or type of text. Initial work in the field of analyzing the type of text was undertaken by Biber (1991). He distinguishes between the genre and the text type. According to this work, the genre is defined primarily based on external criteria. For example, "newspaper articles are found in the news section of newspapers"(Biber, 1989). In contrast, the text types are defined by analyzing the syntactic and lexical features that occur in the text.

Biber (1991) analyzed the language structure and phonemes occurring in a large variety of different text corpora. By performing a statistical analysis he discovered five main dimensions to characterize the type of text. The first one is "involved versus information production". As described in Biber (1993) "involved production" can be conversations like personal letters or public conversations, while examples for "information productions" are official documents and academic prose.

The five dimensions presented in Biber (1991) where used in Biber (1993) to determine eight major types of English texts. These text types were automatically found

using a clustering algorithm. Although there is a strong correlation between genre and text type in the works of Biber, there is no one-to-one mapping between them.

More recently, with availability of even larger corpora, the automatic detection of genre has received even more attention (Kessler et al., 1997; Petrenz, 2009). In these works, the authors do not define the genre based on external features only, but combine internal and external features. A definition based on both features can e.g. be found in Kessler et al. (1997): "We will use the term genre here to refer to any widely recognized class of text defined by some common communicative purpose or other functional traits, provided the function is connected to some formal cues or commonalities and that the class is extensible".

It is worth mentioning that the concepts of topic and genre are orthogonal. But, of course, covariances do exist. For example, in the genre of lectures there will be quite a lot of examples of the topic mathematics, while this will be a quite rare topic in news articles.

1.4 Main Contributions

In this thesis we mainly investigated four different problems that arise when porting a machine translation system to a new task. Since in most scenarios, large amount of mismatching training data are available, the first contribution of this work concentrated on the following question:

1. *Why does mismatching training data lead to lower performance in machine translation and how can we exploit this data optimally?*

In order to answer this question, we analyzed in how the translations of matching and mismatching test sentences are generated. We show that one problem when translating mismatching data is that the selection of the translation can only be made based on less context. Therefore, we propose to use an additional model in phrase-based machine translation to increase the available context.

In many real-world scenarios small amounts of matching in-domain data are available in addition to the mismatching training data. Therefore, the next focus point is:

2. *How can we combine in-domain and out-of-domain information?*

In order to improve the combination of in-domain and out-of-domain information, we present two approaches of combining translation models trained on different data sets. Furthermore, we analyzed what aspects of the translation model need to be adapted and how large the influence of the adaptation is.

As mentioned before, text can be categorized using different criteria. In a first step, we looked at the topic of the text.

3. *What problems arise if we change the topic of the task and how is it influenced by the specificity of the topic?*

In this part of the work, we analyzed why the translation system performance gets worse when translating a text about a new topic. We will compare a task with a very specific topic like university lectures to a task with a more general topic like the translation of TED talks. Furthermore, we will present an approach to better handle out-of-vocabulary (OOV) words that occur if we need to translate input with a very specific topic.

Finally, we concentrate of a different criterion, the genre. Most of the available parallel corpora, like the EPPS corpus (Koehn, 2005), News commentary corpus, BTEC corpus or the TED corpus, contain sentences from the same genre, but with different topics. If we need to adapt the translation system towards a new domain, the genre of this domain might be the same as the one of one of these corpora or more similar to one of the training corpora genres. The question that arises in this case is:

4. *How can we exploit genre similarity of the data in the best way?*

In order to improve the modeling of the genre similarity, we propose two language models that generalizes better than a standard n-gram language model when many topic-specific terms occur.

1.5 Outlook

In Chapter 2, we will give a short overview on SMT systems. In the description we will concentrate on those parts of an SMT system, which are relevant for the experiments described in the later chapters. In the following chapter we will review related work

on the topic of domain adaptation in the context of machine learning and statistical machine translation.

In Chapter 4 we describe the application scenario, in which we tested the proposed methods. We will describe the characteristics of the task and the data. Furthermore, we perform initial experiments using baseline systems.

Afterwards, we describe the problems of generalizing an MT system in order to match many different domains. We will analyze why our baseline model does not perform as well when applying it to a different domain. In order to improve the generalization, we will present the bilingual language model. We show that this model increases the context considered during translation and also, the translation quality especially when using non-matching data. This way we are able to better generalize the translation system.

As mentioned in Section 1.2, another direction of research in domain adaptation is to include the source of the data into the modeling process. In Chapter 6, we explore various ways to combine matching and non-matching training data. Therefore, we review language model combination approaches for the task of lecture translation. Furthermore, we introduce two new approaches for translation model combination. One approach is modeled by a log-linear combination with back-off to a default value, while the other uses the framework of factored translation models. We compare the impact of these approaches and other related approaches on the translation quality.

In Chapter 7, we try to adapt an MT system to a topic using data that only matches in topic, but not in genre. Using the example of Wikipedia article titles we show how data from a completely different genre can be used to help translate topic-specific terms.

Finally, in Chapter 8, we look at the other dimension of text and analyze how to adapt to the genre of lectures. Therefore, it is often not possible to use data that also matches in topic. In a lecture translation system we are using a new approach for continuous space language models. With this approach we are able to make better use of the TED data, which is homogeneous in genre, but covers different topics.

Finally, we present an overview of the results achieved by applying the proposed methods in different scenarios. In the end we will give a conclusion of the work.

2

Basic SMT System

In this chapter we will describe the basic statistical machine translation system that is used in this thesis as a baseline system. This system is further extended to improve its capability to adapt to new translation tasks.

In order to describe the translation system we will introduce the different models that compose the translation system. First, we will describe the preprocessing that we apply to the training data as well as the test and development data. In addition to the general preprocessing, we apply special German preprocessing to handle several characteristics of the German language. Furthermore, we will describe the translation model that is used in the SMT system. Afterwards, the two other models used in the baseline translation model, the language model and the reordering model, will be presented.

After describing the different models we will present the log-linear model used to combine these models into one approach. Finally, the decoder as well as the optimization procedure used in this thesis will be introduced.

2.1 Preprocessing

In a first step of the training process, training data is preprocessed. This includes normalizing special symbols, smart-casing the first word of each sentence and removing long sentences and sentences with length mismatch. For the German language, we also map words written according to the old orthographic rules to their new writing form. Furthermore, if we use German as the source language, we apply compound splitting to

the training and test data to be able to translate unseen German compounds also. For this purpose, we used the count-based method proposed in Koehn and Knight (2003).

2.2 Translation Model

After the preprocessing, the discriminative word alignment approach as described in Niehues and Vogel (2008) was applied to generate the alignments between source and target words. For this, we need to build the generative word alignment using the IBM-4 model using the parallel implementation of GIZA++ (Gao and Vogel, 2008). This information and POS information from the TreeTagger (Schmid, 1994) in both languages are used as features in the discriminative word alignment model. Furthermore, features to model the fertility as well as features to model first-order dependencies are used.

Afterwards, the phrase table is built using the scripts from the Moses package (Koehn et al., 2007). In a first step, the phrase pairs are extracted from parallel data using the word alignment information. Therefore, a phrase pair is extracted, if it does not violate the word alignment. A phrase pair $(\bar{e}, \bar{f})$ with $\bar{e} = e_{i_{start}} \dots e_{i_{end}}$ and $\bar{f} = f_{j_{start}} \dots f_{j_{end}}$ violates the word alignment if a word outside the phrase pair is aligned to a word inside the phrase pair. Given the source sentence $f = f_1 \dots f_J$, the target sentence $e = e_1 \dots e_I$ and the alignment $A = \{(j, i)\} \subset \{1 \dots J\} \times \{1 \dots I\}$, this is formalized by:

$$\begin{aligned} (\bar{e}, \bar{f}) \text{ consistent with } A \leftrightarrow \\ \forall e_i \in \bar{e} : (e_i, f_j) \in A \rightarrow f_j \in \bar{f} \\ \text{AND } \forall f_j \in \bar{f} : (e_i, f_j) \in A \rightarrow e_i \in \bar{e} \\ \text{AND } \exists e_i \in \bar{e}, f_j \in \bar{f} : (e_i, f_j) \in A \end{aligned} \tag{2.1}$$

After extracting the phrase pairs from the parallel corpus, the scripts calculate four scores to be able to evaluate the quality of the phrase pair. First, we use the relative frequency of the phrase pair given the source phrase as well as the inverse relative frequency of the phrase pair given the target phrase. These probabilities can be calculated as:

$$p(\bar{f}|\bar{e}) = \frac{count(\bar{f}, \bar{e})}{\sum_{\bar{f}} count(\bar{f}, \bar{e})} \tag{2.2}$$

$$p(\bar{e}|\bar{f}) = \frac{count(\bar{f}, \bar{e})}{\sum_{\bar{e}} count(\bar{f}, \bar{e})} \tag{2.3}$$

Although these features are often very good to estimate the quality of a phrase pair, their prediction quality is no longer good enough if the source or target phrase only occurs quite rarely. In the extreme case, where the source phrase does only occur once, the first probability will always be one. Therefore, we use two approaches to tackle the problem of having many phrase pairs which only occur rarely. First, we follow Foster et al. (2006) not to use the relative frequencies themselves as features, but the smoothed version. We used a fix-discounting method originally proposed for language model smoothing by Kneser and Ney (1995). In our experiments, we used the modified version of it using interpolation proposed by Chen and Goodman (1996) . In this case the probability is defined as:

$$p(\bar{f}|\bar{e}) = \frac{count(\bar{f},\bar{e}) - D_{count(\bar{f},\bar{e})}}{\sum_{\bar{f}} count(\bar{f},\bar{e})} + \alpha(\bar{e}) D_{count(\bar{f},\bar{e})} p_b(\bar{f},\bar{e}) \tag{2.4}$$

In the equation $\alpha(\bar{e})$ is defined as

$$\alpha(\bar{e}) = \frac{n_{1+}(*,\bar{e})}{\sum_{\bar{f}} count(\bar{f},\bar{e})} \tag{2.5}$$

where $n_{1+}(*,\bar{e})$ is the number of phrase pairs with the target side $\bar{e}$ that occur at least once in the corpus. Furthermore, the smoothing distribution $p_b(\bar{f},\bar{e})$ is defined as:

$$p_b(\bar{f},\bar{e}) = \frac{n_{1+}(\bar{f},*)}{\sum_{\bar{f}} n_{1+}(\bar{f},*)} \tag{2.6}$$

In this equation $n_{1+}(\bar{f},*)$ is defined analogically to $n_{1+}(*,\bar{e})$. The discounting value $D_{count(\bar{f},\bar{e})}$ is defined the same way as the one of the language models in Chen and Goodman (1996). The probabilities $p(\bar{e}|\bar{f})$ in the inverse direction are defined analogically.

Furthermore, we can use the lexical probabilities additionally to approximate the translation probability better. They have the advantage that they are based on word occurrences compared to the previously mentioned ones, which are based on phrase occurrences. Individual words occur more often than phrases. These lexical probabilities are again used in both directions. Given the alignment inside the phrase pair

$a = \{(j,i)|(j,i) \in A \wedge j_{start} \leq j \leq j_{end} \wedge i_{start} \leq i \leq i_{end}\}$, they are defined as:

$$p(\bar{f}|\bar{e},a) = \prod_{j=j_{start}}^{j_{end}} \frac{1}{|\{i|(j,i) \in a\}|} \sum_{i:(j,i)\in a} P(f_j|e_i) \tag{2.7}$$

$$p(\bar{e}|\bar{f},a) = \prod_{i=i_{start}}^{i_{end}} \frac{1}{|\{j|(j,i) \in a\}|} \sum_{j:(j,i)\in a} P(e_i|f_j) \tag{2.8}$$

For several source phrases there are many different translations, which we cannot all consider during decoding due to runtime limitations. Furthermore, most of them have low probabilities and will not lead to good translations of the sentence. Therefore, we limit the number of translations $T(\bar{f}_i)$ for every source phrase $\bar{f}_i$ to a maximum of n by a combination of histogram and beam pruning. We used at most 10 translations for every source phrase. We rank the phrase pairs in order to be able to select the top translations by scoring them using some initial weights that were determined heuristically. These weights are independent from the weights generated by MERT.

2.3 Language Model

To model the target language, we trained a language model on the target side of the parallel data. If not stated differently, we used a 4-gram language model. The n-gram probabilities for the language model were trained using the SRILM toolkit (Stolcke, 2002). We used also modified Kneser-Ney smoothing for the language model probabilities.

2.4 Reordering Model

In our translation systems we use a reordering model that performs the reordering in a preprocessing step. During training, we first learn rules how to reorder the source sentence in a way that its structure is similar to the one of the target sentence. In a preprocessing step prior to the translation of the sentence, different reordering variants of the source sentence using these rules are encoded into a lattice. Afterwards this lattice is used as input to the decoder. The decoder can then translate the lattice from left to right.

To train the reordering model we use parallel data together with the automatically generated word alignment for the translation model. In addition, we generate the part-of-speech (POS) tags for the source sentences using the TreeTagger (Schmid, 1994). As described in Rottmann and Vogel (2007), we learn rules how to reorder the source sentence to generate a monotone alignment. In order to extract these rules, we check every source sequence in the corpus up to a given length for crossing alignment. When a crossing alignment occurs, it means that the source words need to be shuffled in order to translate them in monotone order into the target sequence. In this case, we extract the source sequence and the reordered sequence that leads to an monotone alignment. We then extract different types of rules from these sequences. We use the words themselves as a rule as well as the sequence, where the words are replaced by POS-tags. By doing so, we are able to generalize new sentences better. In addition, we use different combinations of words and POS-tags.

In the experiments we consider sequences of a length up to ten and only store rules that occur at least five times. To evaluate the different rules, we calculate their relative frequency.

Since we are forced to limit the reordering sequence by a length of ten, we are only able to model short-range and mid-range reorderings with these reordering rules. If German is one of the languages we use during translation, often not only short- and mid-range reorderings are necessary, but also long-range reorderings. In contrast to English or French, in German the verb is at the end of a sub-clause and even in the main clause, often the verb consists of two parts. In this case, the second part of the verb is also at the end of the sentence, while the verb in an English sentence is generally located at the second position in a sentence. Therefore, in Niehues and Kolss (2009) the reordering approach was extended to cover long-range reorderings by allowing gaps in the reordering rules.

In the reordering step prior to the translation, a lattice containing different word orders is generated. To build lattices we start with the original source sentence. If a reordering rule can be applied to the source sentence and its probability exceeds the threshold, a path with the reordering word sequence according to the rule is added to the lattice. In this case the first edge of the path is weighted by the relative frequency of the rule and the following edges are assigned a weight of one. If the same reordering can be generated by several rules, only one path with the highest weight of all rules

is added. In our experiments we use a threshold of 0.2 for the short-range reordering rules and a threshold of 0.05 for the long-range reordering rules. In addition, if the same long-range reordering rule can be applied to a sentence more than five times, it is not considered helpful for this sentence and ignored.

After the additional edges are added to the lattice, the weight of the edges in the monotone path is determined. Therefore, we calculate the maximum probability of all reordering rules starting with the word of the edge. The weight of the monotone edges is set as subtraction of the maximum from one.

2.5 Log-linear Model

The different models described in the previous parts are all useful to discriminate between good and bad translations. In our translation system, as in all state-of-the-art statistical machine translation approaches, these models are combined using a log-linear model. The log-linear model defines a probability for a target sentence given the source sentence by:

$$p(e|f) = \frac{1}{Z} exp\left(\sum_{g \in features} \lambda_g g(e,f)\right) \tag{2.9}$$

where the normalizing factored Z is defined as:

$$Z = \sum_{e} exp\left(\sum_{g \in features} \lambda_g g(e,f)\right) \tag{2.10}$$

In this case the first sum iterates over all possible translations e of the source sentence f.

In our baseline system we use nine different feature functions in the log-linear model. First, we have the four different features of the translation model. As described in Section 2.2, we calculate four different scores for every phrase pair. Each of these scores is used as a feature for the whole sentence in the following way:

$$g^{TM}(e,f) = \sum_{i=1}^{\bar{I}} log(p(\bar{e}_i|\bar{f}_i)) \tag{2.11}$$

where the translation of f in to e is generated using the $\bar{I}$ phrase pairs $((\bar{f}_1, \bar{e}_1), (\bar{f}_2, \bar{e}_2),$ .., $(\bar{f}_{\bar{I}}, \bar{e}_{\bar{I}}))$.

Secondly, we use the language model probability as a feature in the log-linear model. This leads to the following feature:

$$g^{LM}(e,f) = \sum_{j=1}^{J} log(p(e_j|e_{j-1},\ldots,e_{j-(n-1)})) \tag{2.12}$$

In this equation n is the length of the context used in the language model.

Furthermore, we use two features to model reordering. First, we use the standard distance-based reordering model used in nearly all current machine translation systems. This is defined as:

$$g^{DRO}(e,f) = \sum_{i=1}^{\bar{I}-1} \frac{1}{4}(sourceStart_{i+1} - sourceEnd_i - 1)^2 \tag{2.13}$$

In this equation $sourceStart_i$ is the index of the first source word of the ith phrase pair and $sourceEnd_i$ is the index of the last source word. In contrast to other approaches, we do not use the original source index of the word, but the index according to the current path in the lattice that we translate. In addition to this model, we also use the weights of the edges in the lattice as an additional feature in the log-linear model. This feature is defined as

$$g^{LRO}(e,f) = \sum_{i=1}^{I} log(w(e_i)) \tag{2.14}$$

where $w(e_i)$ is the weight of the edge of the source word e_i.

Finally, we use two more count features in the log-linear model. First, we use the word count model, the value which represents the number of target words in the sentence. This feature makes it possible to compare translations of different length in a fair way. If we look at the language model feature, we see that its value will always decrease if we increase the length of the sentence, since $p(e_j|e_{j-1},\ldots,e_{j-(n-1)})$ is a probability smaller than one. Therefore, we use the word count feature to prefer longer sentence.

The second count feature is the phrase count feature. Its value is the number of phrase pairs used in the translation. One problem in phrase-based machine translation is that hypotheses produced using different segmentations compete against each other. On the one hand longer phrase pairs have the advantage of including more context information compared to shorter phrase pairs. On the other hand, short phrase pairs

occur more often and therefore their probabilities can be estimated more reliably. One approach to model this is to use the phrase count feature. Its weight can be used to prefer longer or short phrase pairs.

2.6 Decoder

The translation of a source sentence is generated using an in-house phrase-based decoder as described in Vogel (2003). This decoder is capable of taking the reordering lattice as input as described in Section 2.4. In the decoder, a search of a possible translation is performed and the best hypothesis according to the log-linear model is generated

2.7 Optimization

Finally, the weights of the log-linear model need to be optimized. For this, we translate the development data with the decoder and generate an n-best list. In our experiments, we usually use a 300-best list. Then we use the minimum error rate training (MERT) as described in Venugopal et al. (2005) to find the parameters that lead to the best BLEU score (Papineni et al., 2002). This is repeated for several iterations. In the experiments we normally use 20 iterations. These optimized weights are used for translating the test data.

3

Related Work

In recent years, strategies to perform domain adaptation in the field of natural language processing (NLP) gained an increasing interest in the research community. We will first review work that has been done in general on domain adaptation in the machine learning community.

Afterwards, we will focus on the related work in the area of domain adaptation for statistical machine translation. This work is often motivated by the ideas used in other NLP tasks, but often the approaches cannot be applied directly since machine translation is not a classification task like most of the other NLP tasks (Carpuat et al., 2012). Work in the area of machine translation that is specific to the techniques introduced in later chapters will be reviewed in these chapters.

3.1 Domain Adaptation in Machine Learning

In many real-world scenarios of tasks considered in machine learning such as named entity recognition (NER), spam filtering or recapitalization, the assumption that the training data and test data have the same distribution does not hold. Therefore, domain adaptation which tries to model the situation where the distribution of the training and test data is different has gained an increasing interest (Blitzer, 2008; Jiang, 2008).

The approaches presented in the literature can be divided into two main scenarios. In the first scenario, no labeled data from the target domain is available. In this case, only unsupervised domain adaptation can be used to adapt the models to the target domain.

In the second scenario labeled data in the target domain is also available. Therefore, we can use supervised approaches which can leverage the fact that the data from the target domain is more important than the data from the source domain.

An additional strategy to train a classifier, the semi-supervised learning (Chapelle et al., 2006; Zhu, 2006), has been shown to be very useful. In this approach small amounts of label data and large amounts of unlabeled data are used to train a model. This situation is very similar to many domain adaptation scenarios, where we also often have large amounts of out-of-domain data and small amounts of labeled or unlabeled in-domain data. Therefore, this technique has been adapted to the domain adaptation scenario. Dai et al. (2007a) used an EM-based algorithm for domain adaptation and Xing et al. (2007) used a bridged refinement method. Jiang and Zhai (2007a) extended an instant weighting approach to the case where there is labeled and unlabeled data. They showed improvements on the tasks of POS-tagging, NE recognition and spam classification.

If labeled data from the target domain is available, one method to make use of the source and target domain data is to use Bayesian priors. In this approach, the source domain data is used to model priors on the parameters of the model. Then the model is trained on the target domain data by also including these priors. Li and Bilmes (2007) proposed a general Bayesian divergence prior. They show results on the vowel classification and object recognition task using a support vector machine (SVM) and a multi-layer perceptron (MLP). Chelba and Acero (2006) used Bayesian priors in a maximum entropy classifier on the task of capitalizing text.

A different strategy to domain adaptation is to augment the features used in the classifier. In Daumé III (2007) every feature is replaced by a general, a source-domain and a target-domain specific version. The author evaluated this method on different NLP tasks, like NE recognition, POS-tagging, recapitalization and shallow parsing. Jiang and Zhai (2007b) presented a two step approach. They first select features that generalize well. In the second step, features that are useful for the target domain are selected.

Since we normally have only a small amount of in-domain data, but a lot of out-of-domain data, a different strategy is to use mixture models. An approach to include mixture-models for performing domain adaptation in the maximum entropy framework

is presented in Daumé III and Marcu (2006). They evaluated their method on mention type classification and tagging as well as on the recapitalization task.

Finally, Dai et al. (2007b) extends boosting-based learning algorithms to perform domain adaptation. This algorithm is evaluated on the task of text classification.

In the second scenario for domain adaptation in machine learning no labeled data for the target domain is available. In this case the model has to be adapted using only unlabeled training data from the target domain.

One strategy to make use of this data is to re-weight the labeled training examples from the source domain using statistics from the unlabeled target domain data. Lin et al. (2002) analyzed this instant weighting approach for SVM classifiers.

A second approach is to use bootstrapping. In this case, a model trained on the labeled data is used to label the unlabeled data. McClosky et al. (2006) used self-training to adapt a parser to the target domain. Steedman et al. (2003) used co-training introduced by Blum and Mitchell (1998) to adapt a parser to the new domain.

Finally, structural correspondence learning (SCL) can be used to adapt linear discriminate models. This method tries to find a representation of the data that behaves similar in the source and in the target domain. Blitzer et al. (2006) evaluated the model on POS tagging and Blitzer et al. (2007) on sentiment classification.

3.2 Domain Adaptation in Statistical Machine Translation

In recent years more and more interest has been gained by the domain adaptation problem in machine translation. After first improvements by Bulyko et al. (2007) inspired by research in the field of speech recognition, domain adaptation became one of the main focus points in machine translation research (Banerjee, 2013; Carpuat et al., 2012).

As described in the last section, a log-linear model is used in machine translation to combine the different knowledge sources. In contrast to the other models used in an SMT system, this one is trained discriminatively and therefore techniques described in the last section can be applied. But in contrast to the translation model and language model, quite few data is needed to train this model. Therefore, most work in domain

adaptation is focused on adapting the different components of the machine translation system towards the target domain.

Clark et al. (2012) presented an approach to adapt the log-linear model itself. In their method, they use general as well as domain-specific weights for all the models combined in the log-linear model. Thereby, dependent on the current domain, the influence of the models can be changed to get the overall best BLEU score.

In most approaches the two main components of a machine translation system, the language model and the translation model are adapted towards the target domain. Since both models are no discriminative models, the techniques described in the last section cannot be applied directly.

For both models different types of data to perform the adaptation is needed. For the language model, the adaptation techniques use monolingual target language corpora. The main advantage here is that this data is widely available. For the translation model, the first idea is to use parallel data. Since this is often not available techniques to use other types of data have been presented.

A first approach to perform adaptation for statistical machine translation systems was presented in Bulyko et al. (2007). Inspired by the improvements of language model adaptation techniques in speech recognition, they proposed to use a linear and log-linear combination of different language models trained on in-domain and out-of-domain data. The weights were optimized using Powell search directly towards minimizing the TER score.

Similar experiments were done by Koehn and Schroeder (2007), to adapt translation systems for European languages. They also used log-linear combination of different language models, with weights optimized directly towards a machine translation error metric. For the linear combination of language models, they used weights that try to minimize the perplexity of the development data.

A different approach to perform domain adaptation is to select the language model training data from huge background data (Moore and Lewis, 2010). Therefore, they trained first a language model on the in-domain data and a second language model on a randomly selected subset of the background data. Then the cross-entropy of both language models is used to select the language model training data.

If translation model adaptation is performed, often no parallel in-domain training data is available. Therefore, in addition to techniques trying to adapt the translation

model using some parallel in-domain data, approaches using other resources have been suggested. These resources can be monolingual data as well as comparable data or dictionaries.

For the first case, where some in-domain data is available, the approaches can again be divided into mixture-models, instance weighting and data selection. In most cases, language model adaptation is done in addition.

Foster and Kuhn (2007) use linear and log-linear language model and translation model adaptation. They optimize the weights for the different domains on a development set or set the weights according to text distance measures. A different way is to use two translation models and combine them with the alternate decoding path model (Koehn and Schroeder, 2007).

Another method is presented by Bisazza et al. (2011). They only fall back to the general model if the in-domain model does not have any information about the translation unit. Furthermore, they use an indicator feature to prefer the in-domain model.

In the case where it is unknown which part of the data matches the applications, Tam et al. (2007) presented an approach to infer the topics of the test and training data using a bilingual LSA. In this model they enforce a one-to-one topic correspondence between source and target language. Thereby, they can infer the topic only on the source test data. They use this information to then perform marginal adaptation of the language model as well as for the bilingual lexicon. The adapted lexica are then combined log-linearly with the other models.

A different technique is to increase the weight of the in-domain data. Matsoukas et al. (2009) changed the weights of the phrase pairs by assigning discriminative weights to the sentences in the parallel corpus. Thereby it is possible to give more weights to phrase pairs extracted from sentences found in the in-domain corpus.

Another way to perform instance weighting was presented in Foster et al. (2010). In contrast to the work mentioned before, in this approach the instance weighting is done at the phase pair level and not at the sentence level. Furthermore, a feature-based method is used to determine the usefulness of the phrase pair. Then this model is linearly combined with the in-domain model.

A third approach to adapt the translation model is to perform data selection. A first method was proposed by Hildebrand et al. (2005) which adapts the translation

using similar sentences automatically found using information retrieval techniques.

More recently the approach presented by Moore and Lewis (2010) for the language model has also been used for translation model data selection. Axelrod et al. (2011) and Axelrod et al. (2012) used the language model cross-entropy of the source or target language and in addition, they also showed results using a combination of both. If both scores are combined, the score is no longer negatively affected when one language is morphologically rich. In Duh et al. (2013), a continuous space language model using a recurrent neural network is used for calculating the cross-entropy instead. Since it can better generalize using only small amounts of in-domain data, they could show additional improvements compared to using an n-gram language model for calculating the cross-entropy.

Instead of using only the language model cross entropy, Mansour et al. (2011) combined it with the IBM1 cross-entropy. They could improve the data selection by using a combination of both cross-entropies.

Banerjee et al. (2012a) presented an approach to select the training data according to improvements in translation quality. Small batches of the data are selected and only added to the training if they lead to an improvement of systems' translation quality. In Banerjee et al. (2012b), the sentences which are able to cover OOV words are added to the training data.

In many tasks, we do not have this parallel in-domain data to perform the adaptation. Therefore, authors suggested approaches to also adapt the translation system in this condition. One very successful strategy in this condition is to generate synthetic parallel text by translating the monolingual corpus with a baseline system. Then this corpus can be used to adapt the baseline system.

Ueffing et al. (2007) used monolingual source texts and translated them using a baseline system. Then translations having a high confidence are used as additional training data. In their algorithm this is done for several iterations.

In Schwenk and Senellart (2009) the approach is explicitly used for domain adaptation. They build a baseline system using parallel UN data and then adapt the system towards the news domain using only monolingual data.

In Bertoldi and Federico (2009), target monolingual data is also used and translated in the reverse direction. They showed that this data is more helpful than monolingual source data translated into the target language.

Another approach is presented by Snover et al. (2008). They used cross-lingual information retrieval to find similar target language corpora. This data was used to adapt the language model as well as to learn new possible translations.

In addition, authors tried to use other knowledge sources like dictionaries or comparable data. Wu et al. (2008) adapted the system using hand-made dictionaries and monolingual source and target language texts.

Daumé III and Jagarlamudi (2011) mine comparable data to adapt the translation system to a new domain. They used Canonical Correlation Analysis to find translations for unseen words. A similar approach was presented in Prochasson and Fung (2011). They also use comparable data to find translations for rare words. Carpuat et al. (2012) tried to find new translations as well by using comparable data as well as active learning.

In Carpuat and Wu (2007) an approach to phrase-sense disambiguation using a discriminative classifier is presented. By using this type of translation model, Carpuat et al. (2012) tried to use techniques from domain adaptation for machine learning to adapt the model to the target domain.

If documents from different domains need to be translated, Banerjee et al. (2010) present an approach to combine different adapted translation systems by first determining the domain of an input sentence using a SVM classifier. Then the domain specific system of the selected domain is used to translate the input sentence.

3.3 Comparison of this Work to the Related Work

In Chapter 6, we present and analyze two methods to combine in-domain and general data in a machine translation system. The idea of one of the methods has some similarity with the one presented in Daumé III (2007). Since machine translation in contrast to other ML tasks uses only very few but complex features, we are also need to change the features itself, while Daumé III (2007) only duplicates them. As mentioned before, this task has already drawn the attention of several other research groups (Bisazza et al., 2011; Foster and Kuhn, 2007; Koehn and Schroeder, 2007) on different adaptation tasks. In this chapter we will compare our method to these approaches and point out the differences. Furthermore, we will perform a detailed analysis on the influence of adapting the different aspects of the translation model.

In Chapter 7 we will concentrate on the problem of adapting the machine translation system towards a very specific topic. We will show that in this case the problem of learning new translations for topic-specific words is very important. The problem of learning new translations has been address by Wu et al. (2008) using hand-made dictionaries, by Daumé III and Jagarlamudi (2011) and Prochasson and Fung (2011) by using comparable data and by Banerjee et al. (2012b) by applying text normalization and using additional parallel data.

In the method presented here, we learn the additional terminology from Wikipedia. The advantages of this approach are that Wikipedia is freely available for many languages and very easy to acquire. Furthermore, it covers many different topics providing even translations for very specific topics. We combine the method with techniques to translate different morphological word forms.

To improve the modeling of the genre, we present a class-based language model approach and a continuous space language model. Several continuous space language models (Le et al., 2011; Mikolov et al., 2010; Schwenk, 2007) have been presented recently. For domain adaption, Duh et al. (2013) used a continuous space language model to perform data selection. However, the language model presented in this thesis, is to our knowledge the first one which is directly integrated into the decoder.

4

Application Scenarios

In this chapter we will describe the tasks and applications upon which the approaches presented in the next chapters will be tested. In the first section, we will give an overview of the different translation tasks that we investigate in this thesis.

In the next part, we will describe the data that is available for the tasks. The text corpora differ in size and domain.

Finally, we will perform initial experiments using baseline systems. We will discuss the translation quality and point out the different problems that need to be tackled by the methods described in the following chapters.

4.1 Tasks

One interesting application for machine translation is the translation of lectures and speeches. In this case, the lecture is first recognized by an ASR system. Afterwards, the machine translation system is used to translate the text of the source language recognized by the ASR system into a different language. In our experiments we concentrated on translating from German into English. We applied the MT system to two similar tasks. In the first subtask, we evaluated our approaches on the task of translating TED lectures. For this subtask, we have got the advantage that we have parallel in-domain data from the same source.

In the second subtask, we used German computer science lectures collected at the KIT as a test set. In this case, the in-domain training data from the TED website is

not perfectly matching. Furthermore, the lectures are targeted for students from the university and not for a general audience like it is the case with the TED lectures.

4.2 Data

After describing the task in the previous section, we will now describe the data that we used to train our models as well as the data that we used to test the different approaches described in this work.

4.2.1 Test Data

As mentioned in the previous section, the main task where we applied machine translation is the translation of lectures and speeches. To test the performance on this task, we used two different test sets.

The first test set is extracted from the TED corpus. It consists of talks with a total length of around 3500 segments. In contrast to most of the other corpora, the text is not segmented into sentences, but into segments that are used for subtitling. Usually, these subtitles are shorter than sentences. The TED talks are targeted towards a general audience. Consequently, the vocabulary in these talks is not as specific as it is normally in university lectures. This is also indicated by the smaller vocabulary used in the TED lectures. Furthermore, they are given by very good speakers leading to less spontaneous effects like disfluencies and repetitions which occur more often in less staged presentations like university lectures.

The second test case consists of computer science university lectures. In this use case, we recorded, transcribed and translated computer science lectures from the KIT. Compared to the TED lectures we have got a more difficult vocabulary and need to be able to translate quite specific words. We used around 2,000 sentences as test set. The numbers are summarized in Table 4.1. We calculated all the numbers on the preprocessed corpus as it is then used for translation.

4.2.2 Training Data

The largest freely available parallel corpus for German to English is the “European Parliament Proceedings Parallel Corpus ”(EPPS) (Koehn, 2005). This corpus contains all the proceedings from the European Parliament starting from April 1996. It is

Table 4.1: Overview of the test data

Corpus	Lines	Source		Target	
		#Words	Voc.	#Words	Voc.
TED	3,502	31,414	4,870	31,732	4,089
CS Lectures	2,143	44,619	3,788	47,547	2,937

available in 21 languages. It has been aligned at the document level using the document IDs and then split into sentences. Afterwards the Church and Gale algorithm has been used to align this corpus on a sentence level (Gale and Church, 1993).

The parallel German-English version contains around 40M words. The exact statistics are shown in the overview Table 4.2.

A second important corpus for this language pair is the News commentary corpus (NC). This corpus has been extracted from the web-site "Project Syndicate". On this webpage commentaries from different persons mainly about politics and economics are published and translated into other languages. The size of this corpus is only around 10% of the EPPS corpus. Furthermore, this corpus contains written text compared to the speeches from the EPPS corpus. However, both corpora cover a wide variety of topics.

In addition, we used the BTEC corpus in these experiments. We used an in-house version of this corpus, which is a parallel German-English corpus. The BTEC corpus is again a speech corpus. It contains transcriptions and translations of spoken dialogs of basic travel expressions. Therefore, a characteristic of this corpus is that is contains very short segments and that the language used in the corpus is rather simple.

The last important parallel corpus is the TED corpus. The TED corpus is collected from the TED website. The corpus contains the subtitles and their translations of the TED talks. These talks are short presentations given by invited speakers on very different topics. However, the style of the talks is quite similar. All the talks are very well prepared and the talks are usually given by good speakers. Furthermore, since they are geared towards a general audience, quite few technical terms are used. For the purpose of training an MT system, we joined adjacent segments into sentences, since the corpus is originally split into shorter segments.

Table 4.2: Overview of the parallel training data

Corpus	Lines	Source Words	Target Words
EPPS	1.5M	41.5M	41.9M
NC	99K	2.6M	2.4M
BTEC	109K	832K	840K
TED	51K	923K	942K
Lectures	7.4K	144K	150K

In addition, we used a corpus of university lectures for some oracle experiments. This training data is very similar to the test set of university lectures and it is quite unrealistic that such well matching data is available in a real-world scenario. Therefore, we only used this data to show how good we can get with quite small, but very similar data.

4.2.3 Development Data

After training the statistical machine translation models on the large data described in the previous part, we normally tune the weights of the different models on a development data set. Since the number of different models is quite small, ranging from 10 to 20 in most of our experiments, only small amounts of data compared to the training data size is needed. In the experiments we typically used around 2,000 sentence of development data.

Depending on the scenario we used different development sets. In a first scenario, we simulated a situation where no in-domain data is available. Consequently, we could not train our log-linear weights on that data. Instead, we used additional data from the EPPS corpus. Of course, this data does not fit the testing condition as well as in-domain data and therefore the resulting translation quality will be worse.

In another scenario, we used TED talks of around 2,000 segments to optimize our weights. In this case, the development and test data match better. Finally, in some experiments we also used computer science lectures (CS) as development data.

The different development sets are summarized in Table 4.3.

Table 4.3: Overview of the development data

Corpus	Lines	Source		Target	
		#Words	Voc.	#Words	Voc.
EPPS	2,000	57,668	8,069	58,880	6,265
TED	1,711	16,064	3,607	16,475	2,921
CS	1,420	31,467	2,877	34,442	2,141

4.3 Initial Experiments

Before we describe the different techniques we used to adapt an SMT system towards a task, we will analyze the different tasks in some initial experiments. Therefore, we will analyze the influence on in-domain data on the different tasks and have a look at the problems that arise when using the systems on a new task. We will focus on the number of out-of-vocabulary words as well as on the context that is used during translation.

In all the experiments, we measured the translation quality in case-sensitive BLEU scores.

4.3.1 TED Translation Task

In a first series of experiments we looked at the translation quality of the TED translation task. Here we calculated the BLEU scores on the TED test set mentioned before. The results are summarized in Table 4.4.

The baseline system does not use any in-domain data. This system is only trained

Table 4.4: Initial experiments on the TED task

Condition	Dev	Test	OOV	Real OOV	Avg. Phr. Len.	
					Source	Target
Baseline	27.54	21.34	422/314	307/245	1.75	1.70
+ TED Dev	24.09	21.59	422/314	307/245	1.84	1.86
+ TED Training	26.22	24.12	334/255	255/205	1.89	1.90
TED only	25.34	24.50	862/673	726/592	1.51	1.51

on the EPPS, news commentary and BTEC corpus. The aforementioned EPPS development set was used as the development set. The BLEU scores on the development data are not comparable to the other results, since it uses a different development set. Using this data, a BLEU score of 21.34 on the test set can be achieved.

In the test set, 422 words cannot be translated since they do not occur in the training corpus. These 422 occurrences result from 314 different words. Compared to the size of the test data of 31,414 words, this means that the OOV rate is 1.34%. As mentioned before, the TED talks are targeted towards a very broad audience so that only very few specific words are used. Therefore, the OOV rate is quite low.

Many OOV words are proper names, since these occur quite rarely. If we translate European languages, proper names are often the same in both languages. Therefore, a simple strategy to handle OOV words is to pass them to the target language. In order to see how often this strategy does not solve the problem, we calculated a second number denoted as *real OOV* in Table 4.4, which ignores all the OOV words which also occur in the reference translation. In this case, we only have 307 OOV words, leading to an even smaller OOV rate of 0.98%.

We also analyzed the average phrase length, since statistical machine translation systems normally perform better if longer phrase pairs can be used. In this case, we use phrase pairs with an average source length of 1.75 words and an average target length of 1.7 words.

In the second system, we optimized the same models on the development data of the TED corpus. Although we only use around 2,000 segments of in-domain data, we could already improve the translation quality by 0.25 BLEU points. Since no additional training data is used, the OOV rate does not change at all, but the different scaling factors now generate translations using longer phrase pairs. The average phrase pair length is increased by roughly 0.1 words.

Afterwards, we added the in-domain monolingual and parallel training data and retrained the models. Although the in-domain data is quite small compared to the out-of-domain data, this improves the translation quality. The performance on the development data could be improved by around 2 BLEU points and on the test data even by 2.5 BLEU points. In addition, the number of OOV words could be reduced further using this data. Using the modified definition of OOV words, the OOV rate could be reduced to 0.81%. Furthermore, the average phrase length is increased slightly.

Finally, we did one additional experiment using only the data from the TED corpus for training. In this case, the number of OOV words increases dramatically, since we are using a much smaller training corpus. Furthermore the phrase pair length is considerably smaller. On the development data, this system performs better than the system using no TED data, but worse than the system using all data. In contrast, on the test data this system achieved the best performance. Consequently, concatenating all the training data seems not to be the best approach.

4.3.1.1 Examples for the TED Translation Task

In addition to the automatic evaluation, we also show some example translations in Example 4.1. These example sentences from the TED test set illustrate some problem that arise when using the translation system for a new domain. The translations were generated with the "+ TED Training" system described in the previous part.

In the first example, a translation which is common in the largest training domain, the EPPS corpus, is used to generated the translation. But this translation do not fit well for the TED domain. In the example, the German word *Schiff* is translated into *vessel* instead of *ship*.

In the second example the verb *sich vorstellen* has to be translated into *to imagine*, but in this case it is translated into *present*. The problem is that the verb *vorstellen* is very often translated into *present* in the European Parliament.

Finally, there are two examples were we do not generate the correct translations since we are using a very small context when generating translations for new domains. In the first example the German word *vor* is translated into *before* instead of *ago* and in the second case we cannot correctly generate the agreement between *these* and *story*.

4.3.2 Computer Science Lectures Translation Task

We performed similar experiments for the second translation task, the translation of university lectures (CS task). Here we used the computer science lectures as a test set. The initial experiments are summarized in Table 4.5.

For the task we also build a baseline system that is trained on the EPPS, news commentary and BTEC corpora. Since we want to test the situation where no in-domain data is available, we again optimized the system on the EPPS development set. This leads to an initial BLEU score of 20.21 on this task. In contrast to the TED task,

Source:	der Baupläne des **Schiffs**.
Reference:	on the blueprints of the **ship**.
Translation:	the design of the **vessel**.
Source:	wir wollen **uns** das best mögliche Ergebnis **vorstellen**.
Reference:	we want to **imagine** the best case scenario outcome
Translation:	we want to **present** the best possible result.
Source:	es gibt Hinweise darauf, dass schon die Neandertaler **vor 60000 Jahren**
Reference:	there is evidence that Neanderthals, **60000 years ago**,
Translation:	there are indications that the Neanderthals **before 60000 years**
Source:	denn **diese** Liebes Geschichte,
Reference:	because **this** love story,
Translation:	because **these** love story,

Example 4.1: Examples of the TED task

Table 4.5: Initial experiments on the CS task

Condition	**Dev**	**Test**	**OOV**	**Real OOV**	**Avg. Phr. Len. Source**	**Target**
Baseline	27.54	20.21	1437/624	1150/521	1.64	1.62
+ TED Dev	24.09	21.50	1437/624	1150/521	1.71	1.78
+ TED Training	26.22	22.97	1219/563	959/467	1.76	1.80
TED only	25.34	21.09	2383/985	1993/85	1.42	1.43
CS Dev	25.37	22.72	1437/624	1150/521	1.50	1.64
+ TED Training	25.84	23.55	1219/563	959/467	1.58	1.74
Lectures Training	29.59	27.49	924/492	749/416	1.76	1.92

a lot more special terms are used in university lectures. Therefore, the OOV rate is considerably higher. The size of this test set is 44,619 words, therefore the OOV rate is 3.22% and the real OOV rate is 2.58%. The average phrase length of the initial system is 1.64 source words and 1.62 target words.

In contrast to the last task, in this task we have different types of in-domain data. First, there is the TED data. Form this source, we have quite a lot of data, but it does not fit the task perfectly. As already mentioned, the TED lectures do not use as specific vocabulary as the TED lectures and furthermore, they are more fluently spoken than university lectures. Secondly, we can use some parallel data also collected at the university on computer science lectures. This data is perfectly matching the task, but of course it is difficult to produce this data. Therefore, we would like to limit the need for this data.

If we look at the first case and use only the in-domain development data, we see that we again improve the translation quality by around 1.3 BLEU points. The improvements are substantially higher than for the other task. Furthermore, the average phrase length is increased in a similar way.

When also adding the in-domain monolingual and parallel training data, the performance can be increased even further. The gains are not as big as for the TED task, but the performance could be improved by around 1.5 BLEU points. Furthermore, the OOV rate could be improved slightly to a real OOV rate of 2.14%.

If we build a system only on the TED data, the performance drops significantly. We are still better than the system using no TED data at all, but the system using only the TED development data already performs better. So it is no longer a good solution to just use the best matching data.

In the second scenario, we have development data from university lectures. In this case, as shown in line five of the table, we can even gain 2.5 BLEU points, by just using the in-domain development data instead of the EPPS development data. One main difference seems to be, that longer translations are generated since the average source length is decreased, while the target length stays the same.

If we now add the training data from the TED domain, we can improve even more by 0.8 BLEU points. In this case, again longer phrase pairs are selected.

In a final experiment we used the parallel training data collected at the university instead of the TED training data. As mentioned before this is not possible for most

tasks, it is more of an oracle experiment of how good you can become in this task with in-domain data.

Although this corpus is very small, we could improve the translation quality by 4 BLEU points compared to the previously best system leading to a BLEU score of 27.49. Furthermore, the real OOV rate could be decreased considerably to 1.67%. Similar to the other experiments, the additional data also leads to the use of longer phrase pairs.

4.3.2.1 Examples for the CS Translation Task

For this translation task we show some example translations of the baseline system in Example 4.2. They were generated by the system trained on all data including the TED training data and optimized on the TED development set. These examples show problems when porting the translation system to the CS translation task.

In the first example, we have again the problem that translations were generated using example translations from the European Parliament. In this case, the word *zufällig* is translated into *adventitious* instead of *random*.

In the next two examples, there are several words which cannot be translated by the baseline system. The problem is that in the computer science lectures there are several very specific terms. For these terms no translations can be found in the out-of-domain parallel training data.

And finally, there is an example where there are several variables. These variables are treated as unknown words in the translation system. As can be seen in the examples, this leads to problems in finding the correct word order.

Source:	wir haben **zufällige** Gewichte.
Reference:	we have **random** weights.
Translation:	we have **adventitious** weights.
Source:	nun, **Perzeptronen**, eine tolle Sache, da hat man sehr viel drüber geschrieben und viel publiziert in diesen Jahren.
Reference:	now, the **perceptrons**, an awesome thing, much was written about this and much was published during these years.
Translation:	now, **Perzeptronen**, a great thing, as it has a great deal over written and published in recent years.
Source:	das sind in der Regel Basis Elemente wie Würfel Kegel, **Kegelschnitt**, **Rotations-Ellipsoid** und anderer **Torus** ...
Reference:	these are usually base elements such as cubes, cones, the **cone cut**, **rotation ellipsoid** and the other **torus**, ...
Translation:	these are generally base elements such as cube cone, **Kegelschnitt**, **rotation-Ellipsoid** and other **Torus** ...
Source:	das ist ein Vektor der hier **Y zwei plus Y drei** auf diesen Punkt zwei deutet, ja?
Reference:	this is a vector **Y two plus Y three** which points at this point two, right?
Translation:	this is a vector here **Y plus two Y** to this point three two suggests, yes?

Example 4.2: Examples of the CS task

5

Exploiting Mismatching Data

As many other machine learning tasks, machine translation suffers from the problem that its performance is worse if the training and test data do not match. We train our model to perform best for inputs drawn from the training distribution. When the test data is drawn from a different distribution than the training data, the prediction might no longer be as reliable.

If we want to improve the translation quality for this condition, the first aspect is to check whether we exploit the mismatching training data in an optimal fashion. As we mention in the introduction, one way to improve the translation quality on test data from another domain is to improve the generalization of the model. Therefore, we need to know why the model does not predict the translation on data from different domains as well.

One important point during translation is the available context. Since natural languages are ambiguous, often context information is necessary in order to find the correct translation. Therefore, we will first illustrate the problem on an example and then analyze, what happens if we need to translate an out-of-domain text. We will see that in this case the average phrase pair length is lower and thus, less contextual information is available to determine the translation.

Afterwards, we will describe the bilingual language model (Niehues et al., 2011) and its motivation. The bilingual language model makes parallel contextual information across phrase pairs available. Using this model, we can increase the context that is used to determine the translation of a word and therefore improve the generalization of the

translation system. This improves the translation quality especially when translating out-of-domain documents.

Finally, we will evaluate the approach on different translation tasks and show the influence on the translation quality.

5.1 Context Information in Phrase-based Machine Translation

Since all natural languages have ambiguities, it is not possible to translate the words independently of each other. Instead, there are always words which have different translations in the target language. To generate the correct translation for these words additional information are needed. Mostly, these ambiguities can be resolved by using the context the word is used in.

If we examine the German word *vor*, we will often translate it into *before*. For example, in the German phrase *vor der Abstimmung* we could translate like *before the vote*. But there are other situations, where we need to translate the German word *vor* into *ago*. For example, if we translate the phrase *vor einigen Jahren* into *several years ago*. In this example, we would need more context to select the correct translation. For example, the word *years* should indicate that the translation *ago* is better.

As shown in the examples, in many of these cases, the context within the sentence can already help to determine the correct translation. Of course, there are also examples, where even more context is needed. For example, the resolution of anaphors, like *he* is often only possible by having a context longer than one sentence.

In phrase-based SMT the default approach to model the bilingual context information is to use phrase pairs. In the phrase pairs we will implicitly encode context information. Therefore, this context is limited by the phrase boundaries. No bilingual information outside the phrase pair is used for selecting the target word. The effect can be shown in the following example sentence:

Original: *..., dass schon die Neandertaler vor 6000 Jahren ...*

Using our phrase-based SMT system, we get the following segmentation into phrases on the source side: *dass schon # die # Neandertaler # vor # 6000 Jahren*. Using this segmentation the translation of *vor* is not influenced by any other the source word.

However, apart from this segmentation, other phrases could have been conceivable for building a translation:
dass, *dass schon*, *dass schon die*, *schon*, *schon die*, *die*, *Neandertaler*, *vor*, *6000*, *6000 Jahren* and *Jahren*. Since we use a reordering lattice, phrase pairs that match a reordered source word sequence could be also used. Consequently, the phrase for *Jahren vor* could be used as well.

Using the phrase pairs mentioned above, different segmentations can be used to translate the phrase *vor 6000 Jahren*. It can use the segmentation into *vor* and *6000 Jahren* as done by the baseline system. Instead it could also translate the reordered source sequence *6000 Jahren vor*. As shown in Figure 5.1, in this case the model can use the same segmentation or the translation can be generated by segmenting it into *6000* and *Jahren vor*. In the phrase-based system, the decoder cannot make use of the fact that some of the segmentation variants lead to the same translation, but has to select one and use only this information for scoring the hypothesis.

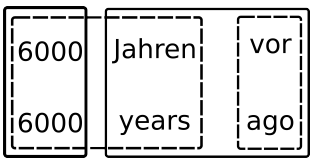

Figure 5.1: Alternative segmentations into phrase pairs

Consequently, if the first segmentation of the reordered sequence is chosen, the fact that *Jahren* is translated to *years* affects the translation of *vor* only by means of the target language model. No bilingual context, however, can be carried over the segmentation boundaries. This can lead to a wrong translation of *vor* into *before* as it was done by the baseline system.

If the training and test data would have matched better, the chances of having seen a translation for *for 6000 Jahren* would have been higher. Then the translation system could have used all the bilingual context information.

In order to analyze the context information that is available during translation, we calculated the average source and target phrase pair length that was used to generate the translation. We used the initial translation system described in Section 4.3. We tested the performance of this system that uses no in-domain training data on three different test sets. The results are summarized in Table 5.1

Table 5.1: Average phrase pair length of the initial system

Test set	BLEU	Avg. source PL	Avg. target PL
EPPS	27.17	2.19	2.22
TED	21.34	1.75	1.70
CS	20.21	1.64	1.62

Table 5.2: Different phrase pair lengths of the initial system

Dir.	Test set	1	2	3	4	5	6	7
Source	EPPS	34.30%	31.46%	20.04%	9.38%	4.83%		
	TED	49.76%	31.71%	13.21%	4.07%	1.25%		
	CS	54.69%	30.54%	10.87%	3.09%	0.81%		
Target	EPPS	39.17%	26.51%	17.57%	9.94%	4.40%	1.70%	0.72 %
	TED	56.38%	25.78%	11.62%	4.36%	1.42%	0.30%	0.12%
	CS	61.82%	23.13%	9.03%	4.04%	1.36%	0.49%	0.10%

The first test set contains documents from the EPPS corpus. This test set is very similar to the training data and therefore very well covered by the phrase pairs. In this case, we can use quite long phrase pairs with an average length of 2.19 source words and 2.22 target words. If we try to translate data from the TED corpus, we see that the average phrase pair length drops substantially. On this test set the average phrase pair length is only 1.75 words on the source side and 1.70 words on the target side. If we go over to translate computer science university lectures, we see that the average phrase length lowers even more.

In Table 5.2 we show the percentage of phrase pairs of the different lengths. As shown in the table, the percentage of phrase pairs with length two stays roughly the same throughout all data sets. But the amount of longer phrase pairs drops considerably, when going from an in-domain test set to an out-of-domain one. For the one word phrase pairs the opposite is the case. Here we have a huge increase in the number of phrase pairs when going from an in-domain test set to an out-of-domain test set.

After showing that the phrase pair length drops when translating out-of-domain data instead of in-domain data with one translation system, we also investigated, what happens when we add in-domain data to an existing translation system. The results

Table 5.3: Average phrase pair length for the TED task

Test set	BLEU	Avg. Source PL	Avg. Target PL
Baseline	21.59	1.84	1.86
TED Data	24.12	1.89	1.90

Table 5.4: Average phrase pair length for the CS task

Test set	BLEU	Avg. Source PL	Avg. Target PL
Baseline	22.72	1.50	1.64
TED Data	23.55	1.58	1.74
Lecture Data	27.49	1.76	1.92

for these experiments are shown in Table 5.3 for the TED translation task and in 5.4 for the university lecture translation task. Adding the TED data to both translation tasks increases the phrase pair length. By having more matching training data, we are able to use longer phrase pairs and thereby have more context in the translation process. This results also in an improved BLEU score. If we add a very small amount of in-domain lecture data instead of the TED data, we see for the translation of the lecture data even a higher improvement in phrase pair length. This additional data matches the test data even better and therefore, longer phrase pairs are available.

In summary, we see that we can use longer phrase pairs, if the training data and test data match better. In this case, a longer context is available when deciding how to translate the source words. If we want to build a translation system for a new domain, we have the problem that often this data is not available. In this case only very local bilingual context can be used to make the decision on the translation options. To improve the modeling of the bilingual context we introduce the bilingual language model in the next section.

5.2 Bilingual Language Model

If the machine translation system is used for a new domain, we are not able to encode enough bilingual context information with the phrase-based machine translation approach. This motivates to use ideas from a different approach to statistical machine

translation in the phrase-based machine translation system as done by the bilingual language model (Niehues et al., 2011).

Casacuberta and Vidal (2004) proposed to use a stochastic finite state transducer based on bilingual n-grams. For example, this approach was successfully applied by Allauzen et al. (2010) on the French-English translation task. In this so-called n-gram approach the translation model is trained by using an n-gram language model of pairs of source and target words, called tuples. While the phrase-based approach captures only bilingual context within the phrase pairs, in the n-gram approach the n-gram model trained on the tuples is used to capture bilingual context between the tuples. As in the phrase-based approach, the translation model can also be combined with additional models such as language models using log-linear combination.

Inspired by the n-gram-based approach, the bilingual language model extends the translation model of the phrase-based SMT approach by providing additional bilingual word context. The bilingual language model is based on tokens of target words and their aligned source words. Then we use an n-gram based language model on these token. Thereby it is possible to model the bilingual context across phrase boundaries.

In addition to the bilingual word context, this approach enables us also to integrate a bilingual context based on part of speech (POS) into the translation model as it has been done for the n-gram approach in Crego and Yvon (2010). When using phrase pairs, it is complicated to utilize different kinds of bilingual contexts, since the context of the POS-based phrase pairs should be bigger than the word-based ones to make the most use of them. There is, however, no straightforward way to integrate phrase pairs of different lengths into the translation model in the phrase-based approach, while it is relatively easy to use n-gram models with different context lengths on the tuples. We show how we can use bilingual POS-based language models to capture longer bilingual context in phrase-based translation systems.

The advantages of the additional model can be illustrated on the example mentioned in the previous section. In Figure 5.2 we show the phrase pairs that are used to generate the translations. In addition, we use now bilingual context by the bi-words. Thereby, in addition to the target context, the source context stays available across segment boundaries for the calculation of the language model score of the sentence. The context which is used while calculating the probabilities of the bi-words is illustrated by the lines. For example, when calculating the bilingual language model score for the bi-word

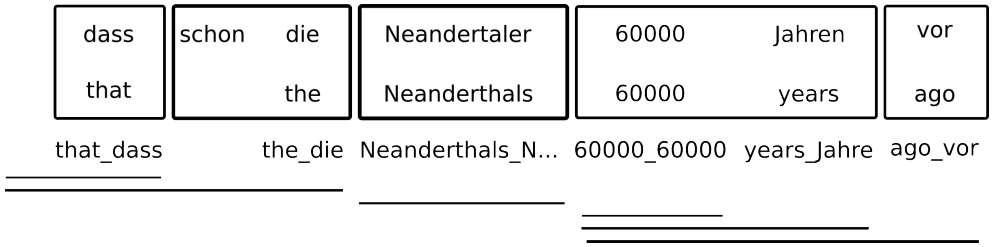

Figure 5.2: Example of segmentation and bilingual tokens

ago_vor $P($ *ago_vor* $\mid$ 6000_6000 years_Jahren) we can see that through the bilingual tokens not only the previous target word but also the previous source word is known and can influence the translation even though it is in a different segment.

5.2.1 Related Work

As previously mentioned, another approach to model the bilingual context during translation is the n-gram approach. It was presented in Mariño et al. (2006), derived from the work of Casacuberta and Vidal (2004), which used finite state transducers for statistical machine translation. In this approach, units of source and target words are used as basic translation units. Then the translation model is implemented as an n-gram model over the tuples. As it is also done in phrase-based translations, the different translations are scored by a log-linear combination of the translation model and additional models.

Crego and Yvon (2010) extended the approach to be able to handle different word factors. They used factored language models introduced by Bilmes and Kirchhoff (2003) to integrate different word factors into the translation process. In contrast, we use a log-linear combination of language models on different factors in our approach.

A first approach of integrating the idea presented in the n-gram approach into phrase-based machine translation was described in Matusov et al. (2006). In contrast to our work, they used the bilingual units as defined in the original approach and additional word factors were not used in their approach.

Hasan et al. (2008) used lexicalized triplets to introduce bilingual context into the translation process. Therefore, they calculate the probability $p(f|e, e')$ not only depending on the target word f and e, but also on an additional trigger word e'. This

additional source word can be outside the phrase pair and therefore the model is capable of using context information for translating ambiguous phrases.

Other approaches address this problem by integrating word sense disambiguation engines into a phrase-based SMT system. In Chan and Ng (2007) a classifier exploits information such as local collocations, part of speech or surrounding words to determine the lexical choice of target words, while Carpuat and Wu (2007) use rich context features based on position, syntax and local collocations to dynamically adapt the lexicons for each sentence and facilitate the choice of longer phrases. In Mauser et al. (2009) an maximum-entropy model using input features based on all source words of the sentence was used to model a longer source context.

Furthermore, Stroppa et al. (2007) and Gimpel and Smith (2008) integrated source side features to model the context beyond phrase boundaries. They use word based as well as class or syntactic motivated features.

5.2.2 Model

The bilingual language model is a standard n-gram-based language model trained on bilingual tokens rather than on simple words. These bilingual tokens are motivated by the tuples used in n-gram approaches to machine translation. We use different basic units for the n-gram model compared to the n-gram approach, in order to be able to integrate them into a phrase-based translation system.

In this context, a bilingual token consists of a target word and all source words that it is aligned to. Thus, given a sentence pair $e_1^I = e_1...e_I$ and $f_1^J = f_1...f_J$ and the corresponding word alignment $A = \{(i, j)\}$ the following tokens are created:

$$t_j = \{f_j\} \cup \{e_i | (i, j) \in A\} \tag{5.1}$$

Therefore, the number of bilingual tokens in a sentence equals the number of target words. If a source word is aligned to two target words, two bilingual tokens are created. For example, if we have the German word *zum*, which often translates into *to the*, we create the following bilingual tokens: *to_zum* and *the_zum*. If, in contrast, a target word is aligned to two source words, only one bilingual token is created consisting of the target word and both of the aligned source words.

The existence of unaligned words is handled in the following way. If a target word is not aligned to any source word, the corresponding bilingual token consists only of

the target word. In contrast, if a source word is not aligned to any word in the target language sentence, this word is ignored in the bilingual language model. This is the case the word *schon* in the example sentence in Figure 5.2.

Using this definition of bilingual tokens, the translation probability of source and target sentence and the word alignment is defined by:

$$p(e_1^I, f_1^J, A) = \prod_{j=1}^{J} P(t_j | t_{j-1} ... t_{j-n}) \tag{5.2}$$

The logarithm of this probability is used in the log-linear combination of a phrase-based translation system as an additional feature. It is worth mentioning that although it is modeled using a conventional language model, the bilingual language model is an extension to the translation model, in a way that the translation for the source words is modeled and not the fluency of the target text. As it is a sequence model, the order of the bi-words is also modeled. Therefore, this model helps to create the correct word order additionally.

To train the model, a corpus of bilingual tokens can be created in a straightforward way. In the generation of this corpus the order of the target words defines the order of the bilingual tokens. The common language modeling tools can be used to train the bilingual language model. As it was done for the normal language model, we used modified Kneser-Ney smoothing.

5.2.3 Comparison to Tuples

While the bilingual tokens are motivated by the tuples in the n-gram approach, there are major differences. They are mainly due to the fact that the tuples are also used to guide the search in the n-gram approach, while the search in the phrase-based approach is guided by the phrase pairs and the bilingual tokens are only used as an additional feature in scoring.

While no word inside a tuple can be aligned to a word outside the tuple, the bilingual tokens are created based on the target words. Consequently, source words of one bilingual token can also be aligned to target words inside another bilingual token. Therefore, we do not face the problems of embedded words, where there is no independent translation probability.

With this model, we do not create a monotonic segmentation of the bilingual sentence, but only use the segmentation according to the target word order. Therefore, it is not clear where to put source words, who have no correspondence on the target side. As mentioned before, they are ignored in the model.

An advantage of this approach, however, is that there is no problem handling unaligned target words. In this case, bilingual tokens with an empty source side are created. The position of the unaligned target words is guided by the segmentation into phrase pairs.

Furthermore, it is not required to apply additional pruning of the vocabulary, since this is already done by the pruning of the phrase pairs. In our phrase-based system, in most cases we allow only for ten translations of one source phrase.

5.2.4 Comparison to Phrase Pairs

Using the definition of the bilingual language model, we can again examine the introductory example sentence. We saw that when translating the phrase *vor 6000 Jahren* using a phrase-based system and a reordered source sequence *6000 Jahren vor*, the translation of *Jahren* into *years* can only be influenced by either the preceding *6000 # 6000* or by the succeeding *vor # ago*. However, it is not influenced by both of them at the same time, since either the phrase *6000 Jahren* or the phrase *Jahren vor* has to be chosen when segmenting the source sentence for translation. If we now examine the context that can be used when translating this segment applying the bilingual language model, we see that the translation of *Jahren* into *years* is on the one hand influenced by the translation of the token *6000 # 6000* within the bilingual language model probability $P(years\_Jahren|6000\_6000)$.

On the other hand, it is also influenced by the translation of the word *vor* into *ago* encoded into the probability $P(ago\_vor|6000\_6000, years\_Jahren)$. In contrast to the phrase-based translation model, this additional model is capable of using context information from both sides to score the translation hypothesis. In this way, when building the target sentence, the information of aligned source words can be considered even across phrase boundaries.

Furthermore, in the phrase pairs, we can only learn patterns that are continuous on the source side and on the target side. Consequently, we are not able to learn the pattern that *vor* $\star$ *Jahren*, where the $\star$ is a placeholder for a sequence of words, is

translated into *years ago*. In contrast, for the bilingual language model the patterns have to be continuous on the target side only. So the mentioned phonemena can be modeled by the sequence *years_Jahren ago_vor*.

5.2.5 POS-based Bilingual Language Models

When translating with the phrase-based approach, the decoder evaluates different hypotheses with different segmentations of the source sentence into phrases. The segmentation depends on available phrase pair combinations but for one hypothesis translation the segmentation into phrases is fixed. This leads to problems, when integrating parallel POS-based information. Since the amount of different POS tags in a language is very small compared to the number of words in a language, we could manage much longer phrase pairs based on POS tags compared to the possible length of phrase pairs on the word level.

In a phrase-based translation system the average phrase length is often around two words. For POS sequences, in contrast, sequences of four tokens can often be matched. Consequently, this information can be helpful only, if a different segmentation could be chosen for POS-based phrases and for word-based phrases. Unfortunately, there is no easy way to integrate this into the decoder.

If we compare how the bilingual language model is applied to the application of the phrase based translation system, it is much easier to integrate the POS-based information. In addition to the bilingual token, for every target word we can generate a bilingual token based on the POS information of the source and target words. Using this bilingual POS token, we can train an additional bilingual POS-based language model and apply it during translation. Word and POS sequences are scored separately by two different language models which cover different n-gram lengths. Therefore, even if the context of the POS-based bilingual language model is longer than the one based on the word information, it is no longer problematic.

The training of the bilingual POS language model is straightforward. The parallel corpus of POS tags is generated by running a POS tagger over both source and target side of the initial parallel corpus. Subsequently, the corpus of bilingual POS tokens is built using this parallel corpus of POS tags and the alignment information for the respective words in the text corpora.

The POS tag for every source and target word is required during decoding as well. Since we build the sentence incrementally, the tagger cannot be applied directly. Instead, we store the POS source and target sequences also during the phrase extraction. When creating the bilingual phrase pair with POS information, it is possible to have different possibilities of POS sequences for the source and target phrases. However, in this case only the most probable one for each phrase pair is kept. For an Arabic-to-English translation task, we compared the generated target tags with the tags created by the tagger on the automatic translations. It was found that the tags are different on only less than 5% of the words.

Using the alignment information as well as the source and target POS sequences, we can create the POS-based bilingual tokens for every phrase pair and store it in addition to the normal phrase pairs. At decoding time, the most frequent POS tags in the bilingual phrases are used as tags for the input sentence and the translation is generated based on the bilingual POS tokens built from these tags together with their alignment information.

5.3 Results

After presenting the bilingual language model in the last section, we will now focus on its impact on the translation quality. In a first step, we measured its impact on the translation quality on the systems presented before.

In the second part of the section, we will describe results using the bilingual language model in different conditions. Thereby, we are able to show that on greatly different tasks it is possible to improve the translation quality using this model.

5.3.1 Speech Translation Tasks

The results for these experiments are summarized in Table 5.5 for the TED task and in Table 5.6 for the computer science (CS) lectures task. If we look at the BLEU scores for the TED translation task, we see that the translation quality was improved in all conditions. In addition, the improvements by up to 0.7 BLEU points on the first two configuration of this task are substantially higher than the last configuration in Table 5.5. In contrast to the last configurations, in the first two configurations there was no parallel in-domain training data available. Therefore, there are less long phrase

Table 5.5: Overview of results for bilingual language model on the TED task

Condition	No BiLM		BiLM		Gain	
	Dev	Test	Dev	Test	Dev	Test
Baseline	27.54	21.34	27.72	21.87	+0.18	+0.53
+ TED Dev	24.09	21.59	24.35	22.36	+0.26	+0.77
+ TED Training	26.22	24.12	26.32	24.24	+0.10	+0.12

Table 5.6: Overview of results for bilingual language model on the CS task

Condition	No BiLM		BiLM		Gain	
	Dev	Test	Dev	Test	Dev	Test
Baseline	27.54	20.21	27.72	20.97	+0.18	+0.76
+ TED Dev	24.09	21.50	24.35	22.44	+0.26	+0.94
+ TED Training	26.22	22.97	26.32	23.60	+0.10	+0.63
CS Dev	25.37	22.72	25.53	23.21	+0.16	+0.49
+ TED Training	25.84	23.55	26.24	24.38	+0.40	+0.83
Lecture Training	29.59	27.49	30.10	28.17	+0.51	+0.68

pairs available and the parallel context available during translation is relatively small. Consequently, the additional context information introduced by the bilingual language model is more important.

The picture for the university lectures is similar. The performance was improved by up to 1 BLEU. Unlike the previous task on TED, significant improvements were yield even when using TED training data. This may due to the fact that the TED data does not match the university lecture domain perfectly.

In conclusion, we saw on both tasks the additional model helps to adapt a machine translation system towards a new domain especially when there is no in-domain training data available. In this case the bilingual language model can help to use more context information that normally is not available when using the translation system for a new domain. However, also in conditions where there is parallel training data available, it is possible to improve the translation quality by using the bilingual language model.

Context Length One motivation for using the bilingual language model is its capability to capture the bilingual contexts in a different way. To see whether additional bilingual context is used during decoding, we analyzed the context used by the phrase pairs and by the n-gram bilingual language model.

However, a comparison of the different context lengths is not straightforward. The context of an n-gram language model is normally described by the average length of applied n-grams. For phrase pairs, normally the average target phrase pair length (avg. Target PL) is used as an indicator for the size of the context. Thus these two numbers cannot be compared directly.

To be able to compare the context used by the phrase pairs to the context used in the n-gram language model, we calculated the average left context that is used for every target word where the word itself is included, i.e. the context of a single word is one. In case of the bilingual language model the score of the average left context is exactly the average length of applied n-grams in a given translation. For phrase pairs the average left context can be calculated in the following way: A phrase pair of length one gets a left context score of one. In a phrase pair of length two, the first word has a left context score of one, since it is not influenced by any target word to the left. The second word in that phrase pair gets a left context count of two, because it is influenced by the first word in the phrase. Correspondingly, the left context score of a phrase pair of length three is six (composed of the score one for the first word, score two for the second word and score three for the third word). To get the average left context for the whole translation, the context scores of all phrases are summed up and divided by the number of words in the translation.

For each of the two tasks, we selected the condition leading to the largest gains and showed the results of the context analysis in Table 5.7. As it is shown, the context used by the bilingual n-gram language model is longer than the one by the phrase pairs. The average n-gram length increases from 1.71 to 2.18 and from 1.69 to 2.13 respectively for the two given test sets.

If we compare the average n-gram length of the bilingual language model to the one of the target language models, the n-gram length of the first is evidently smaller, since the number of possible bilingual tokens is higher than the number of possible monolingual words. This was also observable with the perplexities of the two language models on the generated translations. While the perplexity of the target language

model is 133 and 137 on the two test sets, the perplexity of the bilingual language model is 378 and 437.

Table 5.7: Context used by the bilingual language model

Metric	TED		CS	
	No BiLM	BiLM	No BiLM	BiLM
BLEU	21.59	22.36	21.50	22.44
Avg. Target PL	1.66	1.78	1.78	1.71
Avg. PP Left Context	1.71	1.67	1.69	1.66
Avg. Target LM N-Gram	2.68	2.62	2.65	2.58
Avg. BiLM N-Gram		2.18		2.13

Overlapping Context An additional advantage of the n-gram-based approach is the possibility to have overlapping contexts. If we always use phrase pairs of length two, only half of the adjacent words would influence each other in the translation. The others are influenced only by the other target words through the language model. If we, in contrast, would have a bilingual language model which uses an n-gram length of two, this means that every choice of word influences the previous and the following word.

To analyze this influence, we counted how many boundaries of phrase pairs are covered by a bilingual n-gram. For the test set containing the TED lectures, 8,563 of the 14,272 boundaries between phrase pairs are covered by a bilingual n-gram. When translating the university lectures 12,678 of 24,953 boundaries are covered. Consequently, in both cases in 50 to 60 percent of the boundaries additional information can be used by the bilingual n-gram language model.

Bilingual n-Gram Length For the systems on the computer science task that gained the most from using the bilingual language model, we performed an additional experiment comparing different n-gram lengths of the bilingual language model. To ensure comparability between the experiments and to avoid additional noise due to different optimization results, we did not perform separate optimization runs for each

Table 5.8: Different n-gram lengths of the bilingual language model tested on the CS task

BiLM Length	Avg. n-Gram Length	BLEU
No		21.50
1	1.00	21.53
2	1.69	22.37
3	2.01	22.44
4	2.16	22.44
5	2.16	22.42
6	2.17	22.41

of the system variants with different n-gram length, but used the same weights for all of them. The system using no bilingual language model was trained independently.

In Table 5.8 we can see that the BLEU score increased until the bilingual language model reaches an order of three. Up to a n-gram order of four, the available context keeps increasing. For higher order bilingual language models, nearly no additional n-grams can be found in the language models. Also the translation quality does not increase further when using longer n-grams.

Example In addition to the automatic evaluation using the BLEU metric, the output of the different systems with and without the bilingual language model was investigated. An example sentence where we can see the improvements on the translation quality by using the bilingual language model is shown in Figure 5.1. As already mentioned previously we are able to use more bilingual context due to the bilingual language model and therefore translate the preposition *vor* in the correct way.

Source:	es gibt Hinweise darauf, dass schon die Neandertaler **vor 60000 Jahren**
Reference:	there is evidence that Neanderthals, **60000 years ago**,
No BiLM:	There are indications that the Neanderthals **before 60000 years**
BiLM:	There are indications that the Neanderthals **60000 years ago**

Example 5.1: Example translation using the bilingual language model

Table 5.9: German-English news translation task using bilingual language model

Metric	Test 1		Test 2	
	No BiLM	BiLM	No BiLM	BiLM
BLEU	30.37	30.52	44.16	45.09
TER	50.27	50.06	41.02	40.52

5.3.2 Additional Experiments

In order to show that the bilingual language model improves the translation quality on other conditions than the previously mentioned speech translation task as well, the performance gains on several other tasks were measured. First, we use a similar German to English system to translate two different sets of German news text into English. Furthermore, we used the model to improve the Arabic to English system on news data as well as web data. The Arabic to English system is trained on the GALE Arabic data.

5.3.2.1 German-English News Translation

In Table 5.9 the results of translation performance on the German-English translation task are summarized.

As depicted, the improvements of translation quality vary considerably between the two different test sets. While using the bilingual language model improves the translation by only 0.15 BLEU and 0.21 TER points on Test 1, the improvement on Test 2 is nearly 1 BLEU point and 0.5 TER points. One reason for the scores on the second test set showing a great deal of improvements compared to the first one is that the first test set uses one reference while the second one uses two.

5.3.2.2 Arabic-English

The Arabic-English system was optimized on the MT06 data. As test set the in-house test set Dev07-nw (News) and wb (Web Data) from the GALE project team Rosetta was used.

The results for the Arabic-English translation task are summarized in Table 5.10. The performance was tested on two different domains, translation of news and web

Table 5.10: Arabic-English translation task using bilingual language model

System	News			Web		
	Dev	Test		Dev	Test	
	BLEU	BLEU	TER	BLEU	BLEU	TER
NoBiLM	48.42	52.05	40.77	48.42	41.90	47.14
+ BiLM	49.29	53.51	40.04	49.29	43.12	46.66
+ POS BiLM	49.56	53.71	39.85	49.56	43.28	46.40

documents. On both tasks, the translation could be improved by more than 1 BLEU point. Measuring the performance in TER also shows an improvement by 0.7 and 0.5 BLEU points.

By adding a POS-based bilingual language model, the performance could be improved further. An additional gain of 0.2 BLEU points and decrease of 0.3 points in TER could be reached. Consequently, an overall improvement of up to 1.7 BLEU points was achieved by integrating two bilingual language models, one based on surface word forms and one based on POS.

As for the speech systems, we also compared the context used by the different models for this translation direction. The results are summarized in Table 5.11. As it was for the other language pairs, the context used in the bilingual language model is larger than the one used by the phrase-based translation model.

Another observable point is that when using the POS-based bilingual language model, shorter phrase are used. This supports the analysis that both bilingual language models model the context considerably accurately, thereby more frequently appearing short phrases can be used to generate the translation.

Table 5.11: Bilingual context used in the Arabic-English translation

Metric	News			Web		
	No	Word BiLM	POS BiLM	No	Word BiLM	POS BiLM
BLEU	52.05	53.51	53.71	41.90	43.12	43.28
Avg. Target PL	2.12	2.03	1.79	1.82	1.80	1.57
Avg. PP Left Context	1.92	1.85	1.69	1.72	1.69	1.53
Avg. BiLM N-Gram		2.66	2.65		2.33	2.31
Avg. POS BiLM			4.91			4.49

6

Adaptation by Combining In-Domain and Out-of-Domain Data

In the previous chapter, we described an approach to handle out-of-domain data better. In this chapter, we will evaluate different methods of exploiting in-domain data for domain adaptation, in addition to the out-of-domain data.

In general, the performance of the SMT system is improved when the training data is selected from sources similar to the test data. As seen in the previous chapter, for example, we are able to use longer phrase pairs and therefore have more context to decide what the correct translation is. Furthermore, if we have training data from similar sources, only translation options that match the application are present in the data, which will be therefore used for generating the translations.

It is not possible for many real-world scenarios to gather enough in-domain data. One approach to overcome this issue is to use all available data to train a general system and adapt the system using in-domain training data. For this, we will investigate different methods to integrate the in-domain data into the different models of the translation system.

In the introduction, we discussed different dimensions of text similarity. However, the strategies presented in this chapter will not account for this. For this task, we will use in-domain data that, in general, is machine better the application and therefore these examples should be favored the general ones. Confidence of using the in-domain

data will be automatically learned in the different approaches.

First, we will investigate different approaches to adapt the language model. The main advantage is that here we only need in-domain monolingual target data but no parallel data. We will try the existing linear (Stolcke, 2002) and log-linear methods to adapt the language model.

In Section 6.2.2, we will present a first approach to integrate parallel in-domain data into a machine translation system (Niehues et al., 2010). In this approach, the in-domain and general probabilities are combined in a log-linear manner and phrase pairs that are only seen in the in-domain phrase table are assigned default probabilities. In the next section, we will present another strategy to integrate the in-domain data using factored translation models (Niehues and Waibel, 2010). This approach enables an easy integration of different in-domain corpora.

After briefly reviewing two additional state-of-the-art approaches for integrating in-domain data, we will analyze and compare the key aspects of these approaches in Section 6.3. Although these techniques origin from different ideas of how to model the text data sources, they have many aspects in common and can be differentiated by four key aspects (Niehues and Waibel, 2012a). After analyzing the influence of the different aspects, we will present a final approach making use of the advantages of the different approaches.

6.1 Language Model Adaptation

The first approach to efficiently integrate in-domain data into a translation system is to adapt the language model. The main advantage is that no parallel in-domain corpus is required, but monolingual target data can be used. There are mainly two approaches to integrate the target in-domain data into the system. When adapting the language model, we can linearly combine the general and in-domain language model or this can be done in a log-linear way. These two approaches will be described below.

6.1.1 Linear Language Model Adaptation

The first way to combine different languages model is to combine them linearly. When we have an out-of-domain language model trained on the large out-of-domain corpus LM_{OUT} and an in-domain language model LM_{IN} trained on the in-domain part of the

data, we can compute two probabilities for a word w given its history h. We get an out-of-domain probability $P_{OUT}(w|h)$ computed by the first language model and an in-domain probability $P_{IN}(w|h)$ computed by the in-domain language model.

In order to get a probability, we can combine both probabilities linearly as:

$$P(w|h) = \lambda_{IN} P_{IN}(w|h) + \lambda_{OUT} P_{OUT}(w|h) \tag{6.1}$$

where $\lambda_{OUT} = 1 - \lambda_{IN}$. The advantage of this approach is that one language model probability for the combined language model is available and therefore perplexities can be calculated. By choosing the weights λ properly, we are able to take advantage of both language models. The out-of-domain language model LM_{OUT} is estimating the probabilities more accurate since it was trained on a considerably larger corpus. In contrast, the in-domain language model estimates probabilities more suitable for the task, since its training data is more similar to the corresponding task.

The critical point of this adaptation strategy is to choose the mixing weights λ properly. The first idea is to optimize the weights with the other weights of the log-linear model towards the best translation quality. However, it is not straightforward to optimize these linear weights together with the log-linear weights of the translation model. Therefore, we used a different method successfully applied in many ASR systems.

We minimized the perplexity of the development data as it is implemented by the SRILM Toolkit (Stolcke, 2002). In this case an iterative algorithm minimizes the perplexity of the combined language model on the development data, which is also used for the optimization of the other weights. Since the perplexity of the language model is independent of the other models used in the translation system, we can do this optimization in a first step. Then the combined language model is used in the decoder as the only language model and the weight of the language model in the log-linear model of the translation model is determined as described before.

The problem of this approach is that we perform two optimizations, the linear language model combination and the log-linear MERT, on the same development set. Since the language model weights are selected using this data, the language model can perform better on the development data than on other data. Therefore, the weight for the language model might be overestimated in the MER training. One could argue to use two development sets in order to avoid this problem. But there are several reason we only used one development set. First, as we aim to compare this approach to the

log-linear combination, the conditions should be the comparable. Furthermore, the number of weights is in both approaches the same. Consequently, overfitting problems should be in both techniques the same. And thirdly, in most real-word scenarios, we do not have additional development data to optimize the weights of the language model adaptation.

6.1.2 Log-linear Language Model Adaptation

The second strategy to combine language models is to use a log-linear combination. The previously mentioned probabilities $P_{OUT}(w|h)$ and $P_{IN}(w|h)$, can be combined in the following way:

$$f(w|h) = \lambda_{IN} log(P_{IN}(w|h)) + \lambda_{OUT} log(P_{OUT}(w|h)) \tag{6.2}$$

The correct selection of the weights plays an important role. In this case the resulting feature $f(w|h)$ is no longer a probability. Consequently, we cannot calculate, for example, the perplexity of the combined language model. In contrast to the aforementioned method, however, this approach fits smoothly in the log-linear model of the translation system.

Since we combine them log-linearly, we can add both language models independently into the translation system. The main advantage of this approach is that we can optimize the weights of the language model directly during the MER training. Therefore, we can select the weights in a way that maximizes the translation quality instead of solely examining the perplexity of the language model. While it is in general true that a lower perplexity of the language model increases the translation quality, this is not always the case. Therefore, it is advantageous to directly optimize towards an automatic evaluation metric instead of the perplexity.

6.1.3 Results

We evaluated both methods to integrate in-domain data on the task of translating TED lectures. In Table 6.1 we summarized the results of experiments in two conditions. In the first condition, there is in-domain parallel development data available and therefore it is required to optimize the system on the EPPS development data. In the second condition, in-domain development data from the TED corpus is available.

Table 6.1: Language model adaptation on the TED task

Approach	EPPS Dev		TED Dev	
	Dev	Test	Dev	Test
Baseline	27.61	24.26	26.32	24.24
Log-linear	27.55	23.98	27.46	25.26
Linear	27.58	23.42	27.50	25.39

Table 6.2: Language model adaptation on the CS task

Approach	EPPS Dev		TED Dev		CS Dev	
	Dev	Test	Dev	Test	Dev	Test
All Data	27.61	22.69	26.32	23.60	26.24	24.38
Log-linear	27.55	22.45	27.46	23.81	26.84	24.65
Linear	27.58	21.71	27.50	23.84	26.85	25.04

In the first condition, the translation performance can not be improved and the translation quality even decreases. This is especially the case for the linear adaptation approach. In contrast, when the development data is available, the performance can be increased significantly in both conditions. In this case, we can improve the translation quality by at least one BLEU point. Here, the linear adaptation approach performs slightly better.

The results for the computer science lectures task are shown in Table 6.2. In this case we evaluated the approach on three conditions. The case, where no in-domain data is available is examined. One condition with TED development data is evaluated. TED data nearly fits the domain, but there are differences in terms of Topic and slight difference in terms of speaking style. In addition, we also evaluated the approaches with the perfectly fitting development data from the computer science domain.

Similar as before, the translation performance was not improved if we do not have in-domain development data. In the second condition, both approaches lead to nearly the same performance and could improve the translation quality by roughly 0.2 BLEU points. In the last condition, we could improve by 0.3 to 0.6 BLEU points. In this case, again, the linear approach performed better than the log-linear one.

In conclusion, this approach can only improve the translation quality if in-domain development data is available. In this case, the translation quality could be improved by up to one BLEU point. If the development and test conditions are highly similar, the linear approach performs slightly better than the log-linear approach.

6.2 Translation Model Adaptation

In the previous section, adaptation techniques for the language model were shown. In this section, we will now concentrate on the adaptation of the translation model using parallel in-domain data. Two approaches to model the in-domain and out-of-domain characteristics of the training data are proposed in this work.

After presenting the two approaches, we compare them to other state-of-the-art techniques. Thereby, the key differences of these approaches are investigated and these ideas are combined to derive a better adaptation technique.

6.2.1 Problems Associated with Disregarding the Data Source

In the baseline approach, the phrase-based translation system is trained on the concatenated corpus of all available training data. By doing so we lose all information about the origin of the data and consequently every training sentence is equally important for generating the translations. In many cases this simplification is acceptable, but it is no longer the case if the training corpus consists of an in-domain and out-of-domain set. In this case, the information learned from the in-domain set should be considered more important than the one from the out-of-domain set.

The simplification is especially problematic, since often the size of the in-domain training data is very small compared to that of the out-of-domain data. To be able to make better use of in-domain examples, the translations from the in-domain data should be taken into account more significantly in the training process than the ones from out-of-domain data.

For example, for the German-English translation task, the largest available parallel corpus is the proceedings of the European Parliament. In the context of this corpus, some words have different English translations than they would have in university or TED lectures. If all sentences are treated equally, the probability of the translations specific to the proceedings would be more probable, since they were seen more often.

For example, the German word *Schiff* is translated into *vessel* in the proceedings while it should be translated into *ship* in university or TED lectures.

In order to overcome this problem, we need to combine the in-domain information for the corpus and the out-of-domain information in a more sophisticated way. In this process, it is important not to lose the better estimated information from the out-of-domain corpus, while making best use of the better fitting in-domain knowledge.

6.2.2 Domain Adaptation using Back-off Probabilities

In the phrase table adaption approach using back-off probabilities (Niehues et al., 2010) we try to combine the estimated phrase table probabilities of the in-domain phrase table with the general ones trained on all available data in a log-linear way.

In a first step, a phrase table on all available training data needs to be trained. This general phrase table contains all possible phrase pairs $(\bar{f}_i, \bar{e}_i)$ with their two lexical probabilities $\Phi^G_{lex}((\bar{f}_i, \bar{e}_i))$ and two conditional probabilities $\Phi^G_{cond}((\bar{f}_i, \bar{e}_i))$ estimated by the relative frequencies of the phrase pairs. Furthermore, the in-domain lexical probabilities $\Phi^{IN}_{lex}((\bar{f}_i, \bar{e}_i))$ and in-domain conditional probabilities $\Phi^{IN}_{cond}((\bar{f}_i, \bar{e}_i))$ can be calculated by training a phrase table only on the in-domain part of the corpus. Certainly, we will not be able to calculate these probabilities for all the phrase pairs, since not all phrase pairs occur in the in-domain corpus. Furthermore, they are often less accurately estimated, since the corpus is considerably smaller. On the other hand, they model the in-domain data more precisely.

Once we have the eight different scores for a phrase pair, we need to find a way to combine them. As mentioned before, the advantage of combining them log-linearly is the possibility to estimate the weights for the different scores directly during the optimization of the translation system. By combining the scores, the number of scores for the translation model is increased from four to eight. One problem, however, is that the MER training used in the optimization becomes unstable if a large number of weights is used. Since it is beyond the scope of this thesis to develop an optimization that is more stable with a large number of features, we analyzed if all features are needed.

The standard phrase pair has two lexical and two conditional probabilities. The lexical probabilities are introduced to smooth the conditional probabilities. The conditional probabilities are estimated at the phrase level. In contrast, the lexical probabil-

ities are estimated at a word level and afterwards the lexical probabilities for all words in the phrase pair are combined. In general, because word tokens are seen more often than phrase pairs, the lexical probabilities can be estimated more accurately than the conditional probabilities. This is especially the case for long phrase pairs.

The in-domain conditional probabilities were already smoothed by the general conditional probabilities, since the phrases are observed more often in the general corpus than in the in-domain one. Hence, it is not necessary to use in-domain lexical probabilities to smooth the phrase pair translation probability.

In conclusion, we use the four probabilities from the general phrase table as well as the two conditional probabilities from the in-domain phrase table. Therefore, we have only six phrase table scores instead of eight . The two in-domain lexical probabilities are not considered in this model.

After combining the scores in a log-linear way, there is nevertheless the problem of unknown probabilities. Since several phrase pairs do not occur in all corpora, we cannot calculate all the probabilities mentioned before for some phrase pairs. However, we cannot simply set these probabilities to zero since we combine them log-linearly and the logarithm of zero is not defined. Therefore, this case needs to be handled in a different way.

For the general probabilities, $\Phi^G_{lex}((\bar{f}_i, \bar{e_i}))$ and $\Phi^G_{cond}((\bar{f}_i, \bar{e_i}))$, the problem does not occur. Since we do not calculate these probabilities only on the out-of-domain data but on all data, every phrase pair which occurs in the in-domain data also occurs in the general phrase table.

In contrast, the in-domain conditional probabilities $\Phi^{IN}_{cond}((\bar{f}_i, \bar{e_i}))$ are only calculated on the in-domain data. Since many phrase pairs occur solely in the out-of-domain data and not in the in-domain data, the probabilities are not defined for these phrase pairs.

Since these phrase pairs did not occur in the in-domain corpus, they are mostly not able to generate good translations for this type of data. Therefore, the probability is backed off to a default probability in this case. In the experiments the lowest occurring value is taken. If there is another phrase pair that occurs in the in-domain phrase table, the in-domain condition feature will consequently have a higher value for this feature. Therefore, it will be preferred if other models do not suggest different pairs.

Table 6.3: Phrase table adaptation using back-off approach on the TED task

Approach	EPPS Dev		TED Dev	
	Dev	Test	Dev	Test
Baseline	27.61	24.26	26.32	24.24
+ PT Adapt	27.59	24.29	28.03	25.50
Log-linear LM	27.55	23.98	27.46	25.26
+ PT Adapt	27.56	23.77	28.40	25.89
Linear LM	27.58	23.42	27.50	25.39
+ PT Adapt	27.66	24.27	28.27	25.90

In the other case, all phrase pairs will have the same value and therefore the decision will solely be based on the other models.

6.2.2.1 Results

We tested this approach to adapt the phrase table on the same configuration as we did for the language model adaptation in the last section. The results are summarized in Table 6.3 for the TED translation task and in Table 6.4 for the task of translating computer science university lectures.

For both tasks, we see that the translation quality cannot be improved if we do not use in-domain development data. The BLEU score even decreases. In this case, the weights for the in-domain and out-of-domain models cannot be estimated well, since we do not have any in-domain example in the training process.

If we examine at the result for TED, we find that the translation quality is increased by 1.2 BLEU points if we do not use language model adaptation beforehand. If the translation system has already been adapted by language model adaptation, the translation model quality can still be improved by up to 0.6 BLEU points.

If we have only the TED development data to optimize the translation system for the university lectures, we see in the last series of experiments that the translation quality is improved only slightly by the the language model adaptation. The translation model adaptation, in contrast, can improve the translation quality under this conditions considerably. The BLEU score was improved between 0.7 and 1.2 BLEU points for this

Table 6.4: Phrase table adaptation using back-off approach on the CS task

Approach	EPPS Dev		TED Dev		CS Dev	
	Dev	Test	Dev	Test	Dev	Test
All Data	27.61	22.69	26.32	23.60	26.24	24.38
+ PT Adapt	25.79	22.50	28.03	24.28	27.29	25.31
Log-linear	27.55	22.45	27.46	23.81	26.84	24.65
+ PT Adapt	27.56	21.89	28.40	25.00	27.50	25.40
Linear	27.58	21.71	27.50	23.84	26.85	25.04
+ PT Adapt	27.66	22.46	28.27	24.58	27.58	25.56

task. Furthermore, the improvements seem to be the same even if we have adapted the system by language model adaptation before.

If we look at the last condition, where there is development data from the university lectures available, we can again improve the translation quality by 0.5 to 0.9 BLEU points. While the linear language model adaptation can outperform the log-linear one considerably when using no phrase table adaptation, this is no longer the case if we combine the language model and translation model adaptation.

In conclusion, we can get additional improvements on all conditions if we perform the translation model adaptation as described before. While the linear language model adaptation performs better when using no translation model adaptation, this is no longer the case if we combine language model and translation model adaptation.

6.2.3 Domain Adaptation using Factored Translations Models

A different approach to adapt the translation model is to model the influence of the in-domain and out-of-domain data explicitly (Niehues and Waibel, 2010). In this approach we model this by introducing the corpus identifier (corpus ID) as an additional target factor in the factored translation model (Koehn and Hoang, 2007). This enables us to adapt the SMT system by introducing two new types of features into the log-linear model in a phrase-based SMT system. First, we use relative frequencies to model the generation of the corpus ID tags similar to the translation model features that are used to model the generation of the target words. We can use features comparable to the word count and language model features to judge the generated sequence of corpus

IDs. Using the general framework of factored translation models leads to a simple integration of this approach into state-of-the-art phrase-based systems.

Factored translation models as presented in Koehn and Hoang (2007) are able to tightly integrate additional knowledge into the system. In most cases, the approach is used to incorporate linguistic knowledge, such as morphological, syntactic and semantic information. In contrast, we will use the approach to integrate domain knowledge into the system by introducing a corpus ID.

This corpus ID is a unique identifier of the different corpora that are used for building the translation system. For example, we can use the corpus ID "IN" for the in-domain part of the corpus, e.g. the TED corpus, and the "OUT" tag for the remaining part of the training corpus. A different approach would be to use four different corpus IDs for the four corpora used in building the translation system for the TED task as described in 4.2.2. In most experiments we only used two corpus IDs, since some preliminary experiments showed that it is important to distinguish between the out-of-domain data and the in-domain data, but not that important to distinguish between the different out-of-domain copora. The resulting representation for using two corpus IDs of an example sentence is shown in Figure 6.1.

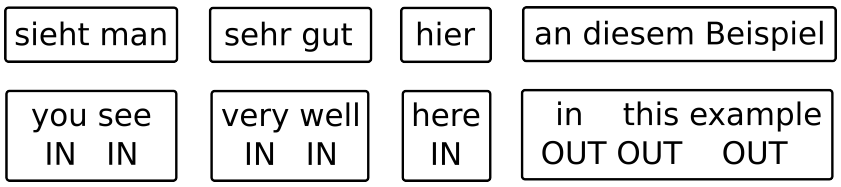

Figure 6.1: Example of German-English translation with corpus ID factors

In order to train this model we need to store for every phrase pair the corpus from which it was extracted. Using this information, the occurrences of the phrase pairs are no longer equally important, but we can, for example, prefer phrase pairs that were extracted from the in-domain corpus by using additional models. Phrases extracted from the out-of-domain corpus generate the *OUT* factor on the target side. Similarly, phrase pairs learned from the in-domain part will generate an *IN* factor on the target side.

Many phrase pairs occur in different parts of the corpus. For example, the phrase pair *hier # here* shown in the example in Figure 6.1 also occurs in the out-of-domain

part of the corpus. In these cases, both phrase pairs will be extracted and the decoder will select one of them depending on the models described in the following sections.

With this approach it is possible to see which parts of the translation are learned from the in-domain training examples and which parts are translated by using phrase pairs from the out-of-domain corpus. This information can then be used to judge the quality of the translation. Translations which are generated from in-domain phrase pairs will probably be better translations than the ones generated by phrase pairs extracted only from the out-of-domain corpus.

To be able to model this, we add two types of features to the log-linear model used in the translation system. The first one, which we call the *Domain Factors Translation Model*, models the probability that a sequence of corpus ID tags is generated. Similar to the translation model of words, we use features based on relative frequencies to model this probability. The different features are described in detail in the next section.

A second group of features is used to judge the corpus ID tag sequence similar to the target language model. We count the number of in-domain tags and use this as an *Domain Factors Sequence Model*. Different approaches to model this probability will be described in Section 6.2.3.2.

Since we use the general framework of factored translation models, the weights for these features can be optimized during the training on the development data of the log-linear model using, for example, minimum error rate training. The resulting weights prefer in-domain phrase pairs in a way that leads to the best translation performance on the development data.

6.2.3.1 Domain Factors Translation Model

The Domain Factors Translation Model describes the probability that a sequence of corpus ID tags is generated. If we look at the example shown in Figure 6.1, the features of the model should capture the probability of generating the sequence *IN IN IN IN IN OUT OUT OUT* if the input sequence is *sieht man sehr gut hier an diesem Beispiel*.

As mentioned before, this is similar to the phrase translation model in state-of-the-art SMT approaches. As described in Section 2.2, the phrase translation probability consists of a log-linear combination of four different probabilities. First, we use the conditional probability and the inverse conditional probability approximated by the

relative frequency. As shown in Equation 2.2, they can be calculated using the co-occurrence counts ($count(\bar{f}, \bar{e})$) of the source and target phrase. In addition, the lexical translation probabilities in both directions are used.

In our factored model we no longer have only the co-occurrence count depending on the source and target phrase $count(\bar{f}, \bar{e})$, but, in addition, a co-occurrence count depending on three parameters, $count(\bar{f}, \bar{e}, \bar{d})$, where $\bar{d}$ is the sequence of corpus ID tags. Consequently, we can extend the existing probabilities by three more possible features.

To begin with, we define the probability of the corpus ID tags given the source phrase $P(\bar{d}|\bar{f}, \bar{e})$, which can be approximated analogously to the existing translation probabilities as

$$P(\bar{d}|\bar{f}, \bar{e}) = \frac{count(\bar{f}, \bar{e}, \bar{d})}{\sum_{\bar{d}} count(\bar{f}, \bar{e}, \bar{d})} \tag{6.3}$$

Secondly, we define the probability of the target phrase given the source phrase and domain tag sequence $P(\bar{e}|\bar{f}, \bar{d})$. Since we cannot extract a phrase pair partly from one corpus and partly from another one, the corpus ID tags for all words of one phrase pair are the same. Consequently, this probability is the same as the probability of the target phrase given the source phrase restricted to the phrases extracted from the corpus indicated by the corpus ID tags. The probability can be calculated as

$$P(\bar{e}|\bar{f}, \bar{d}) = \frac{count(\bar{f}, \bar{e}, \bar{d})}{\sum_{\bar{e}} count(\bar{f}, \bar{e}, \bar{d})} \tag{6.4}$$

Finally, we define the probability with switched roles for the source and target phrase $P(\bar{f}|\bar{e}, \bar{d})$. This can be approximated as

$$P(\bar{f}|\bar{e}, \bar{d}) = \frac{count(\bar{f}, \bar{e}, \bar{d})}{\sum_{\bar{f}} count(\bar{f}, \bar{e}, \bar{d})} \tag{6.5}$$

6.2.3.2 Domain Factors Sequence Model

In the last part we described how to model the generation of the corpus ID tags. One main advantage of this approach is that we are able to introduce models to judge the different possible domain tag sequences for a given source sentence.

In the example given in Figure 6.1 another possible translation of the source sentence would generate the tag sequence *IN IN OUT OUT IN OUT OUT OUT*. By looking

only at the corpus ID sequence we should prefer the translation shown in the figure, since it uses more phrase pairs that occur in the in-domain corpus.

When defining the Domain Factors Translation Model, we adapt the standard translation model. However, when defining the Domain Factors Sequence Model, we cannot simply extend the language model approach used for the target words. There we would train a language model on the corpus ID tag sequence and then using this language model to evaluate the tag sequence. A training sentence always comes from a single document. Consequently, all words would have the same corpus ID tag. In the test case, however, we do not want to only generate sentences using phrase pairs extracted from the same corpus. A language model would not be able to meaningfully score sequences having different corpus IDs. Since there is no corpus on which to train such a language model, we have to use different types of models. To model this we propose two features.

Since we are not able to train a language model, we use a unigram model in the experiments, although the framework supports general sequence models. The first technique is to do this at the phrase level leading to a model similar to the phrase count model. Instead of counting all the phrases we can simply count the phrases with in-domain corpus ID tags. In the example shown in Figure 6.1, the first three phrase pairs are from the in-domain corpus. Therefore, the feature would have a value of three.

A second approach is to use an in-domain word count feature similar to the already existing word count feature. In this case, we do not count the phrase pairs that are extracted from the in-domain corpus, but the target words. In the example, the first five words were generated by phrase pairs learned from the in-domain data. The feature value of this feature is therefore five. We evaluated both types of features and present the results in the next section.

6.2.3.3 Results

As done for the Back-off approach, the domain adaptation using factored translations models approach was evaluated on the task of translating German TED and computer science lectures into English. In a first step, we concentrate on evaluating the two different domain factor sequence models. In the next step, we focus on the different scores in the domain factor translation model.

In the experiments reported in Section 6.2.2 it became apparent that we can only improve the translation quality if some in-domain development data is available. Therefore, we focused this time only on the condition where it is available.

Evaluation of the Domain Factor Sequence Models The results of our experiments using the different Domain Sequence Models are summarized in Table 6.5. In these experiments we did not use any language model adaptation. In the first two columns, we present the results for the TED translation task using TED development data. The next two columns show the results for the translation task of university lectures, where no additional development data is available. Consequently, in this condition, the TED data is used as development data as well. Finally, the last two columns show the results for the computer science university lectures if there is development data from the CS domain available.

We analyzed the influence of the Domain Factor Sequence Model using two different settings. This settings differ in the Domain Factor Translation Model. In the first configuration, we used the domain probability and in the second configuration, we used the source and target probabilities.

It is important to use one type of Domain Factor Sequence Model. If we do not use a Domain Factor Sequence Model, we often cannot improve over the baseline system and in several cases even perform worse. This is especially the case for the more difficult task of the translation of university lectures, where the test and training conditions do not match as well as in the TED condition. If we, in contrast to not using any Domain Factor Sequence Model, use the word count or phrase count feature, we can improve the translation model quality between 0.6 and 1.7 BLEU points for all the tasks.

If we compare both types of Domain Factor Sequence Models, the word count and phrase count based one, we can see no significant differences. But it seems to be important to use one of these models, because it is not possible to prefer in-domain translations without this model. The differences between the to model is that the word count model can distinguish between short and long in-domain phrase pairs since short ones have less words and therefore the will not be preferred as much as long ones. But this differences seems not to be very important.

In the following experiments we will therefore only use the phrase-based Domain Factor Sequence Model.

Table 6.5: Domain Factor Sequence Models

		TED		CS			
Translation Model	**Sequence Model**			**Dev on TED**		**Dev on CS**	
		Dev	**Test**	**Dev**	**Test**	**Dev**	**Test**
Baseline		26.32	24.24	26.32	23.60	26.24	24.38
Domain	None	26.42	24.67	26.42	23.54	25.29	24.15
	Word Count	27.71	25.60	27.71	24.39	27.06	25.11
	Phrase Count	28.17	25.90	28.17	24.25	27.65	25.19
Source/Target	None	26.92	24.86	26.92	23.54	25.94	24.85
	Word Count	28.02	25.62	28.02	24.21	27.55	25.59
	Phrase Count	28.11	25.48	28.11	24.63	27.66	25.13

Evaluation of the Domain Factor Translation Model After analyzing the Domain Factor Sequence Model, we will now concentrate on the Domain Factor Translation Model. We looked at three different configurations. In the first one, we only used the domain frequency. In the second one, we use the source and target relative frequencies as we did in the last approach for domain adaptation (cf. Section 6.2.2). Finally, we tested a configuration using all three probabilities.

The results for the TED translation task are summarized in Table 6.6. We tested all configurations without language model adaptation and with log-linear or linear language model adaptation. In all the conditions the translation quality could be improved by using any phrase table adaptation. The largest improvements of around 1.4 BLEU points were gained using no language model adaptation. However, even if we already use language model adaptation, the translation model quality can be improved by 0.7 BLEU points using the technique will all features. If we compare the different Domain Factor Translation Models, no significant difference can be seen for this task. So it seems to be important to use features that indicate how well the phrase pairs fits the domain, but all the features seems to perform similarly in this task.

In Table 6.7 we tested the same systems on the task of translating university lectures. In this set of experiments, we did not use any additional development data, but used the weights obtained from the optimization on the TED task. For this condition, we can see significant improvements over the baseline system using the translation model

Table 6.6: Domain Factor Translation Models for the TED task

Approach	No LM Adapt		Log-linear		Linear	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.32	24.24	27.46	25.26	27.50	25.39
Dom. Freq.	28.17	25.90	28.37	25.75	28.33	25.76
Source/Target	28.11	25.48	28.51	25.86	28.41	25.90
All	28.08	25.70	28.25	25.99	28.31	25.79

Table 6.7: Domain Factor Translation Models for the CS task (Dev on TED)

Approach	No LM Adapt		Log-linear		Linear	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.32	23.60	27.46	23.81	27.50	23.84
Dom. Freq.	28.17	24.25	28.37	25.12	28.33	24.46
Source/Target	28.11	24.28	28.51	24.76	28.41	24.67
All	28.08	24.65	28.25	24.45	28.31	24.90

adaptation approach as well. On all conditions, we can improve the translation model quality by 1 to 1.2 BLEU points, even if we already use language model adaptation.

The results for the university lectures using matching development data are shown in Table 6.8. Here, considerable improvements over the baseline system could again be reached. Depending on the condition, the translation quality could be improved by 0.3 to 0.8 BLEU points.

We can gather from the experiments that it seems to be important to use some kind of Domain Factor Translation Model, but less important which scores to use. They seem to model similar aspects.

Results for News Task We performed some additional tests on the task of translating German documents from the news-commentary domain into English in order to analyze whether this approach also works for different tasks. In this setup we used the news-commentary corpus as in-domain data. Results are summarized in Table 6.9. In the experiments we used the word count as a Domain Factor Sequence Model and all three probabilities in the Domain Factor Translation Model. The translation model

Table 6.8: Domain Factor Translation Models for the CS task (Dev on CS)

Approach	No LM Adapt		Log-linear		Linear	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.24	24.38	26.84	24.65	26.85	25.04
Dom. Freq.	27.65	25.19	27.86	25.27	27.76	25.16
Source/Target	27.66	25.13	27.47	25.05	27.72	25.31
All	27.37	25.16	27.86	25.34	27.73	25.29

Table 6.9: Domain adaptation using Factored Translation Models on the NC task

System	Dev	Test
Baseline	25.90	29.03
+ LM Adaptation	26.68	29.24
+ PT Adapt	27.07	29.69

adaptation could increase the translation quality further, even after performing language model adaptation. The BLEU score could be increased by 0.4 BLEU point on the test data. Consequently, this approach can be used to perform domain adaption on very different applications. We saw improvements on a speech translation task as well as on a text translation task.

6.3 Detailed Analysis and Combination of Different Translation Model Adaptation Strategies

In the previous section we presented two approaches to perform domain adaptation by integrating small amounts of additional in-domain parallel data. These techniques have in common that they try to encode the domain specific knowledge without losing the information learned from the much bigger parallel corpus. They differ in the way they model this knowledge. In the first approach, we train two different phrase tables and afterwards try to combine the features of the different phrase tables into a new one. We will refer to this one as *Back-off approach*. In the second one we use the factored translation model framework (Koehn and Schroeder, 2007) to include the source corpus of the phrase pair as an additional factor (*Factored approach*). Additional approaches

were suggested to adapt the translation model under the mentioned condition and will be reviewed in Section 6.3.1.

Although the modeling is different, several aspects of all approaches are similar. For example, most of them use the conditional probabilities restricted to the in-domain corpus as some type of feature. Furthermore, all these strategies try to combine different features in a log-linear way.

In order to better understand what is important when adapting the translation model, we analyzed the difference between the approaches (Niehues and Waibel, 2012a). Furthermore, we evaluated the different aspects of the techniques with respect to their influence on the translation quality.

By comparing the different approaches of translation model adaptation, we found two main aspects of the model that can be adapted. The first aspect is the candidate selection, where we determine for every possible source phrase, which translation options to consider during decoding.

The second aspect we can adapt is the selection of the phrase pair scores. Different possibilities of calculating phrase pair scores are used to encode the information about the origin of the training data.

6.3.1 Additional Approaches to Domain Adaptation

A first approach to integrate the source of the data into the translation model was presented in the *log-linear adaptation* approach (Koehn and Schroeder, 2007). In their work they used two phrase tables, one trained on the in-domain corpus and another one trained on the out-of-domain data. Then they used different weights in the log-linear model for the in-domain and out-of-domain scores. Therefore, they used 8 different scores in the log-linear model.

A different approach is the *Fill-up approach* presented by Bisazza et al. (2011). In this method the in-domain scores are used if the phrase pair occurs in the in-domain phrase table. Only for phrase pairs that do not occur in the in-domain corpus, they use the scores calculated on the out-of-domain phrase table. To distinguish between in-domain and out-of-domain phrase pairs, they introduce an additional indicator feature. With the help of this indicator feature, they can share the weights for the four scores of both phrase tables and only five scores are represent in the phrase table. They showed that this improves the optimization process using MERT.

6.3.2 Candidate Selection

In our scenario there are three different phrase tables. One trained only on the in-domain data providing the candidate translations $T_{IN}(\bar{f}_i)$ for a given source phrase $\bar{f}_i$, one trained on the out-of-domain data ($T_{OUT}(\bar{f}_i)$) and one trained on all data ($T_{ALL}(\bar{f}_i)$). Consequently, for a given translation task there are 3 different sets of phrase pairs, which were determined by candidate selection as described in Chapter 2.2. Each of these sets contain at most n translations for a given source phrase due to pruning.

In Figure 6.2 we show the situation for three source phrases A, B and C. In the general phrase table trained on all data, the source phrases A and B have the maximal number of translation options. For the out-of-domain phrase table, this is only the case for A, since some of the translations only occur in the in-domain text. Finally, in the in-domain phrase table no translation for C can be found.

The first approach here is to use the phrase pairs which are selected from the phrase table trained on all data $T(\bar{f}_i) = T_{ALL}(\bar{f}_i)$. In this case we would not adapt the component of the candidate selection at all. This approach was used by the back-off and factored approaches and will be referred to as *NoAdapt*.

A second approach is to use the union of the candidates selected from the in-domain and the out-of-domain phrase table $T(\bar{f}_i) = T_{IN}(\bar{f}_i) \cup T_{OUT}(\bar{f}_i)$. This was done in the log-linear and fill-up approaches. We will refer to this method as *UnionOut*. Instead of the out-of-domain phrase table, we could alternatively use the phrase table trained on all data and combine its candidate phrase pairs with the in-domain phrase pair selection $T(\bar{f}_i) = T_{IN}(\bar{f}_i) \cup T_{ALL}(\bar{f}_i)$. We will refer to this method as *UnionAll*.

As it is illustrated with source phrase A in Figure 6.2, when applying the union operation it can happen that more than n translation options end up in the final phrase table. But since many translation options occur in both phrase tables, the number is often less than $2n$.

A third approach is to mainly rely on the phrase pairs of the in-domain phrase table. Only if there are no or not enough candidate translations for one source phrase, we will fill up the candidate list with the ones suggested by the out-of-domain phrase table or the general phrase table, respectively. This leads to the definition of the translation candidates set as: $T(\bar{f}_i) = T_{IN}(\bar{f}_i) \cup T^k_{ALL}(\bar{f}_i)$, where $T^k_{ALL}(\bar{f}_i)$ are the top k translation of $T_{ALL}(\bar{f}_i)$ and $k = n - |T_{IN}(\bar{f}_i)|$, where n is the maximum number of

translation candidates and therefore k is always larger than or equal to 0. In our case n is 10. In contrast to the *Union* approaches, we will still have at most 10 translations for every source phrase. Analog to the *UnionAll* and *UnionOut* approaches, we will refer to these ones as *PaddingAll* and *PaddingOut*.

In a last approach, we allow only to fall back to the candidates of the out-of-domain phrase table, if there are no translations at all for the source phrase in the in-domain phrase table. In this case we will consider all out-of-domain candidates for this source phrase.

$$T(\bar{f}_i) = \begin{cases} T_{IN}(\bar{f}_i): & T_{IN}(\bar{f}_i) \neq \emptyset \\ T_{OUT}(\bar{f}_i): & else \end{cases}$$

We will refer to this approach as *SourcePadding*. In the example in Figure 6.2 out-of-domain candidates will only be used for source phrase *C*.

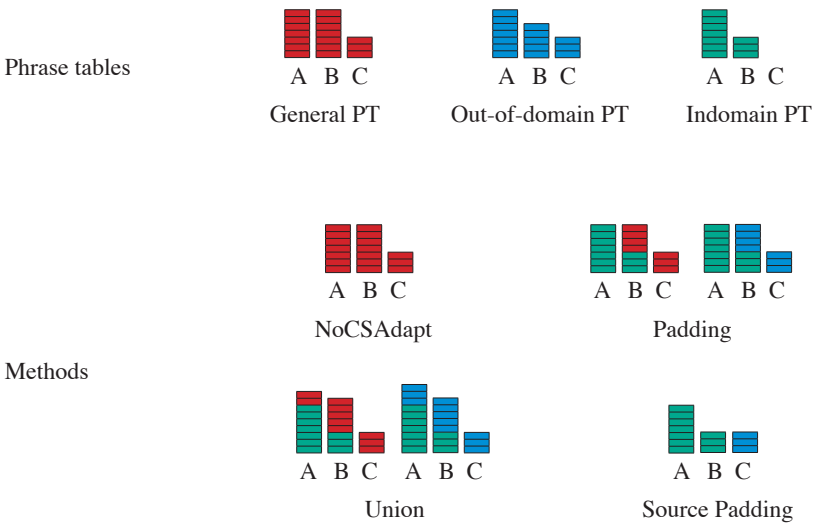

Figure 6.2: Adaptation approaches for candidate selection

6.3.3 Selecting Scores for the Phrase Table

The other step in the translation model that can be adapted is the scoring of the phrase pairs. Here the approaches differ in four key aspects. But first we need to

change the definition of the translation probability slightly for the adapted translation model. Originally the probability for a phrase pair $(\bar{e_i}|\bar{f_i})$ is defined as:

$$\begin{aligned} log(p(\bar{e_i}|\bar{f_i})) &= \sum_{s=1}^{S} \lambda_s log(\Phi_s(\bar{e_i}|\bar{f_i})) \\ &- log(Z_{\bar{f_i}}) \end{aligned}$$

In most phrase tables, we use $S = 4$ for different features to describe the probability of the phrase pair, which are combined log-linearly. The last term $log(Z_{\bar{f_i}})$ is used to normalize the probabilities so that they add up to 1. We extend it in the following way:

$$\begin{aligned} log(p(\bar{e_i}|\bar{f_i})) &= \sum_{s=1}^{S} \vec{\lambda}_s log(\overrightarrow{\Phi_s(\bar{e_i}|\bar{f_i})}) \\ &+ \sum_{s'=1}^{S'} \lambda_{s'} log(\Phi_{s'}(\bar{e_i}|\bar{f_i})) \\ &- log(Z_{\bar{f_i}}) \end{aligned}$$

Instead of one score derived from the S different phrase table features from one phrase table, we include several sets of phrase table features stemming from different phrase tables in the calculation of the overall phrase table probability. The different features are represented as vectors and the logarithm function is applied separately for each component. Furthermore, in some approaches there might be some additional features S' representing for example the corpus the phrase pair is extracted from.

The first key aspect is the usage of the phrase table features trained on all data. Although these features are not adapted to the target domain, they are often more reliable, since they are calculated on larger amounts of data. Therefore, they might be useful for smoothing the adapted features. We can use them for log-linear smoothing by defining $\Phi_s =< \Phi_s^{All}, \Phi_s^{Adapted} >$ or we can ignore them by using only the adapted features ($\Phi_s =< \Phi_s^{Adapted} >$) and therefore do not need to train an additional phrase table on the whole data. While the back-off and factored approaches extend the features of the general phrase table by the adapted ones, the general ones are not used at all in the log-linear and fill-up approaches.

Secondly, when adding the in-domain scores to the ones calculated on all data, it might not be necessary to use all adapted scores, but just adding one or two of the

four scores might be sufficient. Therefore, we will analyze for which scores we need an adapted version and for which of them we can just use the score from the general phrase table. In this case, for some of the scores, the features will then be defined as $\Phi_s =< \Phi_s^{All} >$ and not $\Phi_s =< \Phi_s^{All}, \Phi_s^{Adapted} >$.

In addition to removing some of the scores, we can also add scores. In the *Factored Approach*, we propose to use the probability of the domain given this phrase pair. This probability can be estimated by the number of times the phrase pair occurs in the domain divided by the absolute number of occurrences in the whole corpus.

Thirdly, the out-of-domain features and the in-domain features cannot be calculated for all phrase pairs, but only for the ones that occur in the corresponding corpus. If a phrase pair does not occur in the in-domain corpus, its in-domain probability is unknown. Therefore, we suggest different ways to handle unknown probabilities in the individual approaches.

Log-linear As it is done in the log-linear approach for domain adaptation, we can use either the in-domain or the out-domain scores and then use different scaling factors for each of them. That means both in-domain and out-of-domain phrase pairs have 8 scores. For in-domain phrase pairs the four out-of-domain phrase table features are set to 1 and for the out-of-domain phrase pairs the in-domain features are set to 1. For phrase pairs from the in-domain corpus this leads to the definition $\Phi_s^{Adapted} =< \Phi_s^{IN}, 1 >$ and for the out-of-domain phrase pairs we use the definition: $\Phi_s^{Adapted} =< 1, \Phi_s^{OUT} >$.

Back-off In the back-off method the in-domain scores are used for in-domain phrase pairs. For phrase pairs that only occur in the out-of-domain phrase table each score is set to the worst value that occurs in the in-domain phrase table for this score. In this case, for all phrase pairs from the in-domain corpus, we get the definition $\Phi_s^{Adapted} = \Phi_s^{IN}$ and for all the other phrase pairs:

$$\Phi_s^{Adapted} = min_{(\bar{e_i}|\bar{f_i})} \Phi_s^{IN}(\bar{e_i}|\bar{f_i})$$

Indicator An indicator feature signals whether the phrase stems from the in-domain or from the out-of-domain phrase table. As the first four scores we use the probabilities from the in-domain and out-of-domain phrase table, respectively, and the last one being the indicator feature. This additional feature will have

Table 6.10: Characteristics of the different phrase table adaptation approaches

Characteristic	Log-lin.	Back-off	Factored	Fill-up
Selection	UnionOut	NoAdapt	NoAdapt	UnionOut
General Scores		X	X	
Adapted Scores	all	2	2	all
Unk. Probability	Log-lin.	Back-off	Indicator	Indicator
Unique		X		X
Number of Scores	8	6	7	5

the value 1 for all in-domain phrase pairs and $exp(1)$ for all out-of-domain phrase pairs.

The fourth and last aspect is the treatment of phrase pairs which can be assigned both in-domain and out-of-domain scores, because they occur both in the in-domain and in the out-of-domain corpus. In this case, the back-off and fill-up approach suggest to use only the in-domain scores, while the other two approaches add the phrase pair to the phrase table twice, once with in-domain and once with out-of-domain scores.

An overview of the different aspects of the four approaches to phrase table adaptation as mentioned in the related work is given in Table 6.10.

6.3.4 Results

In Sections 6.3.2 and 6.3.3, we analyzed approaches to integrate additional data into the translation system. Thereby, we identified the things they have in common as well as their key differences. Now, we want to evaluate the influence of these differences on the translation quality.

We look at the tasks of translating TED lectures as well as the university lectures from German to English. First, we compare the different candidate selection strategies. Afterwards, we investigate the influence of the different scores on the translation quality.

6.3.4.1 Candidate Selection

In the first series of experiments, we analyze the influence of the candidate selection methods. Before considering the translation quality itself, we analyze the size of the phrase tables generated by the different methods.

Table 6.11: Number of phrase pairs generated by different candidate selection methods

	TED		CS	
	#PP	**%**	**#PP**	**%**
In	140K	40%	109K	31%
Out	335K	96%	338K	97%
All	348K	100%	347K	100%
UnionOut	425K	122%	408K	118%
UnionAll	413K	118%	399K	115%
PaddingOut	366K	105%	363K	105%
PaddingAll	364K	104%	361K	104%
SourcePadding	250K	72%	258K	74%

In Table 6.11 the number of phrase pairs selected by the different methods for two test sets are presented. The in-domain phrase table contains 30% to 40% of the phrase pairs that are in the general phrase table and the out-of-domain phrase table around 95%.

If we take the union of in-domain and out-of-domain (UnionOut) or in-domain and general phrase table (UnionAll), the size increases by around 20%. By using the Padding method, the phrase table increases only by around 5%, compared to the phrase table trained on all data. If we only use out-of-domain phrase pairs for source phrases, which did not occur in the in-domain corpus, the phrase table size is reduced by around 30%, compared to the general phrase table.

After looking at the phrase table sizes, we measure the quality of the translations generated using these phrase tables. We use the scores as described in the Back-off method. Since all phrase tables use the same features, we perform the first experiments without running separate optimizations for the different methods. The results for the TED translation task are shown in Table 6.12. For the task of translating computer science lectures the results are summarized in Table 6.13.

For the TED task we can only see small improvements by adapting the candidate selection. The best systems can improve by around 0.1 BLEU points over the system that does not perform any adaptation in the candidate selection step. The two Union methods as well as the Padding methods perform similarly well and lead to the best

Table 6.12: Candidate selection for the TED task (No optimization)

	Language Model Adaptation		
Candidate Selection	**No**	**Log-lin.**	**Linear**
NoCSAdapt	25.50	25.89	25.90
UnionOut	25.52	25.96	**25.96**
UnionAll	25.52	25.96	29.95
PaddingOut	**25.53**	**25.97**	25.94
PaddingAll	25.53	**25.97**	25.94
SourcePadding	25.50	25.90	25.76

results. The SourcePadding method leads to slightly worse results than the best two methods. The differences are bigger, if the language model is also adapted towards the target domain.

In the next experiments we analyze the performance on the task of translating university lectures. As mentioned before, in this task the training and test conditions do not match as well for the TED task. We analyze two conditions. In the first one, the systems were optimized on data from the TED corpus. In the second condition, better matching data from the computer science domain was used for optimizing the model weights.

In the first condition, the Union and Padding methods perform best. However, the Union method outperforms the Padding technique. We can again improve by around 0.1 BLEU point over the system using no adaptation in the candidate selection step. Furthermore, the SourcePadding technique performs worse than using no adaptation of the candidate selection step.

For the system that is optimized on the university lectures, the picture looks differently. Here, the system using no adaptation outperforms all adaptation approaches. The problem might be that in this condition the phrase tables differ more between the different approaches and using the same weights for all approaches hinders the performance of the adaptation approaches. Therefore, we perform additional experiments, where we optimized each approach individually. The results are summarized in Tables 6.14, 6.15 and 6.16.

However, we point out that in all conditions and for all methods it does not matter

Table 6.13: Candidate selection for the CS task (No optimization)

Candidate Selection	Language Model Adaptation					
	None		Log-lin.		Linear	
	TED	**CS**	**TED**	**CS**	**TED**	**CS**
NoCSAdapt	24.28	**25.31**	25.00	**25.40**	24.58	**25.56**
UnionOut	**24.38**	24.87	**25.09**	24.98	**24.64**	25.16
UnionAll	24.37	24.86	25.08	24.97	24.63	25.15
PaddingOut	24.29	24.85	24.99	24.96	**24.64**	25.15
PaddingAll	24.28	24.84	24.98	24.94	24.55	25.14
SourcePadding	24.16	24.76	24.81	24.88	24.44	25.08

Table 6.14: Candidate selection for the TED task

Candidate Selection	Language Model Adaptation					
	None		Log-lin.		Linear	
	Dev	**Test**	**Dev**	**Test**	**Dev**	**Test**
NoCSAdapt	28.03	25.50	28.40	25.89	28.27	25.90
UnionOut	28.14	25.42	28.43	25.96	28.61	25.90
UnionAll	**28.34**	**25.75**	**28.69**	26.04	28.43	26.01
PaddingOut	28.20	25.60	28.68	**26.09**	**28.66**	26.24
PaddingAll	28.19	25.40	28.53	**26.09**	28.61	26.12
SourcePadding	28.13	25.48	28.49	26.08	28.45	**26.29**

whether we combine the in-domain phrase table with the out-of-domain phrase table or with the phrase table trained on all data.

For the TED task (Table 6.14), we obtain the worst results in two of three conditions when performing no adaptation. Consequently, the adaptation seems to help. Furthermore, UnionAll and PaddingOut always perform better than the baseline.

In Table 6.15 we present the results on the university lectures when optimizing on the TED corpus. In this case, the situation is slightly different. Here again, SourcePadding is worse than the other two approaches to adapt the candidate selection step. Union produces the best translation in all cases and UnionAll is always better than the baseline.

Table 6.15: Candidate selection for the CS task (Dev on TED)

Candidate Selection	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
NoCSAdapt	28.03	24.28	28.40	25.00	28.27	24.58
UnionOut	28.14	23.97	28.43	25.09	28.61	**24.93**
UnionAll	**28.34**	**24.80**	**28.69**	**25.15**	28.43	24.60
PaddingOut	28.20	24.12	28.68	25.11	**28.66**	24.69
PaddingAll	28.19	24.04	28.53	24.56	28.61	24.28
SourcePadding	28.13	24.16	28.49	23.85	28.45	24.27

Table 6.16: Candidate selection for the CS task (Dev on CS)

Candidate Selection	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
NoCSAdapt	**27.29**	25.31	**27.50**	25.40	**27.58**	25.56
UnionOut	27.16	25.44	27.25	25.85	27.33	25.65
UnionAll	26.95	**25.48**	27.38	25.78	27.47	25.82
PaddingOut	27.10	**25.48**	27.24	25.87	27.40	25.60
PaddingAll	26.88	25.40	27.30	**25.92**	27.23	**25.88**
SourcePadding	26.88	25.21	27.17	25.56	27.20	25.79

Afterwards, we show the results where we optimized on university lectures (Table 6.16). Here, performing no adaptation is always best on the development data, but performance is very poor on the test set. The best performance is achieved by PaddingAll, but again the UnionAll method performs quite competitively.

Although performing individual optimizations for every configuration introduces additional random noise that might have influenced the results, it seems to be important to keep all phrase pairs from the in-domain phrase table. Using no adaptation of the candidate selection performs worse than the Union or the Padding approach in many experiments. Furthermore, especially in the case where the in-domain and test data do not perfectly match, i.e. with CS lectures, it seems also to be important to keep all

phrase pairs from the general or out-of-domain phrase table. Therefore, we will use the UnionAll method for all following experiments.

6.3.4.2 Selecting Scores for the Phrase Table

After analyzing the influence of the candidate selection, the remaining experiments concentrate on the features that can be used as scores for the phrase table entries. In the first group of experiments we analyze which ones of the adapted features contribute to an improved translation quality.

In all systems we use the four scores from the phrase table trained on all data. Then we add one or more of the in-domain scores. We use the in-domain features as described in the back-off method. Again, the results for the three conditions, TED translation task and the task of translating computer science lectures without and with matching development data are shown in the three Tables 6.17, 6.18 and 6.19.

For the TED task, in most conditions adding each single or combinations of features could improve over the baseline system using no adaptation. On average, the relative frequencies and the domain probability contributed most to the improvements, while the lexical features could only improve sightly over the baseline system. Averaged over all three configurations, the best performance was achieved by adding the relative frequencies and the domain probabilities, which led to an increase of around 1 BLEU point over the baseline system. Similar performance is achieved by the system using relative frequency and lexical features and the system using only relative frequencies.

The two conditions translating computer science lectures present similar results. This time the best performance is achieved by using only the relative frequencies for the condition with non-matching development data and using only one relative frequency in the other condition. Furthermore, when using the non-matching development data, not all systems can outperform the baseline system. Again the relative frequencies seem to be very important, while the domain probability has less impact in this condition.

In summary, we evaluated the different phrase table features under nine different conditions. In average over all conditions, the system using both relative frequencies performs best. In the condition where no language model adaptation is performed, this is also the best method. For the configuration using log-linear language model adaptation, the best performance is achieved by using only one relative frequency. In

Table 6.17: Feature selection on the TED task

Feature Selection	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.51	24.36	27.56	25.43	27.65	25.59
rel. Freq 1	28.04	25.32	28.50	25.82	28.37	26.07
rel. Freq 2	28.28	25.54	28.44	25.88	**28.56**	**26.27**
rel. Freq 1&2	28.34	25.75	**28.69**	26.04	28.43	26.01
Lex 1	27.87	25.08	28.40	25.66	28.17	25.84
Lex 2	27.73	25.35	28.42	25.89	28.41	25.65
Lex 1&2	27.47	25.23	28.22	25.53	28.08	25.40
rel. Freq 1&2 + Lex 1&2	28.28	25.63	28.46	26.10	28.43	26.10
Domain	27.95	25.74	28.30	25.93	28.21	25.80
rel. Freq 1&2 + Domain	28.44	**25.79**	28.67	**26.38**	28.48	**26.27**
All	**28.46**	25.71	28.35	25.87	28.52	26.09

Table 6.18: Feature selection on the CS task (Dev on TED)

Feature Selection	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.51	23.84	27.56	24.75	27.65	24.02
rel. Freq 1	28.04	24.25	28.50	25.07	28.37	24.32
rel. Freq 2	28.28	24.29	28.44	23.54	**28.56**	24.19
rel. Freq 1&2	28.34	**24.80**	**28.69**	**25.15**	28.43	**24.60**
Lex 1	27.87	23.92	28.40	25.06	28.17	24.25
Lex 2	27.73	24.22	28.42	24.36	28.41	24.53
Lex 1&2	27.47	23.98	28.22	23.76	28.08	23.87
rel. Freq 1&2 + Lex 1&2	28.28	23.69	28.46	24.50	28.43	24.11
Domain	27.95	24.29	28.30	24.84	28.21	23.75
rel. Freq 1&2 + Domain	28.44	24.10	28.67	24.48	28.48	24.13
All	**28.46**	23.96	28.35	24.80	28.52	23.96

Table 6.19: Feature selection on the CS task (Dev on CS)

Feature Selection	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.37	24.17	26.97	25.42	26.88	25.18
rel. Freq 1	26.91	25.58	27.31	**26.14**	27.38	25.91
rel. Freq 2	26.87	**25.75**	27.25	25.64	27.42	25.65
rel. Freq 1&2	26.95	25.48	**27.38**	25.78	**27.47**	25.82
Lex 1	26.90	25.24	26.98	24.69	27.19	25.56
Lex 2	26.71	24.87	27.03	25.85	26.98	25.23
Lex 1&2	26.65	25.11	27.15	25.83	26.85	24.62
rel. Freq 1&2 + Lex 1&2	26.96	25.47	26.97	25.35	27.43	25.73
Domain	26.70	25.14	27.15	25.51	26.99	25.51
rel. Freq 1&2 + Domain	**26.99**	25.46	27.08	25.74	27.49	**26.05**
All	26.82	25.14	26.86	25.34	27.10	25.04

contrast, if we use linear language model adaptation, the best performance is reached by using the relative frequencies together with the domain probability.

In conclusion, the relative frequencies seem to be the most important feature for adaptation. In some cases, especially when the development and test data match, the domain probability can help to improve the performance further. By using both relative frequencies, we can improve by 1 to 1.4 BLEU compared to using no adaptation, if no language model adaptation is used and by 0.4 to 0.6 BLEU points if language model adaptation is used. In both tasks, these improvements are bigger than the ones gained by adapting the candidate selection process.

In Section 6.3.3 we mentioned two additional aspects of selecting the scores of the phrase pairs. If the phrase pair occurs both in the in-domain and out-of-domain corpus, we can calculate the adapted scores according to the definition for the in-domain or out-of-domain phrase pairs for all approaches to except for the back-off approach. Then we can either use only the phrase pairs with the scores generated by the in-domain data or add the phrase pair to the translation model twice: Once with the indomain scores and once with the out-of-domain scores. We preform preliminary experiments, but we could not find any significant difference between the two approaches. Therefore, we

always use two phrase table entries, one based on the in-domain scores, and one based on the out-of-domain scores. Now we concentrated on the other two aspects of the translation model adaptation: whether to include the general scores in the in-domain phrase table entry in addition to the in-domain scores and how to deal with unknown probabilities.

Since the number and types of features is different between those experiments, separate optimizations had to be run. In all experiments we use the UnionAll method as candidate selection and use two sets of features for one phrase pair, if the phrase occurs in both the in-domain and out-of-domain corpus.

The results are shown in Tables 6.20, 6.21 and 6.22 for the three different conditions. The first system in each table uses no adapted features at all. The next two systems use only the adapted features using the indicator and log-linear method to handle unknown probabilities. The back-off methods can only be used in combination with the general scores. Otherwise all out-of-domain phrase pairs have the same features. The remaining nine systems use both the general scores and the adapted ones. Out of these systems, the first three systems use both adapted relative frequencies as well as both adapted lexical probabilities. The next three systems (*General + rel. Back-off*, *General + rel. Log-lin.*, *General + rel. Indicator*) use only the adapted relative frequencies. The final three systems use the adapted relative frequencies as well as the domain probability (Table: *General + rel. D.*).

The results for the TED translation task presented in Table 6.20, show a similar performance for the different features.. The maximal average difference between the approaches is 0.4 BLEU points. Furthermore, all approaches can clearly outperform the baseline system. The best result is achieved with the *General + Log-lin.* combination. The reason for this may be that this approach uses the largest number of features, which results in more dimensions for the adaptation towards the target domain.

Since the systems using no general scores (*Log-lin.* and *Indicator*) performs in average second and fourth best, the usage of the general scores seems not to be very important for this task. Among the three best systems using the General scores each of the three approaches (Back-off, Log-linear and Indicator) are represented. Furthermore, one of them uses all adapted features, one uses only the relative frequencies and one uses the relative frequencies and the domain probability. Consequently, when have a perfect match between the training and test data, it seems to be important to use an

Table 6.20: Feature combination for the TED task

Feature Combination	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
No	26.51	24.36	27.56	25.43	27.65	25.59
Log-lin.	28.26	25.72	28.63	26.17	28.18	**26.54**
Indicator	28.23	25.78	28.31	**26.51**	28.33	25.99
General + Back-off	28.28	25.63	28.46	26.10	28.43	26.10
General + Log-lin.	**28.52**	**25.88**	28.60	26.25	**28.68**	26.44
General + Indicator	28.34	24.49	28.40	26.09	28.31	26.27
General + rel. Back-off	28.34	25.75	**28.69**	26.04	28.43	26.01
General + rel. Log-lin.	28.23	25.65	28.61	26.14	28.40	25.99
General + rel. Indicator	28.40	25.83	28.48	26.18	28.53	26.18
General + rel. D. Back-off	28.44	25.79	28.67	26.36	28.48	26.27
General + rel. D. Log-lin.	28.24	25.48	27.81	25.66	28.49	25.97
General + rel. D. Indicator	28.46	25.67	28.48	26.28	28.49	26.23

approach to perform phrase table adaptation, but it is not so important which one to use.

Table 6.21 shows the results for the task of CS lectures translation without matching development data. For this task not all features could improve over the baseline system using no in-domain features. It seems to be harder to gain improvements through using phrase table adaptation, if the test domain does not match the in-domain data perfectly. Nonetheless, the best approaches could improve the average performance by 0.8 BLEU points.

Especially, the two approaches using no general scores perform worse than the baseline system. Using the scores from the general phrase table seems to help the performance in cases where the data does not match perfectly . Furthermore, the approaches using fewer adaptive scores perform best in this condition. The best average performance was achieved by the *General + rel. Back-off* approach followed by the *General + rel. Indicator* approach.

Table 6.22 presents the last condition, where we use both development and test data from the computer science lecture domain. This time all approaches outper-

Table 6.21: Feature combination for the CS task (Dev on TED)

Feature Combination	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
No	26.51	23.84	27.56	24.75	27.65	24.02
Log-lin.	28.26	23.93	28.63	24.27	28.18	23.88
Indicator	28.23	23.52	28.31	24.66	28.33	23.42
General + Back-off	28.28	23.69	28.46	24.50	28.43	24.11
General + Log-lin.	**28.52**	24.01	28.60	24.39	**28.68**	24.86
General + Indicator	28.34	25.54	28.40	25.05	28.31	24.29
General + rel. Back-off	28.34	24.80	**28.69**	25.15	28.43	24.60
General + rel. Log-lin.	28.23	24.40	28.61	24.86	28.40	24.73
General + rel. Indicator	28.40	25.05	28.48	24.94	28.53	24.49
General + rel. D. Back-off	28.44	24.10	28.67	24.48	28.48	24.13
General + rel. D. Log-lin.	28.24	23.81	27.81	24.88	28.49	24.46
General + rel. D. Indicator	28.46	24.96	28.48	24.73	28.49	24.77

Table 6.22: Feature combination for the CS task (Dev on CS)

Feature Combination	Language Model Adaptation					
	No		Log-lin.		Linear	
	Dev	Test	Dev	Test	Dev	Test
No	26.37	24.17	26.97	25.42	26.88	25.18
Log-lin.	27.42	25.23	25.91	24.45	27.24	25.30
Indicator	27.34	25.36	27.06	24.99	**27.62**	25.27
General + Back-off	26.96	25.47	26.97	25.35	27.43	25.73
General + Log-lin.	27.28	**25.57**	27.24	25.80	27.33	25.64
General + Indicator	**27.44**	25.45	26.90	25.29	27.04	25.30
General + rel. Back-off	26.95	25.48	27.38	25.78	27.47	25.82
General + rel. Log-lin.	26.60	24.77	27.43	25.73	27.50	25.74
General + rel. Indicator	26.96	25.51	**27.50**	**26.11**	27.22	25.47
General + rel. D. Back-off	26.99	25.46	27.08	25.74	27.49	**26.05**
General + rel. D. Log-lin.	26.89	25.36	27.03	25.74	26.93	24.99
General + rel. D. Indicator	26.95	25.43	27.11	25.42	27.02	25.02

Table 6.23: Overview of results using data weighting

Task	No Adapt		Adapt		Gain	
	Dev	Test	Dev	Test	Dev	Test
TED	26.32	24.24	28.48	26.18	+2.16	+1.94
CS (TED Dev)		23.60		24.94		+ 1.34
CS	26.24	24.38	27.50	26.11	+1.26	+1.73

form the baseline approach. But again, the lowest improvements were achieved by the systems using no general scores. Thus, the general scores seem to be important in conditions when the training and test data do not match perfectly. Again, the average improvement of the best system is 0.8 BLEU points over the baseline system. The best approaches in this condition are *General + rel. D. Back-off* and *General + rel. Indicator*.

In both computer science conditions it seems to be best to use either the relative frequencies or the relative frequencies and the domain probabilities for adaptation. Furthermore, the Back-off and Indicator approaches seem to slightly outperform the Log-linear approach. Averaged over all conditions the best performance was achieved by the approaches *General + rel. Indicator* and *General + rel. Back-off*.

6.3.4.3 Combining In-Domain and Out-of-Domain Data

After analyzing the different approaches of adapting the system using in-domain training data in the last part in detail, we will have a final look at how much improvements can be achieved by performing data weighting. Therefore, we compared the results of the previous chapter with the ones also using adaptation by model combination. We used the best performing approach. That means, we perform log-linear language model adaptation and translation model adaptation. For the translation model adaptation, we used the *Union* approach for candidate selection and calculated the scores according to the *General + rel. Indicator* method. The results are summarized in Table 6.23. Furthermore, we present example sentences in Tables 6.1 and 6.2.

As shown in the table, the performance could be improved significantly in all tasks. The best improvements were achieved in the TED task by around 2 BLEU points. For

the computer science lectures task the results could be improved by 1.3 to 1.7 BLEU points depending on whether matching development data is available.

In the first example in Example 6.1 the German word *Schiffs* is no longer translated into *vessel*, but into *ship*. Although the former is common in the proceedings of the European Union, in the scenario of translating general talks *ship* is a more suitable translation. In the second translation example for the TED task the German word *vorstellen* is used. In the context, the word means *sich etwas vorstellen* (engl. *to imagine*), while the baseline system uses the translation for *etwas vorstellen* (engl. *to present*). Although this is a common meaning in the out-of-domain data, with the adaptation it is possible to generate the correct translation in the given context.

Source:	... der Baupläne des **Schiffs**.
Reference:	... on the blueprints of the **ship**.
No Adapt:	... the design of the **vessel**.
Adapt:	... the design of the **ship**.
Source:	wir wollen uns das best mögliche Ergebnis **vorstellen**.
Reference:	we want to **imagine** the best case scenario outcome
No Adapt:	we want to **present** the best possible result.
Adapt:	we want to **imagine** the best possible result.

Example 6.1: Examples from the TED task

The third example stems from the university lectures task. Due to the adaptation the phrase *zufällige Gewichte* is no longer translated into *adventitious weights*, but correctly into *random weights*.

Source:	wir haben **zufällige** Gewichte.
Reference:	we have **random** weights.
No Adapt:	we have **adventitious** weights.
Adapt:	We have **random** weights.

Example 6.2: Example from the CS task

7

Adapting towards Specific Topics

In the last chapter we developed different ways to adapt the translation system towards a new application using small amounts of in-domain data. In these approaches, we did not distinguish between different aspects of the training data. For example, we did not differentiate between an adaptation towards a new topic or towards a new genre. In contrast, we tried to better exploit data that is more appropriate for the application.

In real-world applications, it is quite common that we may need to adapt our system towards some special aspects of the task. For example, after performing the adaptation approaches mentioned before, we have a reasonably good translation system for translating TED lectures. But if we now need to adapt this system to translate computer science lectures or electrical engineering lectures, we would need special adaptation techniques to adapt the translation system towards a new topic. In this chapter, we will develop these types of techniques.

As described in Chapter 1.3, the topic of a text describes its content. The content can often be described by a hierarchy of topics. While talks such as TED lectures usually have a more general topic like the environment or music, the topics in university lectures can be much more specific. The general topic of the university lectures used in this thesis is computer science. The more specific topic of the lectures is cognitive systems or speech recognition. Some part of the lecture can be about a very specific topic such as the HMM algorithm.

In this chapter, we will first present which problems occur when a system is used for translating text belonging to a very specific topic that is different from the topics of the training data. We analyze how the problems differ in the case of more general

topics of TED talks versus the more specific ones of university lectures. We will see that more specific topics lead to additional problems.

An intuitive approach to tackle these problems would be to collect parallel data that matches the application in topic and genre and use approaches described in the last chapter to integrate this data into the translation system. But if we look at the example of translating university lectures, we notice that we will often not be able to collect this data. There is nearly no publicly available German to English parallel data about computer science. Sometimes we would even need a more specific topic. For example, it would be good to have parallel data about speech recognition. In this case there is nearly no chance of finding any appropriate parallel data.

Since it is rather difficult to acquire useful parallel data, we will concentrate on ways of using other knowledge sources to adapt the system towards the new topic. Wikipedia[1] is a very promising source of information, since it already contains several million articles in 285 languages and is continuously growing. We will try to integrate this knowledge source into our translation system as shown in Niehues and Waibel (2011).

In the case of Wikipedia, we use the titles of lined articles in different languages, since the articles themselves are not parallel. In this data the words only occur as lemmas in most cases. This is also the case for many other additional knowledge sources such as dictionaries. It is especially problematic when we use German as the input language since German is a highly inflected language. We will therefore try to learn translations for inflected word forms from the translations of the lemmas.

7.1 Motivation

In addition to the problems that occur when adapting a model in a machine learning task, SMT-specific problems exist. In many other tasks there is a fixed set of labels $\mathcal{Y}$ and the main problem is to adapt the model in such a way that it selects better fitting labels. In SMT the labels of the prediction tasks are the target words. For example, if we talk about a *Keller* in computer science, we most likely do not mean the *basement* but a *stack*.

[1] http://www.wikipedia.org

Table 7.1: Out-of-vocabulary words of the different tasks

Task	Condition	OOV			mod. OOV		
		total	unique	rate	total	unique	rate
TED	Baseline	422	314	1.34%	307	245	0.98%
	+ TED Training	334	255	1.06%	255	205	0.81%
CS	Baseline	1437	624	3.22%	1150	521	2.58%
	+ TED Training	1219	563	2.73%	959	467	2.15%
	Lectures Training	924	492	2.07%	749	416	1.68%

Merely learning better target words is not sufficient for SMT since we also need to learn new translations for terms that are specific to a topic. That means we need to learn a new set of target labels $\mathcal{Y}$. This is especially the case when adapting the translation system towards a new topic. For example, if we want to translate computer science lectures, we also need to learn translations for terms such as *sampling* or *quantization*.

7.1.1 Out-of-Vocabulary Words

In order to tackle the problems occurring when adapting an SMT system towards a new topic, we first want to analyze the problem of topic specific terms in these conditions. If we have got a topic specific term, we will most probably have no translation for it. In this case, it will be treated as an out-of-vocabulary (OOV) word. In a first step, we will analyze the number of OOV words in the different test conditions.

The default method to handle OOV words when translating European languages is to pass them through. Since many of the OOV words are proper names, which will be the same in both languages, this will often lead to a good translation. In a second step, we will evaluate how well this default strategy performs by looking at the modified number of OOV words. In this case, a word is only counted as an OOV word if it does not occur in the reference translation. For all the remaining OOV words the default approach does not lead to the correct translations.

We summarized the number of OOV words for the different conditions of the two tasks in Table 7.1. These numbers were already given in Tables 4.4 and 4.5 in Chapter 4. Using the methods described in the last two chapters we could improve the translation quality in the different conditions significantly. But the approaches do not attempt to

learn new translations. Therefore, the number of OOV words is still the same as in the baseline systems.

The number of OOV words shows the different characteristic of both task. In the TED lectures, we have common language. Therefore the number of OOV words is quite small, resulting in an OOV rate of 1.34% or 1.06% depending on whether the TED data is used. For the computer science lectures translation task, the picture is different. In this case, the vocabulary used by the lecturer is much more specific. Therefore, the OOV rate nearly tripled to 3.22% and 2.73%.

Furthermore, these words are often very important to understand the content of the lecture. While it may not be problematic if 3% of the functions words are not translated correctly, these words often convey the content of the sentence. Therefore, the sentence can only be understood correctly, if these words are translated correctly.

Since this problem will always occur when we use the lecture translation system for a new application, it is important to develop methods that are able to learn these words automatically.

7.2 Integration of Additional Knowledge Sources

In this thesis we will investigate methods to integrate additional knowledge sources to be able to learn OOV words. Although most of the methods presented here are quite generic and can be easily adapted to new knowledge sources (Cho et al., 2013), we will concentrate on Wikipedia as an example knowledge source.

After a short introduction of the important aspects of Wikipedia that are needed for this work, we will describe the creation of a lexicon that can be included into the translation system. Then, we will describe a technique to disambiguate different meanings of the same word. Finally, we will evaluate the approach on both translation tasks.

7.2.1 Wikipedia

Wikipedia is the most commonly known online encyclopedia containing articles about nearly every subject. These articles are written collaboratively by volunteers. The English version has currently 4.1 million articles.

Wikipedia is even more interesting for machine translation since it is a multi-lingual project. Currently, there are versions in 285 languages available. Of course, most of them are very small compared to the English version. However, five more languages have more than one million articles.

The different language editions of Wikipedia are developed almost independently. Articles about the same subject in two languages are almost never a direct translation of each other and there is therefore no straight forward integration of the knowledge into an SMT system.

For machine translation one very interesting feature of Wikipedia editions in different languages are the so-called inter-language links. These links connect pages in different languages about the same topic. Therefore, we can learn the translation for a concept by following the link from the source language article to the target language.

7.2.2 Related Work

As mentioned in Chapter 3, several approaches to integrate additional resources into a translation system have been presented in recent years. Wikipedia has already been shown to be a valuable resource for natural language processing. For example, Erdmann et al. (2008) proposed to extract bilingual terms from the Wikipedia titles and Yu and Tsujii (2009) use terms from a comparable corpus created from Wikipedia articles using the inter-language links.

De Melo and Weikum (2010) tried to extract multilingual concepts by exploiting the inter-lingual link structure of Wikipedia. They described an approximation algorithm to correct misleading inter-lingual links. In contrast to the work here, their algorithm is much more complex and needs the inter-lingual links of all Wikipedia editions as opposed to only the links between two languages.

7.2.3 Lexicon Creation

Since articles are not translations, we cannot use them as parallel data, but we are able to make use of the inter-language links. As mentioned before, these links indicate that two articles are about the same subject. Therefore, the titles of linked articles will probably be translations.

The first step to create a corpus of translations of Wikipedia titles is to extract the links from the articles and generate an alignment between articles. As a result, we get

two alignments, since there are links in the source language article to a target language article and links in the target language article to a source language article.

The next step in generating the parallel corpus is to symmetrizing this alignment. It is not always the case that there is a link in both directions. Sometimes a source language article is aligned to a target language article, which is not aligned to any source language article or which is aligned to a different source language article. The main reason for this is that both articles are not about the same topic, but merely closely related. In some cases, the link directs to a paragraph of another article, which is itself about a more general topic. Another option is that the article is directly linked to a more general article since there is no equivalent one in the other language.

In contrast to other parallel data, most phrases occur only once in the corpus of Wikipedia titles. Therefore, we cannot calculate reliable statistics such as translation probabilities and consequently it is very important that the corpus is of high quality. Therefore we use the intersection of both alignments in order to get only high quality translations, discarding unreliable links.

7.2.3.1 Disambiguation Pages

The main problem of creating this corpus are the word ambiguities. The German word *Bank*, for example, translates to *bank*, if the financial institution is meant and if the furniture is meant, the translation is *bench*.

In Wikipedia, this is modeled in the following way: There is a disambiguation page for *Bank* with links to the different articles. Then there are separate articles, one for the financial institution and one for the furniture. Typically, the disambiguation page in one language for *bank* is linked to the disambiguation page in the other language. The links between articles give us helpful translations, but the links between the disambiguation pages may be misleading. To avoid problems with these links and to acquire a high quality corpus, we ignore all inter-language links between disambiguation pages. We only use the inter-language links of the articles the disambiguation page is linking to. Since the disambiguation pages are marked by a tag, the identification of these pages is straight forward.

7.2.3.2 Integration

After acquiring the corpus, we can apply the same preprocessing as for the parallel training corpus of the translation system. Since all content words are often written in upper case in the titles, we case each word as it is most often cased in the Wikipedia corpus.

The baseline approach is to use the corpus of Wikipedia titles as a lexicon. Therefore, every pair of titles is used as an additional phrase pair in the phrase table.

By integrating the data in this way, problems with the titles of the ambiguous expressions occur. For example, the title for the *bench* in German is *Bank (Möbel)* (engl.: *bench (furniture)*). During translation we will of course only find the word *bench* alone without the additional domain information in parentheses. Therefore, this phrase pair will almost never match. To bypass this problem, we tried a second approach using the corpus in the same way as a normal training corpus. We trained a GIZA++ alignment on the corpus and then performed phrase extraction. Afterwards, we added the additional phrase pairs to the general phrase table.

7.2.4 Article Disambiguation

As mentioned before, there are separate articles for each meaning of an ambiguous word. This means we have multiple translation options in our phrase table. If we want to apply the translations learned from Wikipedia during the translation, we need to decide which of the titles to use. This is especially problematic for terms that are also surnames of famous persons. For example, the German word for *cone* is *Kegel*, but there are also different articles about persons named *Kegel*. Therefore, we might translate the German word *Kegel* into *Kegel*, since we are using the translations learned from the Wikipedia tiles about the person.

The default approach in statistical machine translation is to calculate the translation probabilities. In our case, however, this would not be very helpful. In case there is more than one person called *Kegel*, the translation of *Kegel* into *Kegel* would have a high probability, since we extracted the phrase pair from several articles.

Instead of relying on relative frequencies, we use additional information to select the best translation from our parallel Wikipedia title corpus. In addition to the translation of the title, we also use the articles themselves by comparing the source article to our

test set. If the article is similar to the test set, the probability that the translation of the title will be a good translation is higher than the probability of a translation extracted from an article about a completely different topic. If we want to translate a computer science lecture, the source text will be more similar to an article about a cone than to one about an author or musician called *Kegel*. Therefore, we use the similarity between the article and the source document as an additional feature.

To measure the similarity between the article and the document that needs to be translated, we represent the Wikipedia article as well as the test documents in the TF-IDF vector space. Then we calculate the cosine similarity to compare the articles. Afterwards, for every unknown source word, only phrase pairs extracted from the title of the article with the highest similarity are considered.

7.2.5 Results

After presenting the approach to include Wikipedia as an additional data source into the machine translation system, we evaluate its performance on the applications mentioned in the last chapters. We measured improvements in translation quality in BLEU, but also analyzed the out-of-vocabulary words. Since the main idea was to find translations for the topic-specific terms, we investigate if we could translate more words using this approach.

7.2.5.1 Integration of Wikipedia

In a first series of experiments we analyzed the difference between the two approaches to integrate knowledge from Wikipedia into the translation system. The number of out-of-vocabulary words for the different development and test sets are summarized in Table 7.2. The number of OOV words of the systems were calculated for the baseline system using no TED training data, but the tendency for the other systems is the same. Since we did not use the development data in this approach for any training and we did not re-optimize the systems, in these experiments they can be considered as additional test sets.

The baseline system does not use Wikipedia. In the lexicon system the Wikipedia titles are integrated as a *lexicon* as described in Section 7.2.3.2. Finally, we evaluated the effect of the Wikipedia titles when integrated as an additional training corpus.

Table 7.2: Integration of Wikipedia data

Data	System	OOV		mod. OOV	
		total	unique	total	unique
EPPS Dev	Baseline	153	143	136	127
	Lexicon	146	136	133	124
	Corpus	135	125	126	117
TED Dev	Baseline	339	271	250	231
	Lexicon	287	234	225	195
	Corpus	230	192	169	173
TED Test	Baseline	422	314	307	245
	Lexicon	373	286	282	227
	Corpus	276	225	247	119
CS Dev	Baseline	1095	378	891	239
	Lexicon	992	359	801	312
	Corpus	601	315	542	279
CS Test	Baseline	1437	624	1150	521
	Lexicon	1236	584	958	483
	Corpus	1007	501	825	429

For the first set, the EPPS development set, the OOV rate is already very small for the baseline system and consequently not much can be gained by the additional data. Since we train the systems on the big EPPS data, this set is already well covered by the baseline system.

For the TED sets, the OOV rate is higher than for EPPS. However, the OOV rate can be reduced. The *lexicon* approach reduces the OOV rate by 10% and the *corpus* approach can even reduce it by 30%. In order to check if these translations are needed, we calculated the modified OOV rate, where only those words are counted as OOVs that cannot be passed through to generate a correct translation. As shown in the table, the number of modified OOV words could also be reduced similarly.

Finally, the table shows the results for the computer science development and test sets. Here the OOVs are a big problem due to the quite specific topic. In this case the number of OOV words could be again reduced by around 10% by the approach using the lexicon. The *corpus* approach could reduce the number of OOV words even more by up to 45%. Again, similar improvement could be achieved for the modified OOVs. One example where the *corpus* approach performs better than the *lexicon* approach is the German word *Abtastung* (engl. *sampling*). This can only be translated using the *corpus* method since the Wikipedia article is called *Abtastung (Signalverarbeitung)* (engl. *sampling (signal processing)*). If we look at the unique OOVs, we can see that the method is mainly able to translate the OOVs that occur several times.

In summary, the number of OOVs can be reduced substantially by the presented approaches on all the sets where they present a problem. In all cases, integrating the Wikipedia titles as a corpus instead of an lexicon reduces the number of OOVs further. The impact of the presented approaches on translation quality is shown in Tables 7.3 and 7.4.

For all the experiments on translation quality, we did not perform any re-optimization. In the *Baseline* system, we use the EPPS development data. The translation quality could not be improved on the EPPS development data by using Wikipedia. This is not surprising, since we already saw earlier that we do not have a lot of OOV words. For the TED data, we could improve in all conditions slightly by 0.05 to 0.1 BLEU points. Although the *corpus* approach is able to translate more words than the *lexicon* method, the translation quality of both approaches is similar. The reason for this may be, that

Table 7.3: Different integration strategies on the TED task

Condition	Baseline		Lexicon		Corpus	
	Dev	**Test**	**Dev**	**Test**	**Dev**	**Test**
Baseline	27.51	21.87	27.51	21.96	27.52	21.98
+ TED Dev	22.79	22.36	22.82	22.43	22.89	22.41
+ TED Training	26.59	26.18	26.63	26.22	26.65	26.18

Table 7.4: Different integration strategies on the CS task

Condition	Baseline		Lexicon		Corpus	
	Dev	**Test**	**Dev**	**Test**	**Dev**	**Test**
Baseline	27.51	20.97	27.51	21.27	27.52	21.38
+ TED Dev	22.79	22.44	22.82	22.78	22.89	22.93
+ TED Training	26.59	24.94	26.63	25.22	26.65	25.33
CS Dev	24.59	23.21	24.88	23.49	25.60	23.66
+ TED Training	26.22	26.11	26.31	26.30	27.20	26.51

the learned translations are not as reliable as the ones from the lexicon approach. The phrase extraction may introduce additional errors.

As expected given the results on the OOV words, the improvements on the CS tasks are larger than the ones for the other tasks. In this case the translation quality could be improved by up to 1 BLEU point on the development data and by 0.5 BLEU points on the test data. Furthermore, it seems to be more important to have many translation options rather than having only the high precision ones, because the *corpus* approach is always 0.1 to 0.2 BLEU points better than the *lexicon* approach.

7.2.5.2 TF-IDF

For the all the sets we calculated the cosine similarity of the TF-IDF vectors to all Wikipedia articles. We then used this score in addition to the four default scores for the phrase pairs. When a phrase pair was extracted from different articles, the pair was kept only once and received the similarity score of the most similar article. We excluded the 10K most frequent words from the calculation of the TF-IDF score.

Table 7.5: Wikipedia integration using document similarity

Task	Condition	No TF-IDF		TF-IDF	
		Dev	**Test**	**Dev**	**Test**
TED	Baseline	27.52	21.98	27.51	21.94
	+ TED Dev	22.89	22.41	22.85	22.40
	+ TED Training	26.65	26.18	26.65	26.18
CS	Baseline	27.52	21.38	27.51	21.39
	+ TED Dev	22.89	22.93	22.85	22.94
	+ TED Training	26.65	25.33	26.65	25.33
	CS Dev	25.60	23.66	25.61	23.67
	+ TED Training	27.20	26.51	27.17	26.49

As shown in Table 7.5, using the TF-IDF score of the article as an additional feature did not improve the translation quality as measured in BLEU. However, in some cases the translation could be improved. For example, the German word *Kegel* (engl. *cone*) is no longer translated into *Kegel* because of an article about a person named *Kegel* but correctly into *cone* because only the translation into *cone* is kept in the phrase table due to the similarity check. However, these are only a few examples. In some cases choosing a phrase originating from a more similar article over the phrase with the highest probability even leads to a wrong translation, because of an erroneous alignment between the words in the Wikipedia titles.

7.3 Quasi-Morphological Operations

Since we are using only the titles of the articles and not the articles themselves, in most cases the words occur only as lemmas. In the test set, however, we also see other morphological forms like genitive or plural forms. This is especially problematic when using languages like German as input language, which are highly inflective.

In the following, we will first motivate our approach to translate inflected words forms using the translations of lemmas. Afterwards, we will describe the method. Then we will describe the training and the integration of the quasi-morphological operations. In the end, we will evaluate the operations in the translation task.

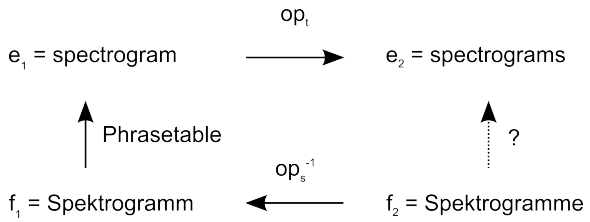

Figure 7.1: Quasi-morphological operations

7.3.1 Motivation

Even when we use Wikipedia, our system might be able to translate the German word *Spektrogramm* (engl. *spectrogram*), but not the plural form *Spektrogramme*. Furthermore, if we can translate *trigonometrische Funktion* using the additional knowledge source, we are not necessarily able to translate the phrase in dative case *von trigonometrischen Funktionen* because the titles of Wikipedia articles usually contain only the lemma, as it is the case for most encyclopedias.

To address this problem, we try to automatically learn rules on how words can be modified. If we look at the first example, we would like the system to learn the following rule. If an *"e"* is appended to the German word, as it is done when creating the Plural form of *Spektogramm*, we need to add an *"s"* to the end of the English word in order to perform the same morphological word transformation. Or in the second example adding an *"n"* to the German adjective represents its transformation into the dative case. For the English equivalent, no transformation needs to be applied since cases do not have a distinct surface word form.

Depending on the part-of-speech, number, gender and/or case of the involved words, the same operation on the source side does not necessarily correspond to the same operation on the target side.

To account for this ambiguity, we use additional information for selecting the target operation. For instance, we should not generate target words that do not exist. Here we have the advantage that we can use monolingual data to determine whether a word exists. Furthermore, we use the endings of the source and target word to determine which pair of operations should be used.

An advantage when translating from German to English, i.e. from a morphologically complex to a less complex language, is that many alternative word forms in German

map to the same English words. This means that we can ignore the target context in many cases when generating the English target word. In contrast, when we translate in the other direction, additional context information would be needed to decide on one of the possible German word forms. This problem is much more complex and has gathered a lot of interest in the machine translation community (c.f. Weller et al., 2013).

7.3.2 Related Work

In recent years, there have been several attempts to model morphology more explicitly in machine translation. Toutanova et al. (2008) showed improvements translating into morphologically rich languages by integrating an additional morphological model using a maximum entropy model introduced by Minkov and Toutanova (2007). Bojar and Tamchyna (2011) tried to handle morphological problems by using reverse self training.

Macherey et al. (2011) tried to learn morphological operations for parts of the translation system. They could improve compound splitting by learning morphological operations for the compound parts.

Our approach to learn the morphological transformations is similar to the Paradigmatic Cascades model learning pronunciation rules for unknown words presented in Yvon (1997). In their framework, operations that change the unknown word into a known word are learned. Since they do not only want to learn the rules for different morphological forms of the same word, but also for completely new words, more complex rules that also change the prefix of the word need to be learned. Furthermore, they apply the rules recursively to the unknown word. This makes their approach more powerful than ours, but also much more computationally complex.

7.3.3 Approach

Given two similar source words f_1 and f_2, an operation op_s that transforms f_1 into f_2 and a target word e_1 which is a translation of f_1, we want to find a target word e_2, which is a translation of f_2. To achieve this we try to find an operation op_t given the source operation op_s and the target word e_1, so that we can generate the target word e_2 by:

$$e_2 = op_t(e_1) \tag{7.1}$$

As shown in Figure 7.1, the complete procedure can be modeled in the following way: To find the translation for an unknown word f_2, we apply the inverse source operation op_s^{-1} to obtain f_1. Then we translate f_1 and apply the target operation op_t to e_1:

$$e_2 = trans(f_2) = op_t(trans(op_s^{-1}(f_2))) \tag{7.2}$$

As a result we obtain e_2, a translation of f_2 which we were not able to translate before.

If we look at the example above, we have the source words $f_1 =$ *Spektrogramm* and $f_2 =$ *Spektrogramme* and the source operation to append an *"e"* $op_s = (\text{“”} \rightarrow \text{“}e\text{”})$. The target word f_1 would then be *spectrogram*. To get the correct translation of f_2, we need to learn that the best target operation op_t in this case is to append an *"s"* $op_t = (\text{“”} \rightarrow \text{“}s\text{”})$. Then we can apply the target operation on *spectrogram* and obtain the correct translation $e_2 =$ *spectrograms*.

To be able to apply the model to our data, we need to define a set of source and target operations. Furthermore, we need to model which target operations to select given the source and target stems as well as the source operation.

7.3.4 Operations

In our experiments, we used two different types of operations. The first type of operations are simple replacement operations as described above. These operations replace the last letters of a word by other letters. The beginning of the word will not be changed. We will refer to the unchanged part of the word as the word stem. We further constrain the first type of operations in three ways to prevent the model to replace the whole word. Therefore, we set the word stem to be at least m letters long. Second, in German as well as in English only very few letters need to be changed at the end of the word in most cases. Therefore, we only allow the model to replace up to $n_s \leq n$ letters by at most $n_t \leq n$ other letters. In our initial experiments, $n = 3$ and $m = 4$ lead to reasonable rules. Therefore, we use these values for all following experiments.

To avoid many operations with the same outcome, we used a third restriction on the operations. We only use operations where the first letter of both sides of the operations is different. As a result, there is always only one operation to change e_1 into e_2. In the example, the operation (“m” $\rightarrow$ “me”) would not be allowed.

If we use Type 1 operations, there are often multiple possible target operations for one source operation. Therefore, we learn a second type of operations which are

dependent on the last k letters of the stem. For example, we could learn the operation ("" → "*e*") given that the word ends with an *"m"*. In our experiments, we restrict the model to endings up to a length of $k \leq 5$ letters.

With Type 2 operations, different operations to convert e_1 to e_2 exist. We can use the operation ("" → "*e*") as well as the same operation given that the word stem ends with an *"m"*. Using this type of operations, we are able to learn different mappings to target operations depending on the ending of the word stems. We can for example learn that, when adding an *"e"* to the source word, we have to either add an *"s"* or an *"es"* to the target word. However, this will result in more operations and therefore the statistics will be less well estimated.

7.3.5 Features

After defining the operations, we need to assign a rank to the target operations given a source operation and a target word:

$$R(op_t|op_s, e_1, f_1, f_2) \tag{7.3}$$

Then we can select the best ranked target operation and apply it to e_1 to obtain a translation for f_2. We used different features to determine the ranking.

We want to ensure that the operation op_t applied to e_1 generates a valid target language word. In most cases it will not occur in the parallel corpus or in the titles of the Wikipedia articles, otherwise we could translate f_2 in the first place. Instead, we can use the complete articles of Wikipedia on the target side. If the word is a valid word in the target language, it will probably occur in the articles. Therefore, we use a feature that indicates whether the generated word is in the target Wikipedia corpus.

Furthermore, a target operation that often coincides with a given source operation should be ranked higher than one that is rarely used together with the source operation. We therefore look at pairs of entries in the lexicon generated from our parallel training data and count for how many of them the source operation can be applied to the source side and the target operation can be applied to the target side. This count is then used as an additional feature. For the second type of operations, we used an indicator feature that the operation can be applied at least c times. We performed several experiments and $c = 3$ lead to the best score on the development data. Therefore, we use that threshold for all experiments.

Table 7.6: Features for quasi-morphological operations

Feature	Operation	
	Type 1	**Type 2**
Valid Word	x	x
Count Feature	x	
Count $> c$		x
Source Context Length		x
Target Context Length		x

We calculate additional features, when we use the second type of operations. We want to prefer operations with more context. Therefore, we use k_s, the length of ending for source operations, as well as k_t, the length of the target operations, as additional features.

For both types of operations we used a product of these features to rank the operation. For the first type, we multiplied the indicator feature for a valid word with the count feature. The second type of operations is ranked by the product of the valid word feature, the count $> c$ feature and both context length features. The features and the corresponding operation types are summarized in Table 7.6.

7.3.6 Training

In order to train a model for the morphological operations, we need to find a set of candidate operations and their associated feature values to be able to rank them. For this we use a word aligned corpus. We used the parallel training corpus together with the automatically generated word alignment. From this corpus, we extracted all source and target words that are aligned to each other to build a lexicon.

In a next step we checked if a source operation exists that changes one source word to another source word and a target operation that can be applied to the aligned target words. If this is the case, we extracted the pair of source and target operations. This provides us with the source and target operations as well as the co-occurrence counts of these operations.

7.3.7 Integration of the Operations

After being able to find a translation for OOV words, we need to integrate the approach into the phrase-based machine translation system.

We will only use the proposed method for OOVs and will not try to improve translations of words that the baseline system already covers. Therefore, in a first step we extract the OOVs from the source text. Since we want to make use of phrase pairs instead of using word-by-word translation for the OOVs, we try to find phrase pairs in the original phrase table that contain a word similar to the OOV word. This is done by looking for phrase pairs, for which a source operation op_s exists that changes one of the source words f_1 into the OOV word f_2. Since we need to apply a target operation to one word on the target side of the phrase pair, we only consider phrase pairs, where f_1 is aligned to one of the target words of the phrase pair.

If at least one target operation exists given f_1 and op_s, we select the one with the highest rank. Then we generate a new phrase pair by applying op_s to f_1 and op_t to e_1.

We do not change the features of the phrase pairs. They are not directly competing with the unchanged phrase pairs, since there is no phrase pair that exactly matches f_2.

7.3.8 Results

We analyzed the impact of the quasi-morphological operations on improving the coverage of the Wikipedia corpus in a group of experiments. First, we analyzed the OOV words again. The results are summarized in Table 7.7. We use replacement operations independent of the word stem ending (Type 1) and the operations dependent on word stem endings (Type 2). For the first type of operations, we only applied the 100 most frequent operations. The translation quality did not improve with more operations.

If we first look at the EPPS data, the quasi-morphological operations change the number of the OOV words only slightly, since we have only quite few unknown words, as already mentioned before. For the TED task, the number of unknown words could be reduced by up to 20%. For the TED development data, the Type 2 approach leads to more translation units than the Type 1 operations.

In the computer science task, the difference is greater. Here the number of OOV words could be reduced by around 30% when adding the operations. For this task, the Type 2 operations again were able to translate more words than the Type 1 operations.

Table 7.7: OOV words for quasi-morphological operations

Data	System	OOV		mod. OOV	
		total	unique	total	unique
EPPS Dev	No MorphOp	135	125	126	117
	Type 1	128	119	119	111
	Type 2	120	112	113	105
TED Dev	No MorphOp	230	192	169	173
	Type 1	202	168	168	149
	Type 2	182	150	155	137
TED Test	No MorphOp	276	225	247	119
	Type 1	239	192	211	167
	Type 2	239	192	211	167
CS Dev	No MorphOp	601	315	542	279
	Type 1	463	263	409	229
	Type 2	415	242	375	213
CS Test	No Morph Op	1007	501	825	429
	Type 1	784	412	615	350
	Type 2	718	378	563	326

Table 7.8: Overview of results using quasi-morphological operations

Task	Condition	No MorphOp		Type 1		Type 2	
		Dev	Test	Dev	Test	Dev	Test
TED	Baseline	27.52	21.98	27.51	21.98	27.51	21.98
	+ TED Dev	22.89	22.41	22.91	22.40	22.95	22.40
	+ TED Training	26.65	26.18	26.66	26.18	26.68	26.18
CS	Baseline	27.52	21.38	27.51	21.53	27.51	21.55
	+ TED Dev	22.89	22.93	22.91	23.09	22.95	23.15
	+ TED Training	26.65	25.33	26.66	25.39	26.68	25.43
	CS DEV	25.60	23.66	25.74	23.83	25.84	23.88
	+ TED Training	27.20	26.51	27.35	26.60	27.39	26.62

We display the influence on the translation quality in Table 7.8. We can observe no improvements for the EPPS data. On the TED sets there are also nearly no improvements. This is in line with the results from the previous experiments. We already saw there that for this task it is more important to have high precision translations than having more. The translations from the quasi-morphological operations are not as high precision as the ones directly learned from Wikipedia.

However, for the computer science translation task, the picture is different. Here we can improve the translation quality further by using the operations. The Type 2 operations could slightly outperform the Type 1 operations. Additional improvements of 0.1 to 0.2 BLEU points can be achieved by using these operations over the system using no morphological operations.

7.4 Summary

As shown in the last experiments, the translation quality on the computer science translation task could be improved significantly by integrating Wikipedia. The best performance was achieved by using the corpus method and by performing quasi-morphological operations in addition. In this case, the number of OOV words could be reduced by more than 50% on the development and test set. Furthermore, the translation quality could be improved by up to 0.7 BLEU points on the test set and 1.2 BLEU points on the development set.

Small improvements could also be achieved for the TED translation task. The improvements are not as big as for the other test set since the vocabulary used in the TED talks is not as specific as the one used in university lectures and therefore, the OOV words are not as problematic as for the first task.

In Example 7.1, two example sentence are shown where the translation could be improved by the information from Wikipedia. In the first example, several topic-specific words like *Kegelschnitt* and *Rotations-Ellipsoid* could be translated using the knowledge from Wikipedia. In the second example, this is only possible due to the quasi-morphological information. The word *Perzeptronen* does not occur in the Wikipedia titles, only the singular form *Perzeptron*. Using the quasi-morphological operations, we are able to generate the English plural form *perceptrons* from the singular *perceptron*.

Source	das sind in der Regel Basis Elemente wie Würfel Kegel, **Kegelschnitt**, **Rotations-Ellipsoid** und anderer **Torus**...
Reference	these are usually base elements such as cubes, cones, the **cone cut**, **rotation ellipsoid** and the other **torus**, ...
Baseline	these are generally base elements such as cube cone, **Kegelschnitt**, **rotation-Ellipsoid** and other **Torus** ...
Wikipedia	these are usually base elements such as cube cone, **conic section**, **rotation ellipsoid** and other **torus** ...
Source	nun, **Perzeptronen**, eine tolle Sache, da hat man sehr viel drüber geschrieben und viel publiziert in diesen Jahren.
Reference	now, the **perceptrons**, an awesome thing, much was written about this and much was published during these years.
Baseline	now, **Perzeptronen**, a great thing, because it was very much over written and much published in these years.
Wikipedia	now, **perceptrons**, a great thing, because it was very much over written and much published in these years.

Example 7.1: Examples of the CS task

8

Exploiting Genre Similarity of Training Data

In the previous chapter we focused on improving the topic modeling of a translation system. Now, we will concentrate on another aspect of the data, the genre. When we analyze the training data available for machine translation, a corpus typically contains texts about different topics, as a whole it represents one genre.

For many topics there is not enough data for training. In contrast, for the TED task, the TED corpus can be used as genre matching training data. The genre of the university lectures is not exactly the same one, since university lectures are more spontaneous presentations compared to the practiced performances of TED talks, but they are very similar in genre. Both can be attributed to more abstract genre of talks. Therefore, for this task training data from a similar genre is available. Furthermore, also for other genres, like political speeches, news, laws and movie subtitles, there are German-English bilingual corpora available. Therefore, it is quite realistic that some amount of in-genre training data is available for most genres.

In Chapter 1.3 we gave a definition for a genre of a text. While the topic mainly influences what a text is about, the genre defines how it is expressed. In the SMT system, the two most important models are the language model and the translation model. In order to better exploit the genre-matching data, it is most promising to adapt one of these two models to the target genre. Since the genre mainly influences how the content is expressed, the straight forward approach is to adapt the language model. This idea has the additional advantage that only monolingual in-genre data

and no parallel data is needed.

We will use the language model adaptation techniques presented in Chapter 6.1. As a motivation, we analyze the problems when using the in-genre data in a standard n-gram-based language model. We will see that one main issue is that the context used for predicting the probabilities is often very limited. In order to overcome this problem, we will present two approaches.

First, we combine the n-gram based approach with class-based approaches and thereby try to increase the context that is used during the application of the language model.

In a second approach, we developed a continuous space language model. One main advantage is that this language model always uses the same context size. Therefore, it does not have the same data sparseness problem as it is the case for the n-gram-based language model.

8.1 Motivation for Genre Modeling

In our experiments we have two different test sets, consisting of TED talks and university lectures. While the genre is the same or very similar depending on the granularity of the genre definition, the two test sets differ in their topic. The TED talks treat different topics, but in a quite general way, since they are directed to a general audience. In contrast, the university lectures are about computer science and held in front of computer science students. Therefore, the vocabulary used in these lectures is more specific.

In a first step, we will analyze how well the language of the particular genres is represented by different language models. Therefore, we measure the applicable context for the language models on the test sets from the genres. The results are summarized in Tables 8.1 and 8.2.

First, we calculated the language model probability of the sentences of both test sets using the baseline language model without training data from the TED domain. When calculating the probabilities we often cannot use the full 4-gram context of the language model, but have to back off to shorter context lengths. We can use longer context lengths, if the language model matches the test data better. In order to analyze how well the language model fits the test data, both tables show how many unigrams,

Table 8.1: Analysis of the different language models on the TED task

Language Model	OOV	1	2	3	4	Average
Baseline	256	3593	14448	10026	7363	2.58
	0.7%	10.1%	40.5%	28.1%	20.6%	
Random	947	8078	17136	7319	2206	2.05
	2.7%	22.6%	48%	20.5%	6.2%	
TED	506	6209	16229	8342	4400	2.28
	1.4%	17.4%	45.5%	23.4%	12.3%	

Table 8.2: Analysis of the different language models on the CS task

Language Model	OOV	1	2	3	4	Average
Baseline	664	6644	18727	14508	9872	2.52
	1.3 %	13.2 %	37.2%	28.8%	19.6%	
Random	1833	12325	22454	10499	3304	2.02
	3.6 %	24.4 %	44.5%	20.8%	6.6%	
TED	1271	11969	2200	10745	4421	2.17
	2.5 %	23.7 %	43.6%	21.3%	8.8%	

bigrams, trigram and 4-grams are used to estimate the probability of the test sentences. In the first line of each table the numbers for the baseline language model are shown. Since the distributions of the different context lengths are similar, both test sets are modeled quite similarly by the baseline language model.

Although, it is quite likely that in-genre data is available, its size is normally significantly smaller than the size of the general data. Therefore, we analyzed how the size of the training data influences the used context length in the two test sets. We randomly selected the same amount of sentences as there are in the TED training corpus from the general corpus and trained a language model on this corpus. The results are shown as *Random* in the tables. As expected, the average context length used in both test set decreases, since fewer n-grams occur in the training data. But still both test sets behave very similarly.

Afterwards, we applied a language model trained on the TED corpus to the test sets. In this case, both test sets behave quite differently. For the TED test sets, the context used for every word can be increased considerably by using the TED language model. For example, the number of 4-grams, that are used when predicting the test sets nearly doubles compared to the language model trained on randomly selected sentences.

In contrast, the context used by the TED language model on the computer science test set is only slightly larger than the one of the random language model. Consequently, the TED language model can apply a larger context on the TED test set than to the lecture test set. For example, for 35% of all words trigrams or 4-grams can by used for the TED data. For the university lectures this is only the case for 30% of the words.

One reason might be that many of the topic specific terms in the computer science lectures do not occur in the TED data. Therefore, no n-grams spanning these words can be used and we often need to fall back to shorter n-grams. If we were able to extend the context for the university lectures test set, without necessarily having to exactly match the n-grams in the TED language model, we could make better use of the genre similarity of TED and university lectures and thereby improve the translation quality.

8.2 Class-based Approaches

By analyzing the n-gram coverage of the genre-matching language model on a test set from an exactly matching genre and a similar genre in the previous section, we realized

that a language model trained on the TED corpus may not be the best way to exploit this data for use cases like the university lectures. There we already mentioned the problem of topic specific terms in the text.

A characteristic of data from one genre is that the overall structure of the text is the same or at least very similar, but different topic-dependent terms are used in the sentences. Griffiths et al. (2004) presented a model which tries to learn this structure in an unsupervised way. In this model, the words are either grouped into syntactic classes or into semantic topics. The context words from the different semantic topics can then be used interchangeably in the sentences. In our work we will ignore the different topics and treat all topic words the same, since we want to model the genre of the text only.

Modeling the sentence structure by an n-gram language model has been shown to be very efficient. But if we use a small language model trained on data matching only in genre, we often lose all the context information at the topic-specific words. Since the topic-dependent words never or only rarely occurred in the training data, they cannot be estimated well given the preceding context. Furthermore, the estimation of the following words is also affected, since the topic-specific words occur infrequently in language model n-grams.

In order to improve the modeling of the genre, it would be necessary to use an n-gram language model which is capable of skipping the topic-dependent words and keeping the context. We propose to use a standard n-gram language model, but to replace the topic-dependent words by a class. Then the language model treats all these words equally. Furthermore, since they are no longer modeled as an out-of-vocabulary word, but as a class, we do not lose the history and we can use contexts spanning over these words. Equation 8.1 describes this in formula.

$$P(w_1 \dots w_L) = \prod_{i=1}^{L+1} P(w_i|h_i) \tag{8.1}$$

$$= \prod_{i=1}^{L+1} P(c(w_i)|c(w_{i-N+1}) \dots c(w_{i-1}))$$

$$c(w) = \begin{cases} CLASS & : w \in T \\ w & : w \notin T \end{cases} \tag{8.2}$$

In this equation, L is the sentence length. w_i is the ith word in the sentence and w_{L+1} is the sentence end marker </s>. N is the context length used by the n-gram

language model. Finally, we use a set T of topic-dependent words. The definition of this set is the most important part of the model. If the set is overly small, the model will work as a normal language model and the additional language model will be unprofitable. On the other hand, if we put excessively many words in this set, we can no longer discriminate between different hypotheses and therefore, the model becomes ineffective. In the extreme case, where all words are in the set T, every n-gram has the same probability and the model is not beneficial.

Since we change the reference, by changing the set T, we cannot compare the perplexity of the language models using different sets. Instead, we can only evaluate the effect of the model on the translation quality.

In preliminary experiments we also tried to cluster the topic-dependent words into several classes by using the POS tag of the word as its class. However, this did not lead to any addition improvements, so we used only one class for all words.

8.2.1 Topic-dependent Words

As mentioned before, one of the most important factors of this model is the definition of this set T of topic-dependent words. In our experiments we tested two different approaches to define the set of words given the two corpora in our setup: the corpus consisting of TED lectures and the corpus from general sources. The first one is based on the vocabulary of the different corpora and the second one uses TF-IDF scores.

8.2.1.1 Class Definition based on OOV Words

The topic-dependent words occurs only in texts about the specific topic. Therefore, they rarely occur in a general corpus. This motivates the definition of the set T by the out-of-vocabulary words. We assume that all words that occur in both the general corpus and in the TED corpus belong to the general vocabulary and should therefore not be considered as topic-dependent words. In contrast, the words that only occur in the TED corpus can be seen as examples of topic-dependent words and will be part of the set T.

If a word is not in the TED corpus, it is not clear whether it is a topic-dependent word or a general word that does not occur that often and that is the reason why it is not part of the TED corpus. Since it is more harmful for the model if we did not include a topic-dependent word into T than modeling a general word for which we

anyway cannot estimate the context by T, we included all words that do not occur in the TED corpus in T.

This leads to the formal definition of the class T as:

$$T = V \backslash (V_{TED} \cap V_{General}) \tag{8.3}$$

Thus, T is defined as all words except the words in the intersection of the vocabulary of the TED corpus and the general corpus.

8.2.1.2 Class Definition based on TF-IDF

The TF-IDF score of a word w and a document d reflects how important a word is to a document given a set of documents like the TED corpus. The TF-IDF score is a well known measurement of the importance of a word in the area of information retrieval. A word with a high TF-IDF score occurs very often in the given document and occurs only in very few documents. This is the behavior that is typical for topic-dependent words.

Therefore, the second approach to select the topic-dependent words depends on the TF-IDF score. In a first step, we calculate the TF-IDF score $tfidf(w, d)$ for every word w and every individual TED talk d of the training data.

We cannot use this score directly for the task, since we need one score for every word in the whole TED corpus and not one score for every word and every individual talk. Therefore, we define the score of a word $tfidf(w)$ as the maximum over all documents.

$$tfidf(w) = max_{d \in D} tfidf(w, d) \tag{8.4}$$

The intuition here is that the topic-dependent word will occur frequently in one or two documents, while it will occur rarely in most others. In the topic-relevant documents the TF-IDF score will be high. Since we want to mark this word as a topic-dependent word, it should also have a high $tfidf(w)$. Therefore, we take the maximum over all documents.

We then define the set T as the n words with the highest TF-IDF scores. In our experiments we set n to 5000. In addition, we again also assign all words that are not in the vocabulary of the TED corpus to this set.

Table 8.3: Analysis of the class-based language models for the TED task

Language Model	OOV	1	2	3	4	Average
TED	506	6209	16229	8342	4400	2.27
	1.4%	17.4%	45.5%	23.4%	12.3%	
OOV	0	5727	16563	8796	4602	2.34
	0%	16.0%	46.4%	24.6%	12.9%	
TF-IDF	0	3558	13215	10288	8625	2.67
	0%	10.0%	37.0%	28.8%	24.2%	

8.2.2 Comparison to Class-based n-gram Models

Traditional class-based language models as introduced in Brown et al. (1992) are defined differently. In this case the probability of a word sequence is calculated as:

$$\begin{aligned} P(w_1 \dots w_L) &= \prod_{i=1}^{L+1} P(w_i|h_i) \\ &= \prod_{i=1}^{L+1} P(c(w_i)|c(w_{i-L+1}) \dots c(w_{i-1})) * P(w_i|c(w_i)) \end{aligned}$$

In contrast to this traditional approach, we ignore the conditional probability of the word given the class. In our scenario this is very hard to guess, since it should be different for every topic. In our model we want to model the common genre of the training data and the test data and try to ignore their difference in the topic. Therefore, we ignore this probability.

This leads to the problem, that we get the same probability when replacing one word of T by another. However,the original language model and the translation model will distinguish between different translations for the topic words.

8.2.3 Results

We tested the approach by using an additional class-based language model on both test sets, the TED lectures and the university lectures. Before evaluating the translation quality of this approach, we again look at the matching n-grams of the new language model. We used the two sets of topic-dependent words described above. The results are shown in Tables 8.3 and 8.4.

Table 8.4: Analysis of the class-based language models for the CS task

Language Model	OOV	1	2	3	4	Average
TED	1271	11969	2200	10745	4421	2.17
	2.5%	23.7 %	43.6%	21.3%	8.8%	
OOV	0	11136	22668	11824	4787	2.20
	0%	22.1 %	45.0%	23.5%	9.3%	
TF-IDF	0	7304	18789	13718	10604	2.55
	0%	14.5 %	37.3%	27.2%	21%	

Table 8.5: Class-based language models on the TED task

Condition	No Class LM		OOV		TF-IDF	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.32	24.38	27.24	25.32	27.02	24.86
+ LM Adapt	27.45	25.26	27.33	25.25	27.65	25.29
+ TM Adapt	28.48	26.18	28.32	25.98	28.32	26.12

First of all, we do no longer have OOV words, since all the words, that are OOV words in the original language model, are now assigned to the set T and therefore can be modeled directly by the class. But also the other n-gram counts change. Using the new models, we are able to use more context in the language model.

In addition, we see that the two test sets behave more similarly again. This approach was designed to especially increase the available context for the university test sets. And we see in the results that especially in this case the context that is used for estimating the probability of the words in this test set can be increased.

The context used in the model using the TF-IDF score is also substantially higher than the one used by the model based on OOV words. This is due to the fact that the set T is larger and therefore, we can fall back to a well estimated context of this class more often instead of using the contexts containing the word. Since we cannot discriminate between the words in the set T, this of course has also disadvantages.

Therefore, we will show the differences in translation quality of the different approaches. These results are summarized in Tables 8.5 and 8.6.

For the TED translation task, both models can improve the translation quality

Table 8.6: Class-based language models on the CS task

Condition	No Class LM		OOV		TF-IDF	
	Dev	Test	Dev	Test	Dev	Test
Baseline	26.32	23.60	27.24	24.27	27.02	23.50
+ LM Adapt	27.45	23.81	27.33	24.25	27.65	24.47
+ TM Adapt	28.48	24.94	28.32	24.20	28.32	24.45
CS Dev	26.24	24.38	26.74	24.77	26.48	24.86
+ LM Adapt	26.84	24.65	26.85	24.75	26.87	25.07
+ TM Adapt	27.50	26.11	27.33	25.89	27.29	26.04

significantly with respect to the baseline system. Furthermore, the model lead to similar or slightly worse performance than the standard n-gram language model trained on the TED data.

If we combine the class-based language model with the normal log-linear language model adaptation technique, no further improvements can be achieved. The same is true if we also add the phrase table adaptation. Again, the performance is similar or slightly worse. Furthermore, both definitions of the topic word set T lead to quite similar results on this task.

When examining the university lectures translation task, the results are different. In Table 8.6 the baseline system uses the TED development set, while the three lower lines of the table present the scores for a system using a university lectures development set.

First of all, the new model can improve the baseline system in most cases. For this translation task, we can even improve the quality by combining the in-genre class language model with the default language model adaptation technique. We can achieve improvements of up to 0.6 BLEU points. In most cases defining the class of topic-dependent words using the TF-IDF score performs better than using the OOV words.

When adding the translation model adaptation, we can no longer improve by adding the class-based language model in the condition without matching development data, we even perform worse. As a conclusion, we can improve the translation quality if we only have in-genre monolingual data and the test data does not match the test set perfectly. Hence, the class-based language model is a possibility to improve exploiting

data that only matches in genre but not in topic.

8.3 Restricted Boltzmann Machine-based Language Models

In the previous section, we showed that with the class-based language model it is possible to better exploit data that matches or is similar in genre. One disadvantage of this approach is that it is difficult to define the set of topic-dependent words, even though the definition of the set is very important for the performance of the system. One reason is that we can only make a hard decision. If a word w is in the set, the model does not distinguish between w and any other words in the set. Therefore, if two n-grams differ only in a way that w is replaced by another word of the set, they are treated the same in the model. Consequently, when predicting the probability of a word given a n-gram containing w the model can fall back to a similar n-gram instead of backing off to a shorter context. However, the class-based language model is not capable of differentiating between w and any other words in the set. In contrast, if the word w is not in the set, the model can differentiate between w and all other words, but it will more often back off to shorter contexts when an n-gram containing w occurs.

A different approach to language modeling became very popular in speech recognition and machine translation in recent years. In the so-called continuous space language models the probability of the text is modeled by a neural network. This approach provides the possibility to perform approximate matches of the history and therefore, we do not loose the whole history due to an out-of-vocabulary word anymore.

Currently, continuous space language models have two main disadvantages. First of all, training is very time-consuming. Therefore, it is possible to train such models only on relatively small amounts of training data. However, we can ignore this problem since we want to better exploit the genre-matching data. We want to train the model only this data, which is typically available in small amounts anyway.

The second disadvantage is that the calculation of the probability is slow. Therefore, it is only used in a re-ranking step. In order to overcome this problem, we presented in Niehues and Waibel (2012b) the first approach to use a continuous space language model directly in the decoder. In this approach, we use a Restricted Boltzmann Machine (RBM) to model the language model probability.

8.3.1 Related Work

Already in the early 1990s initial attempts at using neural networks to model language were presented. In a first approach Nakamura et al. (1990) presented a neural network to predict word categories. Later, Bengio et al. (2003) used neuronal networks for statistical language modeling. They described in detail a language model based on multi-layer perceptrons and could show that this reduces the perplexity on a test set compared to n-gram-based and class-based language models. In addition, they gave a short outlook on energy minimization networks.

Multi-layer perceptrons based language models have successfully been applied to speech recognition by Schwenk and Gauvain (2002), Schwenk (2007) and Mikolov et al. (2010). One main problem of continuous space language models is the size of the output vocabulary in large vocabulary continuous speech recognition. A first way to overcome this is to use a short list. Recently, Le et al. (2011) presented a structured output layer neural network which is able to handle large output vocabularies by using automatic word classes to group the output vocabulary.

A different approach using Restricted Boltzmann Machines was presented in Mnih and Hinton (2007). In contrast to our work, no approximation was performed and therefore, calculation of the language model probabilities was computationally more intensive. This model and the one mentioned before based on feed-forward networks were compared by Le et al. (2010).

Motivated by the improvements in speech recognition accuracy as well as in translation quality, authors tried to use neural networks also for the translation model in a statistical machine translation system. In Schwenk et al. (2007) as well as in Le et al. (2012) the authors modified the n-gram-based translation model to use neural networks to model the translation probabilities.

Restricted Boltzmann machines have already been successfully used for different tasks such as user rating of movies (Salakhutdinov et al., 2007) and images (Hinton et al., 2006).

8.3.2 Overview of Restricted Boltzmann Machines

Before we present language models based on Restricted Boltzmann Machines (RBMs), we will give a brief overview on RBMs. We will concentrate only on the points that are

important for our RBM-based language model (RBMLM). After describing the layout of an RBM, we will explain how the probability can be inferred by the model. Finally, we will briefly discuss the training methods. In this thesis, we adhere to the definition of RBMs as given in Hinton (2010).

RBMs are generative models that have been used successfully in many machine learning applications. As implied by the name, RBMs are a special case of general Boltzmann machines, where the layout is restricted to be a bipartite graph. Due to this restriction, the training of RBMs can be performed efficiently.

In order to use an RBM as a language model, two requirements must be met. On the one hand we need to be able to calculate a probability for any input given to the RBM. On the other hand, we need methods to train the RBM.

Since Boltzmann machines are energy-based models, the first problem can be solved using the energy associated with every possible configuration of the network. This energy can be used to model the probability of the configuration. The lower the energy in the model, the higher the probability of the input. We will describe the associated probability of an RBM in detail later on.

An RBM can be trained efficiently using contrastive divergence (Hinton, 2002). In this standard approach the gradient of the derivation of the log-likelihood of the training data is used to perform a gradient descent. Although the gradient is only approximated in this method, it has been shown to be very efficient in many applications.

8.3.2.1 Layout

A general Boltzmann machine can be described by a weighted graph. In this architecture, the nodes correspond to the units of the Boltzmann machine and the weighted edges represent connections between units. This graph has to be non-cyclic. This means there is no edge from a node to itself. Furthermore, the graph is undirected. Therefore, the weights are symmetric, meaning the weight between nodes i and j is the same as the weight between nodes j and i.

In an RBM, the graph also needs to be bipartite. That means, the units can be separated into two layers and there are only connections between the layers and no connections within the layers. One layer is the visible input layer, whose values are set to the current event, and the second layer consists of the hidden units. Since the graph is bipartite, there are no connections within each layer.

In most cases units are binary units, which can have two states. In the case of RBM-based language models, an n-gram will be represented by the states of the input layer. We will use "softmax" units instead of binary units for the input layer, because the softmax units can have K different states instead of only two. They can be modeled as K different binary states with the restriction that exactly one binary unit is in state 1 while all others are in state 0.

8.3.2.2 Calculating Probabilities in RBMs

As mentioned in the beginning, RBMs are energy-based models, which define an energy for every configuration of the network. Therefore, the network defines a probability for a given set of states of the input and hidden units by using this energy function. Let $v = \{v_i\}_{i \in hidden}$ be the vector of all the states of the input units and h be the vector of all the states of the hidden units. Then the probability of the input and hidden states is defined as

$$p(v, h) = \frac{1}{Z} e^{-E(v,h)} \tag{8.5}$$

using the energy function

$$E(v, h) = -\sum_{i \in visible} a_i v_i - \sum_{j \in hidden} b_j h_j - \sum_{i,j} v_i h_j w_{ij} \tag{8.6}$$

and the partition function

$$Z = \sum_{v,h} e^{-E(v,h)} \tag{8.7}$$

In these formulas a_i is the bias of the visible units, while b_j is the bias of the hidden units. w_{ij} is the weight of the connection between the visible unit i and the hidden unit j.

If we want to assign the probability to a word sequence, we only have the input vector, but not the hidden values. In oder to use the model, we need the probability of this word sequence summed over all possible hidden value. The probability of a visible vector is defined as:

$$p(v) = \frac{1}{Z} \sum_{h} e^{-E(v,h)} \tag{8.8}$$

The problem of this definition is that the number of terms which are summed up in the calculation of the free energy is exponential in the number of hidden units. A better

way to calculate this probability is to use the free energy of the visible vector $F(v)$:

$$e^{-F(v)} = \sum_h e^{-E(v,h)} \tag{8.9}$$

The free energy can by calculated as:

$$\begin{aligned} F(v) &= -\sum_i a_i v_i - \sum_j log(1 + e^{x_j}) \\ x_j &= b_j + \sum_i v_i w_{ij} \end{aligned} \tag{8.10}$$

Using this definition, we are still not able to calculate the probability $p(v)$ efficiently because of Z. However, we can calculate $e^{-F(v)}$ efficiently, which is proportional to the probability $p(v)$, since Z is constant for all input vectors.

8.3.2.3 Training

RBMs are commonly trained using Contrastive Divergence(CD). The aim of this training method is to increase the probability of the training examples. In order to do this, we need to calculate the derivation of the probability of the example v given the weights w_{ij}:

$$\frac{\delta log(p(v))}{\delta w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \tag{8.11}$$

$\langle \cdot \rangle$ denotes the expectation of the value in the angle brackets given the distribution indicated after the brackets. The first term can be calculated easily, since there are no interconnections between the hidden units.

The second term can only be calculated in exponential time. Even if we use Markov-Chain Monte Carlo methods, this can not be done efficiently. Therefore, Contrastive Divergence does not use this gradient, but tries to minimize the Kullback–Leibler divergence. In this case the derivation can be approximated by

$$\frac{\delta log(p(v))}{\delta w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_T \tag{8.12}$$

In this case $\langle \cdot \rangle_T$ is the expectation of the value within the angle backets after T iterations of Gibbs sampling. Although this leads to a very rough approximation of the gradient, it was shown in several experiments that it performs very well. It is possible to approximate the log-likelihood arbitrarily well by increasing T, but this is normally not done in practice, because it does not improve the training.

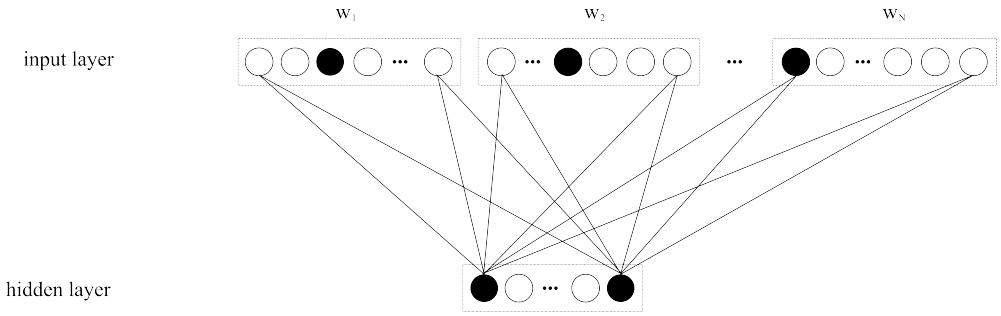

Figure 8.1: Layout of the Restricted Boltzman Machine-based language model

8.3.3 Layout of the RBM for Language Modeling

After giving a general overview of RBMs we will now describe the layout of the RBM that we use for language modeling in detail. The layout of the RBM is shown in Figure 8.1. The input layer of the n-gram language model consists of N blocks of input units, one for every word of the n-gram. Each of these blocks consists of a softmax unit, which can assume V different states representing the words of the vocabulary, where V is the vocabulary size. These softmax units are modeled by V binary units, where always exactly one unit has the value 1 and the other units have the value 0. An example can be found in Figure 8.1, where units with value 1 are shown in black. The vocabulary consists of all the words of the training text as well as sentence start and end marks (<s>, </s>) and the unknown word <unk>.

The hidden layer consists of H hidden units, where H is a free parameter, which has to be set when defining the model.

Using this setup, we need to train $N \cdot V \cdot H$ weights connecting the hidden and visible units as well as $N \cdot V + H$ bias values using the CD algorithm as described in the previous section.

8.3.3.1 Word Factors

The RBM layout described above can easily be extended to also use different word factors as used in the factored translation model (Koehn and Hoang, 2007). We can, for example, use also the part-of-speech (POS) tags of the words or we can use automatically generated word clusters. Such abstract word classes have the advantage that

they are seen more often and therefore their weights can be trained more reliably. In this case, the additional word factor can be seen as a kind of smoothing.

In order to used a factored word representation, each of the N blocks consists of W sub-blocks, where W is the number of word factors that are used. These sub-blocks are softmax units of different sizes depending on the vocabulary size of the factor. Like in the original layout, all the softmax units are then fully connected to all hidden units. The remaining layout of the framework stays the same.

8.3.4 Training of RBM-based Language Models

As it is done in most RBMs, we train our model using Contrastive Divergence. In a first step, we collect all n-grams of the training corpus and shuffle them randomly. We then split the training examples into chunks of m examples to calculate the weight updates. This is done using Equation 8.12. The first term of the equation $\langle v_j h_j \rangle_{data}$ is straightforward to calculate. The second term $\langle v_i h_j \rangle_T$ is approximated using Gibbs sampling as suggested in Hinton (2010). Therefore, first the values of the hidden values are calculated given the input. Then the values of the visible units given the hidden values are calculated. And finally, a second forward calculation is used. In our experiments we used only one iteration of Gibbs sampling with $m = 10$ as suggested by Hinton (2010).

After calculating the updates, we average over all examples and then update the weights using a learning rate of 0.1. As described in Hinton (2010), by averaging over the examples the size of the update is independent of m and therefore the learning rate does not need to be changed depending on the batch size.

Unless stated otherwise, we perform this training procedure for one iteration on the whole corpus.

8.3.5 Sentence Probability

Using the network described before we are able to calculate $e^{-F(v)}$ efficiently, which is proportional to the probability of the n-gram $P(w_1 \dots w_N)$.

If we want to use the language model as part of a translation system, we are not interested in the probability of an n-gram, but the probability of a sentence $S = <\text{s}> w_1 \dots w_L <\!/\text{s}>$. In an n-gram-based language model this is done by defining the probability of the sentence as a product of the word probabilities given the word's

history $P(S) = \prod_{i=1}^{L+1} P(w_i|h_i)$, where $w_i =$ <s> for $i \leq 0$ and $w_i =$ </s> for $i > L$. In an n-gram-based approach, $P(w_i|h_i)$ is approximated by $P(w_i|w_{i-N+1} \dots w_{i-1})$.

In our approach we are able to calculate a score proportional to $P(w_1 \dots w_N)$ efficiently, but for the conditional probability we would need to sum over the whole vocabulary as shown in Equation 8.13, which would no longer be efficient.

$$P(w_i|w_{i-N+1} \dots w_{i-1}) = \frac{P(w_{i-N+1} \dots w_i)}{\sum_{w' \in V} P(w_{i-N+1} \dots w_{i-1} w')} \tag{8.13}$$

One technique often used for n-gram-based language models is to interpolate the probabilities of different history lengths. If we use the geometric mean of all n-gram probabilities up to the length N in our model, we arrive at the following definition for the conditional probability:

$$\begin{aligned} P_{GM}(w_i|h_i) &= \frac{1}{Z_{h_i}} \sqrt[N]{\mu_{GM}(w_i|h_i)} \\ \mu_{GM}(w_i|h_i) &= \prod_{j=1}^{N} P(w_i|w_{i-j+1} \dots w_{i-1}) \\ Z_{h_i} &= \sum_{w'} \sqrt[N]{\mu_{GM}(w'|h_i)} \end{aligned} \tag{8.14}$$

For $N = 1$, we use the uni-gram probabilities $P(w_i)$. To simplify the notation, we also write it as $P(w_i|w_{i-j+1} \dots w_{i-1})$. In this case, $j = 1$ and our history would be $w_i \dots w_{i-1}$. Since $i > i - 1$, we define it as the empty string. Using this definition we

can express the sentence probability $P_{RBM}(S)$ of our RBM-based language model as:

$$
\begin{aligned}
P_{RBM}(S) &= \prod_{i=1}^{L+1} P_{GM}(w_i|h_i) \qquad (8.15)\\
&= \prod_{i=1}^{L+1} \left(\frac{1}{Z_{h_i}} \cdot \sqrt[N]{\mu_{GM}((w_i|h_i)} \right)\\
&= \left(\prod_{i=1}^{L+1} \frac{1}{Z_{h_i}} \right) \cdot \sqrt[N]{\prod_{i=1}^{L+1} \prod_{j=1}^{N} P(w_i|w_{i-j+1} \dots w_{i-1})}\\
&= \left(\prod_{i=1}^{L+1} \frac{1}{Z_{h_i}} \right) \cdot \sqrt[N]{\mu_{RBM}(S)}\\
\mu_{RBM}(S) &= \prod_{i=1}^{L+1} \prod_{j=1}^{N} P(w_i|w_{i-j+1} \dots w_{i-1})\\
&\overset{(\dagger)}{=} \prod_{i=1}^{L} P(w_{i-N+1} \dots w_i)\\
&\quad \cdot \prod_{j=2}^{N} \frac{P(w_{L-j+2} \dots w_L\text{</s>})}{P(\text{<s>})} \cdot P(\text{</s>})\\
&= \frac{1}{Z_S} \prod_{j=1}^{L+N-1} \frac{1}{Z_M} e^{-F(w_{j-N+1} \dots w_j)}
\end{aligned}
$$

In (†) we used the fact that $P(w_i|w_{i-j+1} \dots w_{i-1}) = {}^{P(w_{i-j+1} \dots w_{i-1} w_i)}/_{P(w_{i-j+1} \dots w_{i-1})}$. Then, except for the beginning and the end, all n-gram probabilities for $n < N$ cancel out. Z_M is the partition function of the RBM and $Z_S = {}^{P(\text{</s>})}/_{P(\text{<s>})^{N-1}}$.

To use the probability in the log-linear model we get:

$$
\begin{aligned}
\log\left(P_{RBM}(S)\right) = \quad & log\left(\left(\prod_{i=1}^{L+1} \frac{1}{Z_{h_i}} \right) \sqrt[N]{\frac{1}{Z_S} \prod_{j=1}^{L+N-1} \frac{1}{Z_M} e^{-F(w_{j-N+1}} \right) \qquad (8.16)\\
= \; & - \frac{1}{N}(\log(Z_S) + (N-1) \cdot \log(Z_M))\\
& - \frac{L}{N} \log(Z_M)\\
& - \sum_{i=1}^{L+1} \log(Z_{h_i})\\
& + \frac{1}{N} \sum_{j=1}^{L+N-1} F(w_{j-N+1} \dots w_j)
\end{aligned}
$$

Here the first term is constant for all sentences, so we do not need to consider it in the log-linear model. Furthermore, the second term only depends on the length of the sentence. This is already modeled by the word count model in most phrase-based translation systems. We cannot calculate the third term efficiently. If we ignore this term, it means that we approximate all n-gram probabilities by the unigram probabilities in this term, because in this case Z_{h_i} is 1 and therefore the $\log(Z_{h_i})$ is 0. By using this approximation, we can use the last term as a good feature to describe the language model probability in our log-linear model. As described before, it can be calculated efficiently.

The integration into the decoding process is very similar to the one used in n-gram-based language models. If we extend one translation hypothesis by a word, we have to add the additional n-gram probability to the current feature value as it is also done in the standard approach. We also have to save the context of $N-1$ words to calculate the probability. The only difference is that we add at the end of the sentence not only one n-gram ending with <s>, but all the ones containing </s>.

8.3.6 Results

As we did in the previous evaluation, we tested the RBM-based language models on the TED data and on the university lectures. The results are summarized in Tables 8.7a and 8.7b. In some preliminary experiments, we analyzed the influence of different network layouts and training iterations. These experiments will be described in detail later. In the summary, we report the results with 32 hidden units, which led to the best performance.

In the baseline system, we used only a 4-gram language model trained on the target side of all parallel data. If we look at the TED task, we can improve the translation quality on the test data by 0.8 BLEU points (RBMLM H32 1Iter), if we add a 4-gram RBM-based language model trained only on the TED data for 1 iteration using 32 hidden units. We can gain additional 0.6 BLEU points by carrying out 10 instead of only 1 iteration of Contrastive Divergence.

If we use a factored language model trained on the surface word forms and the automatic clusters generated by the MKCLS algorithm (Och, 1999) (FRBMLM H32 1Iter), we can see an improvement of 1.1 BLEU points already after the first iteration. We grouped the words into 50 word classes with the MKCLS algorithm.

Table 8.7: Overview on RBM-based language models

(a) Results on the TED Task

System	Dev	Test
Baseline	26.32	24.38
+ RBMLM H32 1Iter	27.39	25.13
+ RBMLM H32 10Iter	27.61	25.76
+ FRBMLM H32 1Iter	27.54	25.44
Baseline + TED LM	27.45	25.26
+ RBMLM H32 1Iter	27.64	25.52
+ RBMLM H32 10Iter	27.95	25.67
+ FRBMLM H32 1Iter	27.80	25.64
LM + PT Adapt	28.48	26.18
+ FRBMLM H32 1Iter	28.75	26.19

(b) Results on the CS task

System	TED Dev		CS Dev	
	Dev	Test	Dev	Test
Baseline	26.32	23.60	26.24	24.38
+ RBMLM H32 1Iter	27.39	24.79	26.57	25.04
+ RBMLM H32 10Iter	27.61	24.87	26.91	24.94
+ FRBMLM H32 1Iter	27.54	24.55	26.80	25.29
Baseline + TED LM	27.45	23.81	26.84	24.65
+ RBMLM H32 1Iter	27.64	24.77	27.00	25.01
+ RBMLM H32 10Iter	27.95	24.37	26.95	25.14
+ FRBMLM H32 1Iter	27.80	25.17	27.21	25.45
LM + PT Adapt	28.48	24.94	27.50	26.11
+ FRBMLM H32 1Iter	28.75	25.55	27.53	26.30

If we add an 4-gram-based language model trained only on the TED data (Baseline + TED LM) to the original baseline system, we can improve by 1 BLEU point over the baseline system. The factored RBM-based language model as well as the one trained for 10 iterations can outperform the TED n-gram-based language model by up to 0.4 BLEU points.

If we look at both scenarios for the university lectures test set (Table 8.7b), we can also improve the baseline translation quality with all RBMLM approaches. For this task, all three systems perform better than the system using an additional small TED n-gram based language model. In this case, training for 1 and 10 iterations perform similarly.

For the TED task, as shown in the first table, we can get further improvements by combining the n-gram-based TED language model and the RBM-based language model. Then we use three different language models in our system. As shown in the lower part of Table 8.7a, additional improvements of 0.3 to 0.4 BLEU points can be achieved compared to the system not using any RBM-based language model. Furthermore, it is no longer as important to perform 10 iterations of training. The difference between 1 and 10 training iterations is quite small. The factored version of the language model still performs slightly better than the language model trained only on words.

For the computer science task, the corresponding experiments are shown in Table 8.7b. For this task, it is a little bit different. The translation system using the RBM-based language model based only words can not be improved significantly by adding the n-gram based TED language model. But the system using the factored language model can be improved by adding the small n-gram based language model leading to the best performance on both tasks. The RBM-based language model can improve by 1.4 and 0.8 BLEU points over the system using only n-gram based language models (Basline + TED LM). We can gain more on the computer science task than on the TED task.

Finally, we added the factored RBM-based language model to a system using already language model and phrase table adaptation as described in Chapter 6. The results are shown in the last two lines of both tables. In this case we can improve on the development data for the TED task, but only very slightly on the test data. In contrast, on the university lecture data we can improve more. Here we can still gain 0.61 and 0.19 BLEU points over the fully adapted system.

Table 8.8: Number of hidden units for the TED task

System	Hidden Units	BLEU Score	
		Dev	Test
TED LM		27.09	23.80
RBMLM	8	25.65	23.16
	16	25.67	23.07
	32	26.40	23.41
	64	26.12	23.18

Table 8.9: Number of hidden units for the CS task

System	Hidden Units	TED Dev		CS Dev	
		Dev	Test	Dev	Test
TED LM		27.09	23.10	26.49	24.75
RBMLM	8	25.65	23.44	25.86	24.00
	16	25.67	23.50	26.04	24.23
	32	26.40	24.23	25.90	24.35
	64	26.12	24.38	26.38	24.38

In conclusion, we are able to improve the translation quality by using the RBM-based language model. As expected, more improvements can be achieved on the computer science task. Here we have more topic-specific words and therefore, more often loose the context in the n-gram based language model. Furthermore, the factored-based language model mostly performs better than the language model only based on the surface word forms. The additional smoothing performed by the automatic word classes improve the language modeling.

In a related project, the RBM-based language model was compared to a language model based on feed forward neural networks. The results are summarized in the appendix in Table A.1. In these experiments both language models perform quite similar. While the feed forward neural network can only be used in an additional reranking step, the RBM-based language model was directly integrated into the decoder.

Table 8.10: Number of training iterations for the TED task

System	Iterations	No Large LM		Large LM	
		Dev	Test	Dev	Test
TED LM		27.09	25.02	27.45	25.26
RBMLM	1	26.40	24.64	27.39	25.13
	5	26.72	24.63	27.40	25.24
	10	26.90	24.70	27.61	25.76
	15	26.57	24.71	27.63	25.49
	20	26.16	24.45	27.49	25.60

8.3.6.1 Network Layout

We carried out more experiments on both tasks to analyze the influence of the network layout on the translation quality. We used a smaller system only using the n-gram-based or RBM-based in-domain language models trained on the target side of the TED corpus. The results of these experiments are summarized in Tables 8.8 and 8.9. The first system uses an n-gram-based language model trained on the TED corpus. The other systems all use an RBMLM trained for one iteration on the same corpus.

Comparing the BLEU scores on the development and test data for the TED task, we see that we can improve the translation quality by increasing the number of hidden units to up to 32 hidden states. If we use less hidden states, the network is not able to store the probabilities of the n-grams properly. If we increase the number of hidden units further, the performance in translation quality decreases again. One reason for this might be that we have too many parameters to train given the size of the training data.

If we look at the configurations for the computer science lectures, the performance can improve even by using 64 hidden units. But there are only quite small improvements when going from 32 to 64 hidden units.

Therefore, we used a layout with 32 hidden units in most of the experiments.

8.3.6.2 Training Iterations

One critical point of the continuous space language model is the training time. While an n-gram-based language model can be trained very quickly on a small corpus like the TED corpus without any parallelization, the training of the continuous space language model takes a lot longer. In our case the corpus consists of 942K words and the vocabulary size is 28K. We trained the RBM-based language model using ten cores in parallel and it took eight hours to train the language model for one iteration.

Therefore, we analyzed in detail the influence of the number of iterations on the translation performance. The experiments were again performed on the smaller system using no large n-gram-based language model mentioned before (No Large LM) and the system using a large n-gram language model trained on all data mentioned in the beginning (Large LM). We performed these experiments on the TED task and summarized them in Table 8.10. In the first line we show the performance of the system using a n-gram-based language model trained only on the TED corpus for comparison.

In these experiments, we see that the performance increases up to ten iterations of the training data. Using ten instead of one iteration, we can increase the translation quality by up to 0.5 BLEU points on the development data as well as on the test data. Using the large language model in our baseline, adding the RBM-based language model helps more than adding the small n-gram-based language model. Performing more than ten iterations does not lead to further improvements. The translation quality even decreases again. The reason for this might be that we are facing over-fitting after the tenth iteration. In the smaller setup, using the RBM language model cannot help to outperform the n-gram-based language model. It seems to be important to use the RBMLM only in addition and not as only language model.

In summary, as we see in the results we already archive a good performance after one iterations. Since the training is quite slow, it is important that only very few iterations are needed. Up to ten iterations, small additional improvements can be achieved.

8.3.6.3 Combination with Topic Adaptation

In the last experiments we showed that the best results can be achieved with the RBMLM using the factored approach and 32 hidden units, while training the system for 1 iteration. Furthermore, this language model performs better than the class-based

Table 8.11: Combining RBM-based language models with topic adaptation

System	Topic Adapted		+ RBMLM		Gain	
	Dev	Test	Dev	Test	Dev	Test
TED	28.56	26.18	28.82	26.19	+0.26	+0.01
CS	28.56	25.43	28.82	26.04	+0.26	+0.61
+ CS Dev	28.73	26.62	28.58	26.78	−0.15	+0.16

language models. Therefore, we also tested this language model on the currently best system on all conditions using TED data. This system uses the topic adaptation from the last section.

The results of these experiments are summarized in Table 8.11. For all the cases, the performance on the test set is either as good as the baseline system or better. As it was already the case in the last experiment, the biggest improvements could be achieved on the university lectures. Here, especially if we do not have any perfectly matching development data, significant improvements can be achieved. In this case the best improvement of 0.6 BLEU points was achieved. So it is possible to combine the adaptation towards the topic described in the previous chapter with the adaptation performed in this chapter.

8.3.6.4 RBMLM for English-French

We also tested the RBMLM on the English to French translation task of TED lectures. We trained and tested the system on the data provided for the official IWSLT Evaluation Campaign 2012 (Cettolo et al., 2012). The system is similar to the one used on the German to English tasks. The results for this task are shown in Table 8.12.

The difference between the Baseline system and the systems using RBM-based language models is smaller than in the last experiments, since the baseline system uses already several n-gram-based language models. On the development set both the RBM-based language model as well as the factored RBM-based language model using also automatic word classes could improve the baseline system, by 0.1 BLEU points. For the test set only the factored version can improve the translation quality by 0.1 BLEU points. The reason may be that the system is already adapted very well to the TED data.

Table 8.12: RBM-based language model on the English-French task

System	Dev	Test
Baseline	28.93	31.90
RBMLM	28.99	31.76
FRBMLM	29.02	32.03

So it is possible to also improve the translation quality on a different language pair. Furthermore, even the performance of a very strong state-of-the-art system could be improve by this additional model.

8.4 Summary

In this chapter we analyzed how to best exploit data that is similar to the application in genre. If we look at the available data we have often corpora that represent one genre, but contain parts about different topics (EPPS corpus, TED corpus). In this chapter, we presented two approaches that could improve the translation quality if the genre of the test data is the same or similar to the one of the corpus or parts of the corpus. In a first approach, we trained a class-based language model on the data that matches in genre. In the second approach we presented a new continuous space language model that is fast enough to be used directly during decoding. If we train the model on the genre matching data we are able to gain even higher improvements than the class-based language model. As shown in Table 8.11 improvements of up to 0.6 BLEU points can be achieved even if the system has already been adapted to the task with the approaches presented in the previous chapters.

In addition to the automatic evaluation, we also compared the sentences manually. One example sentence from each test set can be found in Examples 8.1 and 8.2. In the first sentence, we are now able to select the correct singular article instead of the plural one, although there is an addition noun between article and the corresponding noun. With the additional model we are able to generalize over the additional noun although we might not have seen it in the context with the other noun.

In the second example, we are able to put the words *two* and *plus* in the correct word order. Here again, the context in the n-gram-based language model does not

cover the word Y and therefore, the translation model is not able to put the words in the correct word order.

Source:	... denn **diese** Liebes Geschichte, ...
Reference:	... because **this** love story, ...
Baseline:	... because **these** love story, ...
RBMLM:	... because **this** love story, ...

Example 8.1: Example sentence using the RBMLM on the TED task

Source:	das ist ein Vektor der hier **Y zwei plus Y drei** auf diesen Punkt zwei deutet, ja?
Reference:	this is a vector **Y two plus Y three** which points at this point two, right?
Baseline:	this is a vector of here **Y plus two Y three** indicates this point two, yes?
RBMLM:	this is a vector here **Y two plus Y three** indicates this point two, yes?

Example 8.2: Example sentence using the RBMLM on the CS task

9

Results Overview

In previous chapters we analyzed the different approaches developed in this thesis in detail. After showing the influence of these techniques separately, we will give an overview of the techniques and compare their influence in this chapter.

We will first review and summarize the influence on the two tasks used throughout this thesis, the translation of TED task and the translation of computer science lectures. These techniques were also used in several international evaluations. In the second part, we will analyze the influence of the techniques on the systems submitted by KIT to these evaluations and show the performance of the systems compared to other participants.

9.1 Speech Translation Task

Before discussing the results from the evaluation campaigns, we will review the scenarios we analyzed in this thesis. We analyzed two different tasks of a translation system. In the first one, we tried to translate more general talks directed to a broad audience as they occur in the TED talks. The second task of the systems is the translation of university lectures. In this case, we have a more specific audience and therefore the vocabulary used is also more specific.

We analyzed the tasks in different data scenarios. For the TED task, we used three scenarios and for the computer science task we used five. In the first scenario, data matching to the task is not available. In this case, we optimize the systems on parallel data from the EPPS corpus. Then we analyzed each task in a scenario where only a small amount of matching data form the TED corpus is available. This data can be

Table 9.1: Overview on all scenarios without matching training data

Technique	Baseline			Dev on TED			Dev on CS	
	Dev	**TED**	**CS**	**Dev**	**TED**	**CS**	**Dev**	**CS**
Baseline	27.54	21.34	20.21	24.09	21.59	21.50	25.37	22.72
BiLM	27.72	21.87	20.97	24.35	22.36	22.44	25.53	23.21
	+0.18	+0.53	+0.76	+0.26	+0.77	+0.94	+0.16	+0.49
Topic	27.72	21.98	21.55	24.48	22.40	23.15	26.80	23.88
	+0.00	+0.09	+0.58	+0.13	+0.04	+0.71	+1.27	+0.67
Overall	+0.18	+0.62	+1.32	+0.39	+0.81	+1.65	+1.43	+1.16

used to optimize the weights of the log-linear model. This is perfectly matching data for the TED task and roughly matching data for the university lectures. Finally, for the university lectures task, we also analyzed a scenario where we have some perfectly matching development data available.

In addition we investigated the scenarios where more matching data is available. In this case we used this data to train the translation and language model also. We test both tasks in scenarios, where some amount of parallel TED data is also available.

All results are summarized in Table 9.1 and 9.2. In the first table we summarized all scenarios using no matching training data.

In the first scenario we examine the case, where no matching data at all is available. In this case, the mismatching data is exploited better using the bilingual language model as described in Chapter 5 and we can adapt to the topic using additional knowledge sources as Wikipedia as described in Chapter 7. The results for both tasks are shown in the first three columns of Table 9.1.

The bilingual language model improves the performance by 0.5 and 0.8 BLEU point. Compared to other situations, this is a relatively large improvement by the bilingual language model. The additional context is more important when long matching phrases are not available. The system can be improved further by using Wikipedia as an additional knowledge source. As we expected, this mainly helps on the computer science task, since here a more specific vocabulary is used. In this case an additional improvement of 0.6 BLEU points can be achieved.

In summary, with the both techniques an improvement of 0.6 and 1.3 BLEU points

can be achieved for this scenario. Thus, we can conclude that it is possible to improve the performance significantly in a scenario where only mismatching data is available.

In the second scenario a small amount of TED data is available to optimize the system on this parallel data. In this case, we can again use the two techniques mentioned before to improve the performance of our system. The results for this scenario are shown in the same table in the middle columns with the title Dev on TED.

In this case bigger improvement is achieved by using the bilingual language model. On the two tasks, we improved the system by 0.8 and 0.9 BLEU points. As the matching or partly matching development data is able to estimate the correct influence of this model more accurately, the performance is improved further. For the topic adaptation, we have a similar improvements as in the last data scenario.

In this scenario using the TED development data translation quality can be improved by 0.8 and 1.7 BLEU points on the different tasks. This scenario is very common when building a new application. We have a considerably amount of data, but this data mismatches the domain of the application. Furthermore, we have a very small amount of well matching data as in the case of the TED task or roughly matching data as in the university lectures task.

Finally, we also analyzed what happens if we have perfectly matching development data also for the computer science lectures task. Results are shown in the right columns of Table 9.1.

In this case, we have also substantial improvements from the bilingual language model, but not as much as in the scenarios discussed before. The topic adaptation improves the performance to a similar degree.

In summary, we are able to gain 1.2 BLEU on the task of translating university lectures. If we compare it to the last scenario, it is possible to improve the performance by 0.7 BLEU points by using the perfectly matching development data instead of using one from the TED domain.

In the next scenarios we have in addition training data more similar to the test set. In these scenarios we used additional training data from the TED corpus. For the TED task, this data matches the task considerably well. In contrast, for the computer science lectures task, it only matches partly the task. In both cases, we are able to use additional methods presented in this work. First, the system can be adapted by combining models trained on different parts of the data as described in Chapter 6.

Table 9.2: Overview on all scenarios using TED training data

Technique	Dev on TED			Dev on CS	
	Dev	**TED**	**CS**	**Dev**	**CS**
Baseline	26.22	24.12	22.97	25.84	23.55
BiLM	26.32	24.24	23.60	26.24	24.38
	+0.10	+0.12	+0.63	+0.4	+0.83
Model Combination	28.48	26.18	24.94	27.50	26.11
	+2.16	+1.94	+1.34	+1.26	+1.73
Topic	28.56	26.18	25.43	28.73	26.62
	+0.08	+0.00	+0.49	+1.23	+0.51
Genre	28.82	26.19	26.04	28.58	26.78
	+0.26	+0.01	+0.61	-0.16	+0.16
Overall	+2.60	+2.07	+3.07	+2.74	+3.23

Furthermore, we present a technique using RBM-based language models to specially model the similarity of the genre of some data by not considering its topical differences (Chapter 8).

In the left columns of Table 9.2, we show the results when using the development data from the TED corpus. In this case, we can improve the translation quality using the bilingual language model. This time, however, the improvements are smaller, especially for the TED task. One possible reason is that the training data matches better the task and therefore, the context used in the standard translation model is already larger.

Even more improvements is achieved by model combination. The model combination includes different adaptation techniques like the language model combination, adaptation of the candidate selection and log-linear combination of phrase table scores. The language model combination has not been developed within the scope of this thesis. However, as described in Chapter 6, this method contributed only a part of the relatively huge improvements achieved by this approach. All model combination techniques can improve the translation quality by 1.9 BLEU points for the TED task and 1.3 BLEU points for the computer science task. Since the TED data matches better the TED task, it obviously leads to more substantial improvements for this task.

Afterwards, we also perform topic adaptation for this approach. The additional data can not improve for the TED task, but we again can improve the system by 0.5 BLEU points for the computer science task. This is slightly less improvement than the same task in the last scenario, since the baseline system in this data scenario is already trained with more data. Nevertheless a considerable improvement is still observed using this additional mismatching data.

Finally, in this scenario we can also adapt to the genre using the RBM-based language model. Using the additional language model does not change the BLEU scores of the TED task, but it can improve the translation performance additionally by 0.6 BLEU points for the university lectures task.

In summary, we achieve improvements of 2.1 and 3.1 BLEU points with all techniques in the data scenario. In this scenario the improvements are considerable larger than the improvements in the previous scenarios, since for the first time, all techniques developed through this thesis can be applied.

The data used in this scenario is, for example, available for every European language pair and we do not use any university lecture data. Therefore, similar improvements of the translation system can be achieved on any other task of translating speech directed to a specific audience, for example, as in conferences.

Finally, in the two most right columns of Table 9.2 we present the result for the scenario where we also have development data from the university domain. In this case, we again use all the techniques presented in this thesis.

The bilingual language model can again lead to significant improvements of 0.8 BLEU points. Compared to the scenario with TED development data, the improvement is larger is we have perfectly matching development data. Further improvements of 1.7 BLEU points is gained by using model combination. The improvement is nearly 0.4 BLEU points more than in the scenario where matching development data is available. So for both techniques can benefit from better estimated weights in the log-linear model due to the matching development data.

The improvements by topic adaptation are also very similar as the ones in the last scenario. The adaptation towards the genre increases the BLEU score only by 0.2 points compared to the 0.6 BLEU points when we use the TED development set. It is interesting, that although the improvements of the genre adaptation and model combination are similar in this and the last data scenario of the computer science lectures

task, in this scenario nearly all improvements are achieved only by the model combination. So it seems that the model adaptation and the genre adaptation techniques handle similar phonemes in the translation process, but the genre adaptation is more robust to differences between the development and test data.

If we look at the last scenario, it is possible to improve the performance by up to 3.2 BLEU points using the techniques developed and presented in this work. In this scenario, we only use data which does not match or only partly match the task. This type of data is more often available and easier to obtain than perfectly matching data.

In Table 4.5 in Chapter 4 we presented the result of a baseline system using a small amount of parallel data from the university lectures. This data matches perfectly the task and therefore is able to improve the translation quality significantly by 4 BLEU points reaching a BLEU score of 27.49 points. In reality, however, it is nearly impossible to obtain this type of data. Furthermore, if we use this type of data the system performs good only on this specific type of task. Therefore, for another application we need to collect new data for the next application.

If we compare the performance of the system with the lecture training data to the final system using only training data from the TED domain, the system performs almost as good as the oracle experiment without having a significant amount of parallel data. Our final system reached a BLEU score of 26.04 and 26.78 BLEU points on the task of translating university lectures, which are only 1.5 and 0.7 BLEU points worse than the system having the perfectly matching data available.

9.1.1 Examples

In Chapter 4, we examine some example sentences where the baseline machine translation system does not produce satisfying results. In Chapters 5 to 8 we already had a look at the sentences which could be improved by the approaches presented in these chapters. In Examples 9.1 and 9.2 we present again how the translation quality can be improved in all the examples by using all techniques presented in this thesis.

As it is expected in most of the examples, the translation can be improved by making better lexical choices. By adapting the translation system towards the task, it is possible to determine more accurately how to translate lexical items which have different translations in the target language depending on the domain. In the first example from the TED task, we are able to translate the German word *Schiffs* into

ship and not into *vessel* and in the first example from the computer science task, we no longer translate *zufällige* into *adventitious*, but into *random*.

Furthermore, the integration of additional knowledge sources enable us to translate highly topic-specific words. As shown in the example for the computer science lectures translation task, we are now able to translate words like *Perzeptronen* or *Rotations-Ellipsoid*.

Moreover, additional aspects of the translation can be improved. As the training data does not fit well any longer the test condition, the phrase-based machine translation system falls back to smaller units. Therefore, less context is available for selecting the correct translation. Using the techniques presented in this thesis, this can be prevented at some extent and improve the target word order as in example three of the TED task or the last example of the second task. Furthermore, in the last example of the TED task, the agreement is improved.

Source:	der Baupläne des **Schiffs**.
Reference:	on the blueprints of the **ship**.
Before:	the design of the **vessel**.
After:	the design of the **ship**.
Source:	wir wollen uns das best mögliche Ergebnis **vorstellen**.
Reference:	we want to **imagine** the best case scenario outcome
Before:	we want to **present** the best possible result.
After:	we want to **imagine** the best possible result.
Source:	es gibt Hinweise darauf, dass schon die Neandertaler **vor 60000 Jahren**
Reference:	there is evidence that Neanderthals, **60000 years ago**,
Before:	there are indications that the Neanderthals **before 60000 years**
After:	there is evidence that the neanderthal **60000 years ago**,
Source:	denn **diese** Liebes Geschichte,
Reference:	because **this** love story,
Before:	because **these** love story,
After:	because **this** love story,

Example 9.1: Examples of the TED task

Source:	wir haben **zufällige** Gewichte.
Reference:	we have **random** weights.
Before:	we have **adventitious** weights.
After:	We have **random** weights.
Source:	nun, **Perzeptronen**, eine tolle Sache, da hat man sehr viel drüber geschrieben und viel publiziert in diesen Jahren.
Reference:	now, the **perceptrons**, an awesome thing, much was written about this and much was published during these years.
Before:	now, **Perzeptronen**, a great thing, as it has a great deal over written and published in recent years.
After:	now, **perceptrons**, a great thing, because it was very much over written and much published in these years.
Source:	das sind in der Regel Basis Elemente wie Würfel, Kegel, **Kegelschnitt**, **Rotations-Ellipsoid** und anderer **Torus** ...
Reference:	these are usually base elements such as cubes, cones, the **cone cut**, **rotation ellipsoid** and the other **torus**, ...
Before:	these are generally base elements such as cube cone, **Kegelschnitt**, **rotation-Ellipsoid** and other **Torus** ...
After:	these are usually base elements as cube cone, **conic section**, **rotation ellipsoid** and other **torus** ...
Source:	das ist ein Vektor der hier **Y zwei plus Y drei** auf diesen Punkt zwei deutet, ja?
Reference:	this is a vector **Y two plus Y three** which points at this point two, right?
Before:	this is a vector here **Y plus two Y** to this point **three** two suggests, yes?
After:	This is a vector here **Y two plus Y three** indicates this point two, yes?

Example 9.2: Examples of the CS task

9.2 Evaluation Campaigns

Several of the techniques presented in this thesis have already successfully been used in competitive evaluations. In order to show the influence of the techniques, we review the contributions of these techniques to the final system submitted to the evaluations by KIT. Furthermore, we then compare these final systems to systems submitted to the evaluations by other groups.

KIT participated in several of these machine translation evaluations of previous years. We analyze the contribution to the WMT 2012 Evaluation[1], the Quaero 2012 Evaluation and the IWSLT 2012 Evaluation[2].

9.2.1 WMT 2012

This evaluation was part of the *NAACL 2012 Seventh Workshop on Statistical Machine Translation*. The task is to translate general newspaper text from and to English. KIT participated in the German to English, English to German, French to English and English to French task. A detailed descriptions of the system can be found in Niehues et al. (2012).

The tables describing the influence of all the techniques used in the system can also be found in the Appendix B.1. We summarize the contribution of the models presented in this thesis in Table 9.3.

In all language pairs we can improve the translation quality by using the bilingual language model. The improvements are between 0.15 and 0.27 BLEU points. The improvements by the model is not as substantial as for some of the task described in

[1] http://www.statmt.org/wmt12/

[2] http://hltc.cs.ust.hk/iwslt/index.php/evaluation-campaign.html

Table 9.3: Contributions to WMT 2012 results

Technique	DE-EN	EN-DE	FR-EN	EN-FR
BiLM	+0.18	+0.15	+0.27	+0.27
Model Combiation			+0.20	+0.12
Morphological Operations	+0.05			

this thesis. One reason can be that the training data matches better the task and therefore, we have already more parallel context in the phrase pairs available.

In the French-English and English-French systems we used also the phrase table adaptation techniques described in Chapter 6. In this evaluation we used the approaches described as Backoff approach. Instead of preferring better matching data, in this setup the technique was used to prefer data from a cleaner corpus. We used the EPPS and news commentary corpus as in-domain data. As out-of-domain data we used the Giga corpus, which is crawled from the web. This technique could improve the translation quality by up to 0.2 BLEU points.

Finally, for German to English we improved the translation quality by using the quasi-morphological operations described in Chapter 7 by 0.05 BLEU points.

In summary, the system performance was improved by 0.15 to 0.47 BLEU points using the techniques described in this thesis.

On a different test set, the official test set, the system was compared to systems submitted by other teams. The results are described in detail in Callison-Burch et al. (2012). The KIT system participated in the constrained task in all language pair. The main metric of this evaluation is the manual evaluation. In this measure every system which is not significantly worse than the best system counts as a winning system. In Callison-Burch et al. (2012), it is described that "KIT appeared in four tasks and was constrained winner each time". The systems were ranked third in all human evaluation task.

Also the automatic metric scores were calculated. The KIT system was ranked first once, second once and third twice in BLEU. In TED it is ranked first once and second three times.

9.2.2 Quaero 2012

In the Quaero Project[1] KIT participates in additional internal evaluations of MT systems. In this evaluation, the performance of the translation between German and French was measured. In one condition, the participants needed to translate news text from the project-syndicate webpage. In addition, in a speech translation task transcripts from broadcast news needed to be translated. The detailed contributions of the

[1] http://www.quaero.org

Table 9.4: Contributions to Quaero 2012 results

Technique	Text		Speech	
	DE-FR	**FR-DE**	**DE-FR**	**FR-DE**
BiLM	+0.18	+0.47	+0.25	+0.31
Model Combination	+0.11	+0.20		
Morphological Operations	+0.04			

different techniques to the KIT system can be found in Appendix B.2. A summary can be found in Table 9.4.

Again, the bilingual language model improved the translation quality in all conditions. For the text translation task, some parts of the parallel corpus was collected from the same web page. Therefore, we could use this corpus to perform phrase table adaptation using the Backoff approach. For the German to French direction, we also adapted the candidate selection of the translation model by using the CSUnion approach. This let to additional improvements of up to 0.2 BLEU points. In addition, we used the quasi-morphological operations for the German to French text system to translate unknown words.

The KIT system was very competitive, compared to other participants on the official test set. On the text translation task in the German to French direction the KIT system performed the best on TER and was ranked fourth on BLEU. For the opposite direction, the system performed second in both metrics. In the speech translation task, the system was ranked as best in both directions and both metrics.

9.2.3 IWSLT 2012

The IWSLT 2012 evaluation was carried out during the International Workshop on Spoken Language Translation. In this evaluation the task was to translate English TED lectures into French. KIT participated in the MT evaluation, where the human generated transcripts of the lectures were translated as well as in the SLT task. In the SLT task the output of the ASR system was used as an input for the translation. A detailed description of the KIT systems can be found in Mediani et al. (2012).

The contribution of the different models can be again found in the paper or in the Appendix B.3. For both tasks, the bilingual language model improved the performance

Table 9.5: Contributions to IWSLT 2012 results

Technique	EN-FR MT	EN-FR SLT
BiLM	+0.17	+0.17
Model Combiation	+1.15	+0.22
RBMLM	+0.08	

by 0.17 BLEU points. This improvement is similar to the one in the WMT. In this case a very good matching corpus is available, which is the TED corpus. Therefore, the model combination improved the performance substantially more. In this system the TED corpus was used as in-domain data and all other data as out-of-domain data. For both systems we used the Backoff approach. In the MT system, we also adapted the candidate selection by using the CS Union. This let to improvements of 1.15 BLEU points on the MT task and 0.22 BLEU points on the SLT task. Additional 0.08 BLEU points was gained by a continuous space language model based on RBMs.

After developing the system using this development test set, the system was compared to the system of other participants on the official test set and on the progress test set. The evaluation is described in detail in Cettolo et al. (2012). The MT system was ranked second in BLEU, first in TER on the official test set, and third and second respectively on the progress test set. On the progress test set also a human evaluation was performed. In the human evaluation the KIT system was ranked first together with three other systems.

On the SLT task, the KIT was first in all automatic metrics on both test sets. Furthermore, the KIT system was ranked first on the human evaluation on the progress test set.

10

Conclusion and Outlook

Statistical machine translation is currently the most promising approach to machine translation. The main advantage of this approach compared to other approaches like rule-based machine translation is the possibility to learn automatically from parallel data. In rule-based machine translation, instead, expert knowledge is required to build the translation system. By using comparably cheaper parallel data it is possible to build SMT systems for many different language pairs. Nowadays, SMT systems for up to 70 languages are available.

Consequently, if parallel data is available, relatively good translation results can be achieved. One drawback is that there is mostly only data for certain domains like general news or speeches in the European Parliament. For many other domains, such as university lectures, it is very difficult to acquire parallel data.

One problem of current SMT systems is that they perform comparatively well when the training and test data are fairly similar. But the performance drops significantly, if the testing and training conditions differ. For many real-world scenarios no well performing translation systems exist, due to the difficulty of data acquisition.

In order to improve on these conditions, we developed several techniques in this thesis to handle data which does not match the task. Furthermore, we proposed techniques to integrate matching or partly-matching data. We evaluated these approaches mainly on two tasks of translating German speech transcripts into English. In the first task, we translated TED talks and in the second one, university lectures were translated.

In a first step, we considered the condition, where no in-domain training data is available. We showed that the bilingual context available for making the translation

decision is substantially shorter than for the translation task with matching training data. In order to use more bilingual context in the decision process, we introduced the bilingual language model as an additional information source in our translation system. Using this model we were able to improve the translation quality especially in cases where the training and testing data condition do not match.

If in-domain data is available, we investigated three possibilities of exploiting this data. First, we ignored different aspects of similarity between the training data and the task. Then we analyzed what problems arise when we use a translation system for a new topic and how to model the genre of the data more precisely.

In the first approach, we tried to exploit small amounts of in-domain data, without distinguishing between genre and topic. We showed that significant gains can be achieved by adapting the language and translation model. For the language model we used the standard log-linear combination of different individual models. For the translation model, we proposed two new techniques to optimally use the in-domain data by combining models trained on different parts of the data. Furthermore, we analyzed different strategies of phrase table adaptation in detail and showed which aspects are the most important ones for the adaptation.

A further direction of research may be a combination of these techniques with other adaptation techniques like data selection methods (Moore and Lewis, 2010). This is especially helpful if the data source is unknown and therefore it is not possible to define manually which part of the data is the in-domain one. For other tasks, it is important to develop methods to acquire in-domain data. Often it is the case that no parallel data is available, but only other types of data such as comparable data. Then the additional problem of integrating this data optimally into the translation system has to be investigated.

In further techniques developed in this thesis, we tried to distinguish between topic and genre. Since it is difficult to obtain parallel topic matching data for many university lectures, we use the inter-language links of Wikipedia to retrieve additional translations for topic-specific terminology. Using this approach, we improved the translation quality significantly. In the document titles of Wikipedia only the basic forms are used, while a lecture can contain different word forms. In order to solve this problem, we had to generate translations for the different morphological forms of topic-specific terms.

In addition to finding translations for unknown source words, future research may be directed at learning new translations for existing words as mentioned in Carpuat et al. (2012) to improve the translation quality further. In this case, we will need to develop techniques to recognize when a source word has a new meaning.

Finally, we focused on genre matching data, an example of which is the TED corpus. Here, we concentrated on using it in the language model to better model the lecture genre. We developed a new continuous space language model, which allows to keep the context around topic-specific words. Previously, continuous space language models could only be applied in a re-scoring step. With the proposed model, a direct integration into the decoding process could be achieved, reducing the additional latency in a speech-to-speech translation system.

Using all the presented techniques enables us to build a translation system that performs significantly better on a new application, even if no or almost no new data can be collected. All resources that are used in this thesis can be easily found for many language pairs especially for European language pairs.

In all applications analyzed during this thesis, we adapted the system to one type of data. However, there are also applications where input data stems from several domains (Banerjee, 2013). It would be interesting to analyze how the techniques presented here perform in such situations. Furthermore, there are applications where user feedback may be used to adapt the system to the current tasks. The adaptation techniques presented in this thesis would need to be adjusted to accommodate additional feedback from the user.

Appendices

Appendix A

Continuous Space Language Models using Restricted Boltzmann Machines

In a related project, the RBM-based language model was compared to the continuous space language model using feed forward neural networks presented in Schwenk (2007) (FFLM). The results of these experiments are summarized in Table A.1. The RBM-based language model was directly integrated into the decoder, while the FFLM was used in hypothesis reranking. The language models were tested on the German-English TED task using three different setups. First, the system was tested in a baseline system. Afterwards, first language model adaptation and then translation model adaptation was added. As can be seen in the results, both language models perform very similar. In most cases, they could outperform the system without any kind of neural network language model. The RBM-based language model performed two times better than the language model using feed forward neural networks and once worse.

Table A.1: Comparison of language models based on RBMs and feed-forward networks

System	**No CSLM**	**FFLM**	**RBMLM**
Baseline	23.02	23.97	23.77
LM Adapt	24.06	23.78	24.18
LM+TM Adapt	24.57	24.82	24.89

Appendix B

Evaluation Campaigns

B.1 WMT 2012

Table B.1: German-English results for WMT 2012

System	Dev	Test
Baseline	23.64	21.32
+ Lattice Phrase Extraction	23.76	21.36
+ Gigaward Language Model	24.01	21.73
+ **Bilingual LM**	**24.19**	**21.91**
+ Cluster LM	24.16	22.09
+ DWL	24.19	22.19
+ Tree-based Reordering	-	22.26
+ **OOV**	-	**22.31**

Table B.2: English-German results for WMT 2012

System	Dev	Test
Baseline	17.06	15.57
+ POSLM	17.27	15.63
+ **Bilingual LM**	**17.40**	**15.78**
+ Cluster LM	17.77	16.06
+ DWL	17.75	16.28

Table B.3: English-French results for WMT 2012

System	Dev	Test
Baseline	24.96	26.67
+ GigParData	26.12	28.16
+ Big LMs	29.22	29.92
+ All Reo	29.14	30.10
+ **PT Adaptation**	**29.15**	**30.22**
+ **Bilingual LM**	**29.17**	**30.49**
+ Cluster LM	29.08	30.58

Table B.4: French-English results for WMT 2012

System	Dev	Test
Baseline	25.81	27.15
+ Indomain LM	26.17	27.91
+ **PT Adaptation**	**26.33**	**28.11**
+ Big LMs	28.90	29.82
+ **Bilingual LM**	**29.14**	**30.09**
+ Cluster LM	29.31	30.25

B.2 Quaero 2012

Table B.5: German-French results for Quaero text evaluation 2012

System	Dev	Test
Baseline	38.9	36.19
+ **PT Adaptation**	**39.02**	**36.18**
+ LM Adaptation	39.14	36.55
+ **Bilingual LM**	**38.27**	**36.73**
+ Long-range Reordering + GigaLM	40.60	37.08
+ Lattice Phrase Extraction	41.03	37.41
+ DWL	41.23	37.73
+ Cluster LM	41.35	38.19
+ **CSUnion**	**41.60**	**38.31**
+ **MorphOperation**		**38.35**

Table B.6: French-German results for Quaero text evaluation 2012

System	Dev	Test
Baseline	25.8	25.12
+ **PT Adaptation**	**25.89**	**25.32**
+ LM Adaptation	26.28	25.50
+ **Bilingual LM**	**26.54**	**25.97**
+ Long- Range + Lattice Phrase Extraction	26.58	25.93
+ Discriminative Word Lexicon	26.86	25.95
+ Cluster LM	26.82	26.13
+ POS LM	27.00	26.43

Table B.7: German-French results for Quaero speech translation evaluation 2012

System	Dev	Test
Baseline	26.39	28.19
+ **Bilingual LM**	**26.82**	**28.53**
+ Long Range+ GigaLM	27.05	28.62
+ Lattice Phrase Extraction	27.14	28.85
+ Discriminative Word Lexicon	27.35	29.29
+ Cluster LM	27.45	29.46

Table B.8: French-German results for Quaero speech translation evaluation 2012

System	Dev	Test
Baseline	28.35	27.86
+ **Bilingual LM**	**29.27**	**28.17**
+ Long-range Reordering	29.20	28.29
+ Lattice Phrase Extraction	29.28	28.45
+ Discriminative Word Lexicon	29.41	28.68
+ Cluster LM	29.23	29.08
+ POS LM + CSUnion	29.43	29.21

B.3 IWSLT 2012

Table B.9: English-French results for IWSLT 2012 MT task

System	Dev	Test
Baseline	28.28	30.58
+PT Adaptation	**28.50**	**31.73**
+Bilingual LM	**28.93**	**31.90**
+Cluster LM	29.15	32.13
+CSUnion	**29.27**	**32.21**
+DWL	29.37	32.70
+RBM LM	**29.46**	**32.78**
+Agreement Correction	-	32.84

Table B.10: English-French results for IWSLT 2013 SLT task

	Dev on Text			Dev on ASR	
System	**Dev (Text)**	**Test (Text)**	**Test (ASR)**	**Dev (ASR)**	**Test (ASR)**
Baseline	25.37	27.57	21.68	19.11	21.86
+ Adaptation	**25.64**	**28.08**	**21.90**	**19.31**	**22.04**
+ Bilingual LM	**25.07**	**28.08**	**22.07**	**19.14**	**22.28**
+ Cluster LM	25.17	28.79	22.57	19.32	22.40
+ DWL	25.06	28.84	22.79	19.34	22.23
+ Agreement Correction	-	-	22.86	-	-

References

A. Allauzen, J.M. Crego, I.D. El-Kahlout, and F. Yvon. LIMSI's Statistical Translation Systems for WMT'10. In *Proceedings of the Fifth Workshop on Statistical Machine Translation and MetricsMATR (WMT 2010)*, Uppsala, Sweden, 2010. 44

A. Axelrod, X. He, and J. Gao. Domain Adaptation via Pseudo In-domain Data Selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, Edinburgh, UK., 2011. 24

A. Axelrod, Q.J. Li, and W.D. Lewis. Applications of Data Selection via Cross-Entropy Difference for Real-World Statistical Machine Translation. In *Proceedings of the Ninth International Workshop on Spoken Language Translation (IWSLT 2012)*, Hong Kong, 2012. 24

P. Banerjee. *Domain Adaptation for Statistical Machine Translation of Corporate and User-generated Content.* PhD thesis, 2013. 21, 159

P. Banerjee, J. Du, B. Li, S. Kumar Naskar, A. Way, and J. Van Genabith. Combining Multi-domain Statistical Machine Translation Models using Automatic Classifiers. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, Denver, Colorado, USA, 2010. 25

P. Banerjee, S.K. Naskar, J. Roturier, A. Way, and J. van Genabith. Domain Adaptation in SMT of User-Generated Forum Content Guided by OOV Word Reduction: Normalization and/or Supplementary Data. In *Proceedings of the 16th Annual Meeting of the European Association for Machine Translation (EAMT 2012)*, Trento, Italy, 2012a. 24

P. Banerjee, S.K. Naskar, J. Roturier, A. Way, and J. van Genabith. Translation Quality-Based Supplementary Data Selection by Incremental Update of Translation Models. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, Bejing, China, 2012b. 24, 26

Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137–1155, 2003. 128

N. Bertoldi and M. Federico. Domain Adaptation for Statistical Machine Translation with Monolingual Resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation (WMT 2009)*, Athens, Greece, 2009. 24

D. Biber. A typology of English texts. *Linguistics*, 27(1):3–44, 1989. 6

D. Biber. *Variation across Speech and Writing*. Cambridge University Press, 1991. ISBN 978-0521425568. 6

D. Biber. Using Register-Diversified Corpora for General Language Studies. *Computational Linguistics*, 19(2):219–241, 1993. 6

J.A. Bilmes and K. Kirchhoff. Factored Language Models and Generalized Parallel Backoff. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL 2003)*, Edmonton, Canada, 2003. 45

A. Bisazza, N. Ruiz, and M. Federico. Fill-up versus Interpolation Methods for Phrase-based SMT Adaptation. In *Proceedings of the Eights International Workshop on Spoken Language Translation (IWSLT 2011)*, San Francisco, California, USA, 2011. 23, 25, 77

D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003. 6

D.M. Blei, T.L. Griffiths, and M.I. Jordan. The Nested Chinese Restaurant Process and Bayesian Nonparametric Inference of Topic Hierarchies. *Journal of the ACM*, 57 (2):7:1–7:30, 2010. 6

J. Blitzer. *Domain Adaptation of Natural Language Processing Systems*. PhD thesis, 2008. xv, 19

J. Blitzer, R. McDonald, and F. Pereira. Domain Adaptation with Structural Correspondence Learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, Sydney, Australia, 2006. 21

J. Blitzer, M. Dredze, and F. Pereira. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech Republic, 2007. 21

A. Blum and T. Mitchell. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the The Eleventh Annual Conference on Computational Learning Theory (COLT '98)*, Madisson, Wisconsin, USA, 1998. 21

O. Bojar and A. Tamchyna. Improving Translation Model by Monolingual Data. In *Proceedings of the Sixth Workshop on Statistical Machine Translation (WMT 2011)*, Edinburgh, UK., 2011. 108

P.F. Brown, P.V. Desouza, R.L. Mercer, V.J.D. Pietra, and J.C. Lai. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479, 1992. 124

I. Bulyko, S. Matsoukas, R. Schwartz, L. Nguyen, and J. Makhoul. Language Model Adaptation in Machine Translation from Speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2007)*, Honolulu, USA, 2007. 21, 22

C. Callison-Burch, P. Koehn, C. Monz, M. Post, R. Soricut, and L. Specia. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation (WMT 2012)*, Montreal, Canada, 2012. 1, 154

M. Carpuat and D. Wu. Improving Statistical Machine Translation using Word Sense Disambiguation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, Prague, Czech Republic, 2007. 25, 46

M. Carpuat, H. Daumé III, A. Fraser, C. Quirk, F. Braune, A. Clifton, A. Irvine, J. Jagarlamudi, J. Morgan, M. Razmara, A. Tamchyna, K. Henry, and R. Rudinger. *Johns Hopkins Summer Workshop*, chapter Domain Adaptation in Machine Translation: Final Report. 2012. 19, 21, 25, 159

F. Casacuberta and E. Vidal. Machine Translation with Inferred Stochastic Finite-State Transducers. *Computational Linguistics*, 30(2):205–225, 2004. 44, 45

M. Cettolo, L. Bentivogli, M. Paul, and S. Stüker. Overview of the IWSLT 2012 Evaluation Campaign. In *Proceedings of the Ninth International Workshop on Spoken Language Translation (IWSLT 2012)*, Hong Kong, 2012. 142, 156

Y.S. Chan and H.T. Ng. Word Sense Disambiguation improves Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech Republic, 2007. 46

O. Chapelle, B. Schölkopf, A. Zien, et al. *Semi-Supervised Learning*. MIT press Cambridge, 2006. ISBN 978-0262514125. 20

C. Chelba and A. Acero. Adaptation of maximum entropy capitalizer: Little data can help a lot. *Computer Speech and Language*, 20(4):382–399, 2006. 20

S.F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. , *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, 1996. 13

E. Cho, C. Fügen, T. Herrmann, K. Kilgour, M. Mediani, C. Mohr, J. Niehues, K. Rottmann, C. Saam, S. Stüker, and A. Waibel. A Real-World System for Simultaneous Translation of German Lectures. In *Proceedings of the 14th Annual Conference of the International Speech Communication Association (Interspeech 2013)*, Lyon, France, 2013. 98

J. Clark, A. Lavie, and C. Dyer. One System, Many Domains: Open-Domain Statistical Machine Translation via Feature Augmentation. In *Proceedings of the Tenth Conference of the Association for Machine Translation in the Americas (AMTA 2012)*, San Diego, California, USA, 2012. 22

J.M. Crego and F. Yvon. Factored bilingual n-gram language models for statistical machine translation. *Machine Translation*, 24(2), 2010. 44, 45

W. Dai, G.-R. Xue, Q. Yang, and Y. Yu. Transferring Naive Bayes Classifiers for Text Classification. In *Proceedings Of The National Conference On Artificial Intelligence*, volume 22. 2007a. 20

W. Dai, Q. Yang, G.-R. Xue, and Y. Yu. Boosting for Transfer Learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, Corvallis, Oregon, USA, 2007b. 21

H. Daumé III. Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, volume 1785, Prague, Czech Republic, 2007. 20, 25

H. Daumé III and J. Jagarlamudi. Domain Adaptation for Machine Translation by Mining Unseen Words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLP 2011)*, Portland, Oregon, USA, 2011. 25, 26

H. Daumé III and D. Marcu. Domain Adaptation for Statistical Classifiers. *Journal of Artificial Intellegent Research*, 26:101–126, 2006. 4, 21

G. de Melo and G. Weikum. Untangling the Cross-Lingual Link Structure of Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, Uppsala, Sweden, 2010. 99

K. Duh, G. Neubig, K. Sudoh, and H. Tsukada. Adaptation Data Selection using Neural Language Models: Experiments in Machine Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, Sofia, Bulgaria, 2013. 24, 26

M. Erdmann, K. Nakayama, T. Hara, and S. Nishio. An Approach for Extracting Bilingual Terminology from Wikipedia. In *Proceedings of the 13th International Conference on Database Systems for Advanced Applications (DASFAA 2008)*, number 4947, New Delhi, India, 2008. 99

M. Federico, S. Stüker, L. Bentivogli, M. Paul, M. Cettolo, T. Herrmann, J. Niehues, and G. Moretti. The IWSLT 2011 Evaluation Campaign on Automatic Talk Translation. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 2012)*, volume 12, Istanbul, Turkey, 2012. 1

G. Foster and R. Kuhn. Mixture-Model Adaptation for SMT. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech Republic, 2007. 23, 25

G. Foster, R. Kuhn, and H. Johnson. Phrasetable Smoothing for Statistical Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, Sydney, Australia, 2006. 13

G. Foster, C. Goutte, and R. Kuhn. Discriminative Instance Weighting for Domain Adaptation in Statistical Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, MIT, Massachusetts, USA, 2010. 23

W.A. Gale and K.W. Church. A Program for Aligning Sentences in Bilingual Corpora. *Computational Linguistics*, 19(1):75–102, 1993. 29

Q. Gao and S. Vogel. Parallel Implementations of Word Alignment Tool. In *Proceedings of the Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, Columbus, Ohio, USA, 2008. 12

K. Gimpel and N.A. Smith. Rich Source-Side Context for Statistical Machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation (WMT 2008)*, Columbus, Ohio, USA, 2008. 46

T.L. Griffiths, M. Steyvers, D.M. Blei, and J.B. Tenenbaum. Integrating Topics and Syntax. In *Advances in Neural Information Processing Systems*, volume 17, 2004. 121

S. Hasan, J. Ganitkevitch, H. Ney, and J. Andrés-Ferrer. Triplet Lexicon Models for Statistical Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, Honolulu, Hawaii, USA, 2008. 45

S. Hildebrand, M. Eck, S. Vogel, and A. Waibel. Adapatation of the Translation Model for Statistical Machine Translation based on Information Retrieval. In *Proceedings of the 10th Annual Meeting of the European Association for Machine Translation (EAMT 2005)*, Budapest, Hungary, 2005. 23

G.E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, 2002. 129

G.E. Hinton. A Practical Guide to Training Restricted Boltzmann Machines. Technical report, 2010. 129, 133

G.E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), 2006. 128

J. Jiang. *Domain Adaptation in Natural Language Processing*. PhD thesis, 2008. 19

J. Jiang and C.X. Zhai. Instance Weighting for Domain Adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, volume 2007, Prague, Czech Republic, 2007a. 20

J. Jiang and C.X. Zhai. A Two-Stage Approach to Domain Adaptation for Statistical Classifiers. In *Proceedings of the ACM Sixteenth Conference on Information and Knowledge Management (CIKM 2007)*, Lisboa, Portugal, 2007b. 20

B. Kessler, G. Numberg, and H. Schütze. Automatic Detection of Text Genre. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics (ACL '97/EACL'97)*, Madrid, Spain, 1997. 7

R. Kneser and H. Ney. Improved Backing-Off for m-gram Language Modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '95)*, volume I, Detroit, Michigan, USA, 1995. 13

P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings The Tenth Machine Translation Summit (MT Summit X)*, volume 5, Phuket, Thailand, 2005. 8, 28

P. Koehn and H. Hoang. Factored Translation Models. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, Prague, Czech Republic, 2007. 68, 69, 132

P. Koehn and K. Knight. Empirical Methods for Compound Splitting. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2003)*, Budapest, Hungary, 2003. 12

P. Koehn and J. Schroeder. Experiments in Domain Adaptation for Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation (WMT 2007)*, Prague, Czech Republic, 2007. 22, 23, 25, 76, 77

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech Republic, 2007. 12

H.S. Le, A. Allauzen, G. Wisniewski, and F. Yvon. Training Continuous Space Language Models: Some Practical Issues. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, Cambridge, MA, 2010. 128

H.S. Le, I. Oparin, A. Allauzen, J.-L. Gauvain, and F. Yvon. Structured Output Layer Neural Network Language Model. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2011)*, Prague, Czech Republic, 2011. 26, 128

H.S. Le, A. Allauzen, and F. Yvon. Continuous Space Translation Models with Neural Networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2012)*, Montreal, Canada, 2012. 128

X. Li and J. Bilmes. A Bayesian Divergence Prior for Classifier Adaptation. In *International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, San Juan, Puerto Rico, 2007. 20

Y. Lin, Y. Lee, and G. Wahba. Support Vector Machines for Classification in Nonstandard Situations. *Machine Learning*, 46(1-3):191–202, 2002. 21

K. Macherey, A. Dai, D. Talbot, A. Popat, and F. Och. Language-independent Compound Splitting with Worphological Operations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLP 2011)*, Portland, Oregon, USA, 2011. 108

S. Mansour, J. Wuebker, and H. Ney. Combining Translation and Language Model Scoring for Domain-Specific Data Filtering. In *Proceedings of the Eights International Workshop on Spoken Language Translation (IWSLT 2011)*, San Francisco, California, USA, 2011. 24

J.B. Mariño, R.E. Banchs, J.M. Crego, A. de Gispert, P. Lambert, J.A.R. Fonollosa, and M.R. Costa-jussá. N-gram-based Machine Translation. *Computational Linguistics*, 32(4):527–549, 2006. 45

S. Matsoukas, A.-V.I. Rosti, and B. Zhang. Discriminative Corpus Weight Estimation for Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, Singapore, 2009. 23

E. Matusov, R. Zens, D. Vilar, A. Mauser, M. Popovic, S. Hasan, and H. Ney. The RWTH Machine Translation System. In *TC-STAR Workshop on Speech-to-Speech Translation*, 2006. 45

A. Mauser, S. Hasan, and H. Ney. Extending Statistical Machine Translation with Discriminative and Trigger-based Lexicon Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, Singapore, 2009. 46

D. McClosky, E. Charniak, and M. Johnson. Effective Self-Training for Parsing. In *Proceedings of the Human Language Technology conference – North American chapter of the Association for Computational Linguistics annual meeting (HLT-NAACL 2006)*, New York City, New York, USA, 2006. 21

M. Mediani, Y. Zhang, T.-L. Ha, J. Niehues, E. Cho, T. Herrmann, R. Kärgel, and A. Waibel. The KIT Translation systems for IWSLT 2012. In *Proceedings of the*

Ninth International Workshop on Spoken Language Translation (IWSLT 2012), Hong Kong, 2012. 155

T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (Interspeech 2010)*, volume 2010, Makuhari, Japan, 2010. 26, 128

E. Minkov and K. Toutanova. Generating Complex Morphology for Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech Republic, 2007. 108

A. Mnih and G.E. Hinton. Three new Graphical Models for Statistical Language Modelling. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, Corvallis, Oregon, USA, 2007. 128

R.C. Moore and W. Lewis. Intelligent Selection of Language Model Training Data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, Uppsala, Sweden, 2010. 22, 24, 158

M. Nakamura, K. Maruyama, T. Kawabata, and K. Shikano. Neural network approach to word category prediction for English texts. In *Proceedings of the 13rd International Conference on Computational Linguistics (COLING '90)*, Helsinki, Finland, 1990. 128

J. Niehues and M. Kolss. A POS-based Model for Long-Range Reorderings in SMT. In *Proceedings of the Fourth Workshop on Statistical Machine Translation (WMT 2009)*, Athens, Greece, 2009. 15

J. Niehues and S. Vogel. Discriminative Word Alignment via Alignment Matrix Modeling. In *Proceedings of the Third Workshop on Statistical Machine Translation (WMT 2008)*, Columbus, Ohio, USA, 2008. 12

J. Niehues and A. Waibel. Domain Adaptation in Statistical Machine Translation using Factored Translation Models. In *Proceedings of the 14th Annual Meeting of the European Association for Machine Translation (EAMT 2010)*, St. Raphaël, France, 2010. 60, 68

J. Niehues and A. Waibel. Using Wikipedia to Translate Domain-Specific Terms in SMT. In *Proceedings of the Eights International Workshop on Spoken Language Translation (IWSLT 2011)*, San Francisco, California, USA, 2011. 96

J. Niehues and A. Waibel. Detailed Analysis of different Strategies for Phrase Table Adaptation in SMT. In *Proceedings of the Tenth Conference of the Association for Machine Translation in the Americas (AMTA 2012)*, San Diego, California, USA, 2012a. 60, 77

J. Niehues and A. Waibel. Continuous Space Language Models using Restricted Boltzmann Machines. In *Proceedings of the Ninth International Workshop on Spoken Language Translation (IWSLT 2012)*, Hong Kong, 2012b. 127

J. Niehues, M. Mediani, T. Herrmann, M. Heck, C. Herff, and A. Waibel. The KIT Translation system for IWSLT 2010. In *Proceedings of the Seventh International Workshop on Spoken Language Translation (IWSLT 2010)*, Paris, France, 2010. 60, 65

J. Niehues, T. Herrmann, S. Vogel, and A. Waibel. Wider Context by Using Bilingual Language Models in Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation (WMT 2011)*, Edinburgh, UK, 2011. 39, 44

J. Niehues, Y. Zhang, M. Mediani, T. Herrmann, E. Cho, and A. Waibel. The Karlsruhe Institute of Technology Translation Systems for the WMT 2012. In *Proceedings of the Seventh Workshop on Statistical Machine Translation (WMT 2012)*, Montreal, Canada, 2012. 153

F.J. Och. An Efficient Method for Determining Bilingual Word Classes. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL '99)*, Bergen, Norway, 1999. 136

K. Papineni, S. Roukos, T. Ward, and W.-jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, Philadelphia, Pennsylvania, 2002. xv, 18

P. Petrenz. Assessing approaches to genre classification. Master's thesis, 2009. 7

E. Prochasson and P. Fung. Rare Word Translation Extraction from Aligned Comparable Documents. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLP 2011)*, Portland, Oregon, USA, 2011. 25, 26

K. Rottmann and S. Vogel. Word Reordering in Statistical Machine Translation with a POS-Based Distortion Model. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-07)*, Skövde, Sweden, 2007. 15

R. Salakhutdinov, A. Mnih, and G.E. Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, New York City, New York, USA, 2007. 128

H. Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK, 1994. 12, 15

H. Schwenk. Continuous space language models. *Computer Speech and Language*, 21 (3):492–518, 2007. 26, 128, 163

H. Schwenk and J.-L. Gauvain. Connectionist Language Modeling For Large Vocabulary Continuous Speech Recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2002)*, Orlando, Florida, USA, 2002. 128

H. Schwenk and J. Senellart. Translation Model Adapation for an Arabic/French News Translation System by Lightly-Supervised Training. In *Proceedings The Twelfth Machine Translation Summit (MT Summit XII)*, Ottawa, Canada, 2009. 24

H. Schwenk, M. R. Costa-jussá, and J.A. R. Fonollosa. Smooth Bilingual N-Gram Translation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, Prague, Czech Republic, 2007. 128

L. Shu, B. Long, and W. Meng. A Latent Topic Model for Complete Entity Resolution. In *Proceedings of the 25th International Conference on Data Engineering (ICDE 2009)*, Shanghai, China, 2009. 6

M. Snover, B. Dorr, and R. Schwartz. Language and Translation Model Adaptation using Comparable Corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, Honolulu, USA, 2008. 25

M. Steedman, M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim. Bootstrapping Statistical Parsers from Small Datasets. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2003)*, Eacl 2003, Budapest, Hungary, 2003. 21

A. Stolcke. SRILM – An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference of Spoken Language Processing (ICSLP 2002)*, Denver, Colorado, USA, 2002. 14, 60, 61

N. Stroppa, A. van den Bosch, and A. Way. Exploiting Source Similarity for SMT using Context-Informed Features. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-07)*, Skövde, Sweden, 2007. 46

Y.-C. Tam, I. Lane, and T. Schultz. Bilingual LSA-based adaptation for statistical machine translation. *Machine Translation*, 21(4):187–207, 2007. 23

K. Toutanova, H. Suzuki, and A. Ruopp. Applying Morphology Generation Models to Machine Translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLP 2008)*, Columbus, Ohio, 2008. 108

N. Ueffing, G. Haffari, and A. Sarkar. Semi-Supervised Model Adaptation for Statistical Machine Translation. *Machine Translation*, 21(2):77–94, 2007. 24

A. Venugopal, A. Zollman, and A. Waibel. Training and Evaluation Error Minimization Rules for Statistical Machine Translation. In *Proceedings of the Workshop on Data-drive Machine Translation and Beyond (WPT-05)*, Ann Arbor, Michigan, USA, 2005. 18

S. Vogel. SMT Decoder Dissected: Word Reordering. In *Proceedings of the IEEE International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE 2003)*, Beijing, China, 2003. 18

M. Weller, A. Fraser, and S. Schulte im Walde. Using subcategorization knowledge to improve case prediction for translation to German. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, Sofia, Bulgaria, 2013. 108

H. Wu, H. Wang, and C. Zong. Domain Adaptation for Statistical Machine Translation with Domain Dictionary and Monolingual Corpora. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, Manchester, UK, 2008. 25, 26

D. Xing, W. Dai, G.-R. Xue, and Y. Yu. Bridged Refinement for Transfer Learning. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2007)(*, Warsaw, Poland, 2007. 20

K. Yu and J. Tsujii. Bilingual Dictionary Extraction from Wikipedia. In *Proceedings The Twelfth Machine Translation Summit (MT Summit XII)*, Ottawa, Canada, 2009. 99

F. Yvon. Paradigmatic cascades: a linguistically sound model of pronunciation by analogy. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics (ACL '97/EACL'97)*, Madrid, Spain, 1997. 108

X. Zhu. Semi-supervised learning literature survey. Technical report, 2006. 20