Machine Translation of Spontaneous Speech

zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften

der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Eunah Cho

aus Seoul

Tag der mündlichen Prüfung:	11. Febuar 2016
Erster Gutachter:	Prof. Dr. Alexander Waibel
Zweiter Gutachter:	Prof. Graham Neubig

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe, sowie dass ich die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung des KIT, ehem. Universität Karlsruhe (TH), zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet habe.

Karlsruhe, den 28.12.2015

Eunah Cho

Abstract

Recent advances in machine translation have yielded considerable improvements and promoted extensive development of various applications. Machine translation for speech input, or spoken language translation, is one of the rapidly improving fields. With spoken language translation techniques, users' utterance is automatically transcribed and translated into a different language. The promising nature of the task has already encouraged initial development of spoken language translation systems for certain scenarios, such as university lecture translation or telephone conversation translation systems.

When applied to spontaneous speech, however, machine translation and natural language processing techniques often show degraded performance. This is primarily due to distinctive characteristics of spontaneous speech compared to written text, which is the main training data for the techniques traditionally.

The distinctive characteristics of spontaneous speech include speech disfluencies, ungrammatical sentence structures and lack of punctuation marks, which severely degrade the performance of machine translation systems. Automatically generated speech transcripts often contain recognition errors due to such characteristics, which lead to further degraded performance of the applications. The great importance of processing spontaneous speech for machine translation systems promoted various techniques to be developed. A promising approach to avoid this issue is transforming the spontaneous speech transcripts into written-style texts.

This thesis is devoted to the machine translation of spontaneous speech. In this thesis, novel techniques that bridge the gap between spoken language

and written language are presented. Various models which modify spontaneous speech through the insertion of punctuation marks and removal of disfluencies are developed and optimized. The performance of the developed techniques is evaluated by measuring impact on machine translation output, in addition to performance on intrinsic measurements.

In order to insert reliable punctuation and segmentation into speech transcripts, an efficient machine translation-based model is developed. Speech disfluencies are modeled using conditional random fields with semantically-driven features. This model is integrated into a statistical machine translation system in order to achieve a more reliable disfluency decision process. In this thesis, it is also shown that the two issues can be successfully modeled jointly. Using a combined model of conditional random fields and neural networks, we show the improvement of machine translation for speech transcripts. Also, the initial effort on reconstructing spoken language-styled utterances into written-style ones is discussed in this thesis.

In order to model spontaneous speech in varying scenarios, two genres of data are chosen and used to test the models. Recordings of *university lectures* and *multi-party meetings* contain different degrees of spontaneousness and characteristics of spoken language. From the manual annotation and the following analysis on the two corpora, we aim to gain deeper insights on the types and frequencies of speech disfluencies.

Experiments on the two data sets using the proposed techniques show that inserting punctuation marks and segmentation as well as removing speech disfluencies can greatly improve both the quality of the machine translation and the readability of spontaneous speech. In addition, the developed techniques yielded an outstanding performance in evaluation campaigns, applied to different test data in varied scenarios.

Zusammenfassung

Jüngste Fortschritte im Bereich der maschinellen Übersetzung haben die Entwicklung unterschiedlichster Anwendungen ermöglicht. Dadurch hat die Bedeutung maschineller Übersetzung von gesprochener Sprache drastisch zugenommen.

Jedoch birgt das Gebiet der maschinellen Übersetzung und der Verarbeitung natürlicher Sprache noch viele Herausforderungen. Angewandt auf spontane Sprache werden zum Beispiel deutlich schlechtere Ergebnisse erzielt, da die Systeme hauptsächlich auf geschriebenen Texten trainiert werden.

Spontane Sprache charakterisiert sich unter anderem durch ein hohes Maß an Unflüssigkeit, ungrammatische Satzstrukturen und das Fehlen jeglicher Satzzeichen. All diese Eigenschaften bewirken auch, dass automatisch erstellte Transkripte häufig Erkennungsfehler aufweisen, welche sich dann negativ auf nachfolgende Anwendungen auswirken und somit auch die Leistung maschineller Übersetzungssysteme deutlich reduzieren. Da die Verarbeitung spontaner Sprache für die Anwendung maschineller Übersetzung somit von großer Bedeutung ist, wurden verschiedene Methoden entwickelt, die diese Eigenschaften modellieren. Ein vielversprechender Ansatz ist das Umwandeln spontansprachlicher Transkripte in formelle, der Schriftsprache ähnliche Texte.

Diese Arbeit widmet sich der maschinellen Übersetzung spontaner Sprache. Es werden neuartige Verfahren vorgestellt, die eine Brücke zwischen gesprochener und geschriebener Sprache schlagen. Es werden verschiedene Modelle entwickelt und optimiert, welche spontane Sprache durch das Einfügen von Satzzeichen und das Entfernen von Unflüssigkeiten modifizieren. Die Leistungen dieser Verfahren werden anhand von Vergleichen mit manuellen

Transkripten sowie der Auswirkungen auf die maschinellen Übersetzungen ermittelt.

Für das Einfügen von Satzzeichen wird ein effizientes Modell entwickelt, welches sich an Modellen im Bereich der maschinellen Übersetzung orientiert. Ein weiteres Modell wird für die Markierung von sprachlichen Unflüssigkeiten entwickelt, welche unter Zuhilfenahme von *"conditional random fields"* mit semantischen Merkmalen modelliert werden. Letzteres Modell ist in ein statistisches maschinelles Übersetzungssystem integriert, um Unflüssigkeiten noch zuverlässiger erkennen zu können. In dieser Arbeit wird auch gezeigt, dass beide Problemstellungen erfolgreich gemeinsam modelliert werden können. Mit einem Modell, welches *"conditional random fields"* mit neuronalen Netzwerken kombiniert, zeigen wir die Verbesserungen maschineller Übersetzungen von Transkripten. Zusätzlich werden in dieser Arbeit die ersten Bestrebungen diskutiert, gesprochene Äußerungen in eine der Schriftsprache ähnliche Form zu überführen.

Um die Modelle zu testen, werden Daten aus zwei unterschiedlichen Bereichen herangezogen, um spontane Sprache in wechselnden Situationen abbilden zu können. Aufnahmen von Universitätsvorlesungen und Besprechungen mit mehreren Teilnehmern zeigen einen unterschiedlichen Grad an Spontanität und weisen allgemein unterschiedliche Eigenschaften gesprochener Sprache auf. Durch die manuelle Annotation und darauf folgende Analyse der beiden Korpora erhoffen wir uns, ein tiefergreifendes Verständnis der unterschiedlichen Typen und Häufigkeiten sprachlicher Unflüssigkeiten zu erlangen.

Experimente mit beiden Datensätzen unter den vorgeschlagenen Vorgehensweisen zeigen, dass durch das Segmentieren und Einfügen von Satzzeichen sowie durch das Entfernen von Unflüssigkeiten, die Qualität der maschinellen Übersetzung und die Lesbarkeit spontaner Sprache drastisch verbessert werden können. Die hier entwickelten Methoden erzielten außerdem hervorragende Leistungen in Evaluierungskampagnen, wo sie auf unterschiedliche Testdaten und Szenarien angewandt wurden.

To my parents

Acknowledgements

I would like to thank my first supervisor Prof. Alex Waibel for our fruitful discussions leading to this thesis. I am grateful to him for giving me this research opportunity at the Interactive Systems Labs, where I truly enjoyed performing research with the other wonderful researchers. I would also like to thank Prof. Graham Neubig for co-advising this thesis. He provided me with many insightful views and suggestions on this research topic and this thesis.

For helping me learn about machine translation, I would like to thank Jan Niehues, who was always there for me when I had questions about it. I would like to extend my thanks to all past and present members of the machine translation team: Thanh-Le Ha, Teresa Herrmann, Mohammed Mediani, Isabel Slawik, Carsten Schnober and Yuqi Zhang.

With the members of the ASR and dialogue group I could always have inspiring joint discussions. I would like thank them: Jonas Gehring, Michael Heck, Kevin Kilgour, Narine Kokhlikyan, Florian Kraft, Christian Mohr, Markus Müller, Bao Quoc Nguyen, Huy Van Nguyen, Thai Son Nguyen, Christian Saam, Rainer Saam, Maria Schmidt, Matthias Sperber, Sebastian Stüker, Yury Titov, Joshua Winebarger, and Liang Guo Zhang. My special thanks goes to Kevin, who believed in me more than I did.

I would like to also thank my other colleagues at the Interactive Systems Labs, who always made working on projects more enjoyable: Silke Dannemaier, Sarah Fünfer, Klaus Joas, Bastian Krüger, Patricia Lichtblau, Mirjam Mäß, Virginia Roth, and Margit Rödder. I am also indebted to the former research team from Mobile Technology: Christian Fügen, Kay Rottmann, and Thilo Köhler. Working with them I learned a lot about the practical perspectives of research.

I would like to thank Bastian again, for his constant support and encouragement. Last but not least, I am thankful to my parents for their love and care, and, especially, my little sister who is always there for me 24/7, maybe partially because she is almost always on-call, but she always comforts me whenever I need someone.

Contents

List of Figures

List of Tables

LIST OF TABLES

Glossary

$. POS tag: sentence final punctuation mark

ADV POS tag: adverb

AMI Augmented Multi-party Interaction, a project concerned with the development technology to supper human interaction in meetings. Corpus described in McCowan et al. (2005)

APPR POS tag: preposition

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)

CRF Conditional Random Field, discriminative learning framework for sequence labeling

DE German (language)

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

DNN Deep Neural Network

DWL Discriminative Word Lexicon

EBMT Example-Based Machine Translation, a corpus-based approach to machine translation

EN English (language)

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

FL Disfluency class: filler

FR French (language)

GIZA(++) Implementation of the IBM models

HMM Hidden Markov Model

IBM-4 IBM Model 4

IR Disfluency class: interruptions

ITJ POS tag: interjection

IWSLT International Workshop on Spoken Language Translation

KIT Karlsruhe Institute of Technology

LM Language Model

ME Maximum Entropy

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

MKCLS Word cluster algorithm

ML Machine learning

MT Machine Translation

NC News Commentary, parallel corpus of several European languages; Disfluency class: non-copy

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

NN POS tag: noun; Neural Network

OOV Out-of-Vocabulary (word)

PBMT Phrase-Based Machine Translation, statistical machine translation approach

PIS POS tag: indefinite pronoun

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

PPER POS tag: personal pronoun

RBMLM RBM-based Language Model

RC Disfluency class: rough-copy

RNN Recurrent Neural Network

SB Sentence Boundary

SLT Spoken Language Translation

SMT Statistical Machine Translation, comprehensive term for statistical approaches to machine translation

SRILM SRI Language Model, most commonly used language modeling toolkit

SVM Support Vector Machine, supervised model for machine learning

SWBD Switchboard corpus, telephone speech described in Godfrey et al. (1992)

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

TER Translation Error Rate, evaluation metric for machine translation

Test Test (Set), data that is used to test a translation system

TM Translation Model

VVFIN POS tag: finite verb

WER Word Error Rate, evaluation metric for automatic speech recognition

WMT Workshop on Machine Translation

word2vec Framework to learn contiuous representations for words

ZH Chinese (language)

1

Introduction

> *A: "I mean you could imagine more things like uh how you do uhm how you correlate the different uh features over time. In the best of all possible- yeah we might need to get another memory- uhm some more memory for it that could be the problem."*
> *B: "Uhm maybe. I mean I can send it out to some of the orga- to some of the orga- group leaders."*
>
> - Excerpt from multi-party meeting corpus (Cho et al., 2014c)

Analogue to recent development of natural language processing, its related applications' performance has been greatly improved. Machine translation systems, for example, have been widely used to translate one natural written language into another one in the past years. Translating spontaneous utterance such as an excerpt shown above into another human language, however, still remains as a very challenging task. Not only the translation performance is affected, but also the readability of spontaneous utterance is highly affected due to the abundance of disfluencies. Lack of detailed punctuation marks is another reason of bad readability.

The most common sources for the parallel data required to train conventional machine translation systems are well-formed texts, such as news corpora or European parliament proceedings. These systems, therefore, perform relatively well on well-written input sentences. When applied to the transcripts of spontaneous speech, their performance is however drastically degraded. As well as degrading the performance of

machine translation systems, spontaneous speech is also much harder to read. Reducing the difference between spontaneous speech and well-formed text would therefore improve both the readability of the transcripts as well as the performance of the machine translation system. The readability of the aforementioned excerpt, for example, is greatly improved when its speech disfluencies are cleaned up:

> *A: "You could imagine more things, such as, how you correlate the different features over time. We might need to get some more memory for it, that could be the problem."*
> *B: "Maybe. I can send it out to some of the group leaders."*

As this utterance now has a format which resembles written text closer than its original form, it will match better with translation models trained on well-written sentences, yielding better performance.

The key motivation of the research led to this thesis is that the cleaned-up, well-punctuated speech transcripts will improve the machine translation performance as well as the human readability. While an extensive amount of previous works are devoted to address the issue of speech disfluencies (Fitzgerald, 2009; Shriberg, 1994) and imperfect punctuation marks (Ostendorf et al., 2008) in speech transcripts, there are remains to be done for the development of dedicated models focused on the improvement of the machine translation performance. Thus, this thesis aims to adress the issue based on a deeper analysis on the relationship between the characteristics of spoken language and its machine translation.

In this thesis, we emphasize the importance of modeling the characteristics of spontaneous speech for the improved MT. We build various models devoted to each characteristic of spontaneous speech and report their effectiveness in improving machine translation performance. The techniques developed in this thesis are also combined jointly and integrated into an SMT model in order to further investigate their potentials.

1.1 Characteristics of Spoken Language

Spoken language has distinctive characteristics compared to the written language. Written language generally consists of well-formed, grammatically correct sentences. Spoken language, on the other hand, contains speech disfluencies due to its spontaneousness.

A sheer volume of early work discussed speech disfluencies and further underlying phenomena of spontaneous speech from a psycholinguistic point of view (Levelt, 1983; Shriberg and Lickley, 1993). The syntactics and detailed description of ungrammatical, spontaneous speech were also discussed in Hindle (1983). Based on their work, in this section we give a brief description on speech disfluencies that we focus on in this thesis.

Speech disfluency can be classified into different categories. They include filler words, such as *"uh"* or *"uhm"*, and discourse markers, such as *"you know"* and *"I mean"*. They can also include the exact repetitions as well as rough ones, such as in a sentence *"There is, there was a girl"*. Another reason for speech disfluency is correction. In spontaneous speech, speakers sometimes change their words and introduce a new topic. For example, in the sentence *"I would are you okay?"*, the part *"I would"* is aborted and a new topic is introduced. Especially in multi-party conversations, such as meetings, where multiple speakers are involved, the language used by the participants contains even more spontaneousness due to their active interactions and the interruptions between them. For example, a speech segment *"I don't know what, uh, how far"* is found in the meeting data, followed by another speaker's utterance *"I will check for that"*. While the first segment already includes a correction of *"what"* and a filler word *"uh"*, its last part *"how far"* is also interrupted by the followed speaker.

In addition to the speech disfluencies, spoken language varies greatly from written language in its style and form. The usage of colloquial expressions and ungrammatical sentence structures are further differences compared to written texts.

The lack of punctuation marks and sentence segmentation in the speech transcripts causes another difficulty when processing spoken language. Unfortunately, many automatic speech recognition systems insert either no or only unreliable punctuation marks into the speech transcripts that they produce.

One promising strategy to handle this issue is to transform spoken language into a format that closely resembles written text before translation. In this method, conventional large-scale machine translation systems can be used without any additional changes. This approach has the additional advantage that the readability of speech transcripts can be improved.

In this work, different techniques to transform the transcripts of spontaneous speech into well-formed texts are shown and compared to each other. Two different genres of

data, *university lecture* and *multi-party meeting*, are chosen to represent different level of spontaneousness of a spoken language.

Lectures are a source of spoken language, which often exhibits many characteristics of spontaneous speech. Besides its spontaneousness and the following interest in it, the increased necessity of translating lectures for international audiences (Fügen et al., 2007) has recently promoted extensive research using university lecture data. In order to support further research on lecture translation, we have annotated speech disfluencies in the German lecture data collected at KIT (Stüker et al., 2012b).

In our globalized world, teams of different parts of the world are increasingly working together. Internal team meetings held in one language need to be translated into another language in order to make the discussions available to all involved parties. Since human translation is time-consuming and costly, machine translation can be a supportive tool to overcome this problem. For the two-party speech, there have been research efforts investigating speech phenomena in telephone calls, such as SWBD data (Godfrey et al., 1992). Multi-party meeting corpus has been also established (McCowan et al., 2005), where the speech disfluencies are annotated manually in the follow-up work (Besser and Alexandersson, 2007). However, the modeling of multi-speaker speech for improved machine translation remain yet underexplored. In order to support further research in this genre and develop useful models to capture speech phenomena, we choose multi-party meeting corpus as our second speech source and annotated speech disfluencies in it.

In order to evaluate the techniques developed in this work, not only is their accuracy measured but also their impact on the quality of the subsequent translation.

This work demonstrates that inserting proper punctuation marks and sentence segmentation is crucial for translating spontaneous speech. A novel approach to detect and remove speech disfluencies using semantic features is presented. The improvement on the translation quality when integrating the disfluency detection model into a statistical machine translation system is analyzed. In this work, segmentation and punctuation insertion are jointly modeled together with speech disfluency detection, combining two techniques with different strengths. Designed to exploit the synergistic effects between the techniques, this approach results in improvements both in translation quality and readability of spoken language.

1.2 Contribution of this Work

In this work, processing of spontaneous speech before machine translation is tackled from two perspectives. As many automatic speech recognition system generate either no or only unreliable punctuation marks, the impact of punctuation marks and sentence boundaries in speech transcripts is analyzed. Speech disfluencies and their negative impact on the quality of a subsequent machine translation are also investigated, along with new techniques to model them.

First, the importance of punctuation marks and segmentation on machine translation quality is established. In this oracle experiment, human-generated punctuation marks are inserted into the transcripts generated by an automatic speech recognition system, based on word-edit distance. In addition, punctuation marks generated by an automatic speech recognition system are inserted into the manual transcripts, in order to evaluate how unreliable they are and how much the performance of a machine translation system can be degraded. A monolingual translation system is developed for this challenge. It is a translation system, which translates a non-punctuated and non-segmented transcript into punctuated and segmented ones. By using the machine translation system, it is possible to utilize most of the available data that already contains reliable punctuation marks and segmentation. This monolingual translation system is successfully deployed to insert punctuation marks and segmentation into the in-house test data as well as test data of official evaluation campaigns, and improves the translation quality in all instances.

In the following part of this work, the modeling of speech disfluencies using semantic features is investigated. Certain types of speech disfluencies such as repetitions with synonyms and corrections can be detected by observing repetitive or discontinuous semantics. In this work, recurrent neural networks are used to represent each word as a continuous vector. The continuous vector learned from the recurrent neural networks can encode meaningful syntactic and semantic regularities of each word. In order to use the vectors as features efficiently, words are grouped into different clusters based on the word representations. In addition to the word clusters, a translation model is utilized. By examining the potential translation of them, the semantic closeness of adjacent words or phrases can be taken into consideration. Experiments on German lecture data showed that when a conditional random field-based model with semantic features

is applied to the disfluency detection task, it improves the translation performance by 9.8% compared to translating the text with disfluencies. Also, an upper bound of this task is established by translating a clean version of test data where all disfluencies are removed. This scheme is later extended so that it can be integrated into a statistical machine translation system. A conventional approach to remove speech disfluencies for machine translation of spontaneous speech is to process them in a separate preceding step. One potential drawback of this approach is, however, that once an incorrect to remove a disfluency decision has been made, it is hard to recover from it. This can pose a severe problem for machine translation if a removed word should have been kept and conveys an important meaning. In this new scheme, the decision on whether or not a word is a disfluency is passed on to the statistical machine translation system using word lattices. The MT reordering lattices are augmented to include the disfluency probability of each token and expanded to have extra paths which skip over disfluent words. This method improved the translation performance both on manual transcripts and automatically generated transcripts.

Finally, a joint model of punctuation and disfluency for multi-party conversations is devised. In this work, two modeling techniques with different strengths are combined to exploit the synergies between the models. Conditional random fields are used successfully in sequence labeling tasks due to their ability to model first order dependencies, while neural networks are very useful at classification tasks. In this scheme, both models generate two output labels or layers, where one is devoted to detecting punctuation marks and the other one is concerned with predicting disfluencies. The predictions of the models are extracted in probabilities and used as features in a log-linear combination. The results demonstrate that the combined model not only outperforms the individual models in all metrics but also noticeably increases the readability.

The main contribution of this thesis is to develop segmentation and punctuation insertion techniques and disfluency removal methodologies for spontaneous speech, and establish their impact on machine translation performance. In order to analyze varying degrees of spontaneousness and its impact on the task, we used two spontaneous speech data sets from different scenarios.

1.3 Overview

This section gives an overview of the contents of the individual chapters of this thesis.

Chapter 1 gives an introduction to the topic of this thesis.

Chapter 2 shows the challenges of spoken language translation. The characteristics of spoken language compared to written language are given, emphasizing a special processing is required for translation of spoken language.

Chapter 3 describes the fundamental theory applied in this work. Automatic speech recognition is introduced and fundamentals of machine translation system are given. Machine learning algorithms used throughout this work, such as conditional random fields and neural networks are described. Also, the evaluation metrics we used to measure the performance of the models built in this work are introduced.

Chapter 4 presents previous work on this topic and compares the contribution of this work to other previous works.

Chapter 5 introduces the two spontaneous data sets that we used throughout this thesis along with the English ASR system we used. The machine translation systems between multiple language pairs and the data sets we use to build the systems are also described.

Chapter 6 shows the impact of segmentation and punctuation on machine translation performance from the oracle experiments. Then the monolingual translation system, which translates a non-punctuated text into punctuated text, is introduced. We translate punctuated German lecture data into English and show the impact of monolingual translation system on the translation performance. Later on, the effectiveness of the monolingual translation system is compared with other systems in international evaluation campaigns. Also, we show how the input stack of the monolingual translation system is modified so that it can be used in the real-time spoken language translation system. We show that the latency can be efficiently decreased while maintaining the similar performance.

Chapter 7 presents a conditional random fields-based disfluency detection scheme. We devise semantically driven features, in order to capture more disfluencies. Semantic features include word clusters learned from recurrent neural networks and phrase table features. Later this model is integrated into a statistical machine translation model, using word lattices. Each word lattice encodes the disfluency probability learned from the conditional random fields-based model. For potentially disfluent words we introduce a new edge over the word, providing another path to skip over the disfluent word. In this way we can achieve better translation performance of spontaneous speech.

Chapter 8 describes two different ways of punctuation prediction and disfluency removal for multi-party meeting data. In the cascaded model, we first detect speech disfluencies using the conditional random field model. Three punctuation and segmentation schemes are them applied to the meeting data and their performance is compared. In the joint model, we use conditional random fields and neural networks for the joint detection of speech disfluency and punctuation. The two models with complementary advantages are then combined log-linearly. Our experiments show that the joint model not only improves the translation performance of the meeting data, but also the readability.

Chapter 9 shows how the fluent data after disfluency removal can reconstructed in order to fit better to the machine translation systems. Deletion of words and phrases is modeled using maximum entropy models while replacement of expressions is performed by building up an artificial data.

Chapter 10 gives an overview of the results shown in this thesis.

Chapter 11 summarizes the contribution of this work and draws conclusions.

2

Challenges of Speech Translation

With increased performance in the area of ASR, a large number of applications arise, which use the output of ASR systems as input. Many of other applications, machine translation systems are also trained on well-constructed text. It is therefore critical for the systems to have a similarly clean, well-constructed input. Spoken language, however, has very distinctive characteristics compared to written language. In this chapter, we discuss the characteristics and related challenges in machine translation of spoken language.

2.1 Lack of Sentence Boundaries and Punctuation

Currently many of automatic speech recognition systems generate either no or only unreliable punctuation marks in their hypotheses. This poses a technical challenge for the subsequent applications. Machine translation systems, for example, are generally trained on text with well-defined sentence boundaries and human-generated punctuation marks. This difference will naturally cause performance drop on translation of spoken language without any punctuation marks. The quality of segmentation and punctuation will also directly affect the translation quality, since translation models are built on such well-defined sentences. As an ASR transcript without any punctuation marks is merely a stream of words, lacking punctuation marks affects user readability negatively as well.

One way to achieve better translation quality by matching the translation models is to punctuate the ASR transcript prior to the translation process. This way has an

additional advantage that we can use the conventional setup of MT system without any further change (Peitz et al., 2011).

Performance of punctuation prediction on the ASR transcript often suffers from the spontaneousness of the speech. Since a large amount of spontaneous utterance is less grammatical compared to written texts and there are fewer sentence-like units (Rao et al., 2007b), the conventional approach based on language model (LM) shows degraded performance. Moreover, the presence of disfluencies in casual and spontaneous speech increases the difficulty of this task.

Inserting punctuation and segmentation into ASR transcripts brings another challenge, due to its required amount of context information and the system latency. In the following section, we will discuss this thoroughly.

2.1.1 Context and Latency

A real-time spoken language translation (SLT) system has to, among many other challenges, deal with the problem of latency. The latency of a real-time spoken language translation system is the time between when a word is spoken and when its transcription and translation are displayed to the user (Cho et al., 2013a). If the latency is more than a few seconds then the whole translation system becomes unusable and frustrating for the user. Each component adds to the latency, due to computation time, communication time and required future context.

Communication time can be kept to a minimum by having a fast connection and low overhead between the individual components. Computation time may be reduced by running the components on fast servers with multiple cores and by parallelizing those parts of the individual components that can be. It may also require sacrificing accuracy by using smaller, faster models.

In order to reduce the apparent latency the speech recognition component can be configured to output its current best hypothesis about once a second. The displayed output is then often updated by a newer, possibly better, hypothesis. This type of setup has a much higher user acceptance than the alternative setup where the speech recognition component waits until it has a stable hypothesis before outputting it which can sometimes result in 8 or more words appearing at once.

The MT component is even more dependent on context than the speech recognition component and often has to wait for the whole sentence to be recognized before it

can be properly translated. A fast enough MT system can re-translate the sentence each time the ASR system recognizes a new word and change the output displayed to the user. For this to work, however, the MT system requires the ASR output to be segmented into proper sentences.

These design decisions for both the ASR component, the MT component and the real-time spoken language translation system as a whole pose some significant challenges for the punctuation prediction component that converts the text output stream of the ASR component into proper sentences required for the MT system. A major side affect of the ASR component constantly updating its current hypothesis is that the punctuation prediction component has to deal with possibly changing inputs. It also has to have a fast computation time because the ASR system is sending updates very frequently. As the MT component requires sentence boundary information as soon as possible in order to function properly the punctuation prediction component has to function well with only very little future context.

2.2 Disfluency in Spoken Language

Spoken language largely differs from written language. It contains self-repairs and disfluencies. It sometimes includes ungrammatical parts, incomplete sentences or phrases.

Due to context, intonation, situation, and experience, humans are nevertheless able to understand such non-fluent spoken language almost instantaneously (Lickley, 1994). For machines, however, it is much more difficult to handle spontaneous speech.

The above mentioned characteristics hinder language processing and cause a major performance drop. One reason for this is the mismatch between the well-structured training data of the machine translation system and the actual test data, showing all the signs of spontaneous speech - training data for machine translation usually does not contain any disfluencies.

In the process of analyzing the output of automatic speech recognition and machine translation of spontaneous speech, we realized that our performance occasionally suffers not only from less predictable spoken tokens, which are hard to process for the automatic speech recognition systems, but also from disfluencies and pauses that hinder correct n-gram matches. Moreover, disfluencies obstruct correct reordering and phrase-pair matches in machine translation. Incorrect grammar, repetitions and corrections

also make translation difficult. These characteristics of spoken language impede all the different automatic language processing subsystems from automatic speech recognition to machine translation and therefore have a negative effect on the understandability of the output.

2.2.1 Speech Disfluency Types

In this section, we investigate detailed types of disfluencies, such as filler words, repetitions and corrections, false starts, abortions of words or sentences, hesitations, incorrectly used or pronounced words, as well as an imperfect grammar.

Filler words are a common disfluency in spontaneous speech. Filler words or sounds are words or sounds that a speaker utters while thinking about what she is going to say next or how she is going to finish a sentence. Some people insert them constantly in their speech. Obvious filler words or sounds such as *"uh"* or *"uhm"* in English, or *"äh"*, *"ähm"* or *"hmm"* in German are relatively easy to detect. Discourse markers (e.g. *"you know"*, *"well"* in English) and editing terms (e.g. *"I mean"* in English) are considered filler words as well (Shriberg, 1994; Zufferey and Popescu-Belis, 2004). Discourse markers, however, can be occasionally more difficult to distinguish as it depends on the context whether they are considered filler words or not. Examples are *"like"*, *"well"* or *"and"* in English, or *"ja"*, *"und"* or *"nun"* in German.

Another common disfluency is **repetition**, where speakers repeat their words. This is often called as *reparanda* by Shriberg (1994) and Johnson and Charniak (2004). In their work, an edit region is largely grouped into reparandum, interregnum, and repair. A reparandum is defined to be classified either repetition, revision, or restart fragment.

Repetitions of words or phrases as well as the correction are another characteristic of spoken language. The speaker copies exactly what he/she said before or utters a rough copy, only changing a part of a word or a phrase. There are various reasons for this: stuttering, bridging a gap that occurs while thinking, or simply the correction of a word or a phrase.

Simplified examples of such repetitions from our disfluency annotated lecture data with English gloss translation are shown in Table 2.1, in which an example of a identical repetition is on the upper part, and an example of a rough repetition is on the lower part. The details of this lecture data will be given in the next section.

Source	Das sind die Vorteile, **die Sie** die Sie haben.
English gloss	These are the advantages, **that you** that you have.
Source	**Da gibt es** da gab es nur eins.
English gloss	**There is** there was only one.

Table 2.1: Repetitions in spontaneous speech

In the first excerpt, we see the case where the repair phrase is identical to the reparandum. In the second excerpt, on the other hand, what the speaker said is *revised*, so that the previously stated comment is corrected or expanded.

Another recurrent part of spontaneous speech are **false starts**, or *restart fragments*, where speakers begin a sentence but change their plan of what they want to say and continue differently. This type of disfluency occurs when words or sentences are aborted. In extreme cases of false starts, a new context is introduced, putting an abrupt end to the previously discussed idea. As demonstrated in Table 2.2, the speaker starts a new way of forming the sentence after aborting the first several utterances. While the example sentences shown in Table 2.1 contain approximately repetitive words or phrases, this example contains aborted fragments without any repetition. The example shown in this table depicts a case where the context is still kept in the following new utterances. However, in spontaneous speech we occasionally confront other cases where the previous context is abandoned and a new topic is discussed.

Source	**Das ist alles, was Sie** das haben Sie alles gelernt, und jetzt können Sie...
English gloss	**That is all, what you** you have learned all of this, and now can you...

Table 2.2: Restart fragment in spontaneous speech

There are several notable differences in notations of disfluency type between this thesis and the representative previous work (Shriberg, 1994), where the author deployed an annotation method where an edit region consists of *reparandum*, *interregnum*, and *repair*. A *reparandum* can be either repetition, revision, or restart fragment. This may be followed by interregnum, which includes filler words. Afterwards, they also annotated *repair*, which is the actual utterance that the speaker wished to convey. This

work showed deep insights in disfluency phenomena in speech, comparing and analyzing different disfluency categories. In this thesis, on the other hand, we categorized different types of *reparandum* and annotated them separately. Our disfluency type includes *interregnum* as filler words and discourse markers. The *repair* is annotated explicitely. This difference is motivated by our main goal of annotation, which is to model different disfluency types separately and remove them for better machine translation quality.

Spontaneous speech includes a much wider variety of disfluencies. For example, due to hesitations speakers often generate partial words or incomplete sentences/phrases. Partial words are then sometimes incorrectly recognized, due to the non-matching vocabulary and n-grams of ASR systems. Additional characteristics of spontaneous speech include unclear pronunciation, lots of which can also lead to recognition errors, and grammatically incorrect sentences.

The degree of speech disfluency is greatly increased when there are multiple speakers involved. It occurs not only from speakers' hesitation or stuttering, but from their interaction with the other speakers. Such disfluencies are not always viewed as a psycholinguistic phenomenon, but we consider it essential to first attempt to model them together with other psycholinguistic disfluencies as they both are affecting the performance of subsequent applications negatively, showing similar properties such as incomplete sentence patterns or ungrammatical phrases.

3

Basic Speech Translation System

In this chapter, we review the fundamental components of spoken language translation systems, covering both automatic speech recognition system and statistical machine translation system. Whilst a complete speech to speech translation is possible and deployed in many applications, in this thesis we are going to limit our discussion to speech to text translation.

Afterwards, we describe the two machine learning (ML) algorithms that we use for modeling speech phenomena throughout this thesis. In this thesis, conditional random fields and neural networks are used to detect speech disfluency as well as to augment punctuation marks.

Evaluation metrics we used in this thesis are also introduced in this chapter. The performance of the automatic models built within the scope of this thesis is measured intrinsically in F-scores or extrinsically in BLEU (Papineni et al., 2002).

3.1 Automatic Speech Recognition

Automatic speech recognition systems are designed to automatically recognize speech in an audio signal or file and extract a sequence of words. Given the input sequence of feature vectors X, an ASR system finds the most probable sequence of recognized words W as:

$$\hat{W} = \arg\max_{W} P(W|X) \tag{3.1}$$

From the Bayes' rule the Equation 3.1 can be rewritten into as followings.

$$\hat{W} = \arg\max_{W} P(W|X) \tag{3.2}$$

$$= \arg\max_{W} \frac{p(X|W)P(W)}{p(X)} \tag{3.3}$$

$$= \arg\max_{W} P(X|W)P(W) \tag{3.4}$$

Once an acoustic speech input is represented as a sequence of feature vectors X, the probability density $p(X|W)$ is estimated using the acoustic model. The conditional probability $p(X|W)$ represents the probability that the feature vectors X are observed given the word sequence $W = w_0 w_1 w_2 \dots w_{n-1} w_n$.

$P(W)$ is the prior probability of the sequence, calculated by the language model. The language model is derived from the likelihood of a sequence of words. n-gram language models is a commonly used technique. In this model, the probability of a word depends on the preceding $n-1$ words is defined as following,

$$P(W) \approx \prod_{i=0}^{N} P(\omega_i|\omega_{i-1}, \omega_{i-2}, \dots, \omega_{i-(n-1)}) \tag{3.5}$$

where N denotes the number of words in the sequence and w_0 is the start of sequence symbol. As shown in this equation, the probability of a word can be calculated given the sequence of last $n-1$ words.

Based on the language and acoustic model, decoder decides the most probable sequence of words for a given sequence of input features.

3.2 Machine Translation

Machine translation systems are designed to translate one natural language into another. Statistical machine translation systems were first suggested in 90s (Brown et al., 1990, 1993) and have shown a good performance. In this thesis, we used phrase-based statistical machine translation systems for different language pairs to translate each test sets with augmented punctuation marks and/on removed speech disfluencies.

The best translation $\hat{E}$ for a given input sentence F is defined as

$$\hat{E} = \arg\max_{E} p(E|F) \tag{3.6}$$

The first step of the training process is preprocessing. In this step, sentences in the parallel data that have a big length mismatch will be filtered out. Special symbols are normalized. Depending on the language, additional preprocessing steps are applied. For example, compound words in German are split into smaller words, in order to deal with the out of vocabulary (OOV) issue.

In a phrase-based SMT system, the phrase table stores the map between source words or phrase and their aligned target ones. The first step to build the phrase table is to extract phrase pairs from the parallel data. For this, we use the word alignment information. As long as a phrase pair is not violating the word alignment, it is extracted.

Once the phrase pairs are extracted from the parallel corpus, the quality of each phrase pair is measured based on the relative frequency of the phrase pair given the source phrase and the inverse relative frequency of it given the target phrase. In order to estimate the quality of the rarely occurring phrase pairs better, smoothing is applied on the frequencies (Foster et al., 2006). Lexical probabilities are often used to given an additional information.

The language model is built on the target side of the data as described in Section 3.1. Modified Kneser-Ney smoothing (Chen and Goodman, 1996; Kneser and Ney, 1995) is applied to the probabilities.

A commonly used method for reordering is pre-reordering of source sentences prior to the translation. In this method, the rules on how to reorder source sentences according to the word order of target sentences are learned (Rottmann and Vogel, 2007).

Different models described earlier are combined log-linearly, where the weights are optimized on the development data. As described more apparently for MT in Koehn (2009), log-linear models for n feature functions can be expressed in the following form,

$$p(x) = \frac{1}{Z} \exp \sum_{i=1}^{n} \lambda_i h_i(x) \tag{3.7}$$

where λ is the weight for each feature and Z the normalization factor. $h_i(x)$ represents each feature function for input variable x.

As shown in Och and Ney (2002), combining different translation models and language models log-linearly has shown a great improvement on the translation quality. In this scheme, the individual models are encoded as separate features and weighted using interpolation coefficients optimized on a development set. Since the log-linear model

in this scheme defines the target sentence probability for the given source sentence, the previous function in 3.7 can be rewritten as

$$p(E|F) = \frac{1}{Z} \exp\left(\sum_{i \in features} \lambda_i h(E,F)\right) \tag{3.8}$$

where E denotes the target sentence, F the source sentence and Z the normalization factor. Same as before, h denotes each feature to be combined. The log-linear model scores the hypotheses, on which the decoder performs the search for a best translation.

3.3 Conditional Random Fields

Introduced by Lafferty et al. (2001), conditional random fields are a framework dedicated to labeling sequence data. Thus, given the observed sequence, a conditional random field (CRF) models a hidden label sequence. Figure 3.1 illustrates a CRF which predicts the output sequence

$$Y = \{\ldots, y_{t-1}, y_t, y_{t+1}, \ldots\}$$

given the input sequence

$$X = \{\ldots, x_{t-1}, x_t, x_{t+1}, \ldots\}$$

as an undirected graphical model.

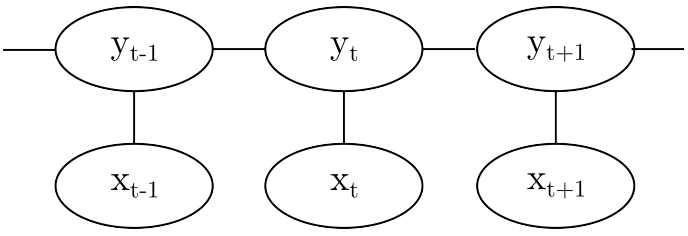

Figure 3.1: Simplified illustration of a conditional random field.

The conditional probability of a CRF model is defined as

$$p(Y|X) = \frac{1}{Z_\lambda(X)} \exp\left(\sum_i \lambda_i h_i(X,Y)\right) \tag{3.9}$$

where Z is a normalization factor devised for well-formed probability distribution. For each feature h_i its weight is λ_i.

CRFs have been applied extensively in diverse tasks of NLP, such as sentence segmentation (Liu et al., 2005), part-of-speech (POS) tagging (Lafferty et al., 2001) and shallow parsing (Sha and Pereira, 2003).

In this thesis, we use the linear chain CRF modeling technique to detect speech disfluencies as well as to augment punctuation marks. Therefore, the input sequence X in Figure 3.1 corresponds to a word sequence and the extracted features on each time step. The globally conditioned hidden variable Y is output labels, which tell us whether the word at the time step is a disfluency or not or whether a punctuation mark should be followed by the time step.

Throughout this thesis, we used two toolkits of CRFs, GRMM package (Sutton, 2006) and CRF++ (Kudoh, 2007), depending on our research purpose. GRMM package supports two-layer CRF modeling. Therefore it is suitable to model disfluency and punctuation together. To model this, we adopt one linear chain over disfluency labels, one over punctuation labels, and another one in-between edges. CRF++ is used for one-layer modeling due to its fast training and less memory usage.

3.4 Neural Networks

In last decades, neural networks (NNs) have been used extensively in various genres of tasks, such as character recognition (Hinton and Salakhutdinov, 2006). Many other NLP problems like language modeling (Schwenk, 2007), phoneme recognition (Waibel et al., 1989) and speech recognition (Hinton et al., 2012; Kilgour, 2015; Maas et al., 2014; Sainath et al., 2013) have been also successfully addressed using NN due to their good classification ability.

As its name suggests, a neural network consists of neurons, which are connected to each other. Output of each neuron is decided by all inputs to the neuron and their weights along with an activation function. Initial works in neural networks (Rosenblatt, 1957) suggested the step function as an activation function.

$$\varphi_{step}(x) = \begin{cases} 1 & \text{if} \quad x > 0 \\ 0 & \text{if} \quad x \leq 0 \end{cases} \tag{3.10}$$

However, a drawback of the step function is that its derivative is almost always 0. Instead, the sigmoid activation function defined as

$$\varphi_{sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{3.11}$$

offers a smoothed version of it. A common activation function for classification tasks, though, is the softmax activation function, which is defined as

$$\varphi_{softmax}(net_i) = \frac{e^{net_i}}{\sum_k e^{net_k}} \tag{3.12}$$

where i denotes each neuron and k is a group of neurons. As shown in this equation, the softmax activation function generalizes the output to form a discrete probability distribution.

The neurons often are build up to multiple layers, formulating multilayer neural networks. An input layer consists of input neurons with their values given. Output neurons serve as the output of the neural network. The other layers between the two layers are called hidden layers.

When neural networks have multiple hidden layers, then are referred to as deep neural networks (DNN) (Hinton et al., 2006). One of the properties of DNN is pre-training. Pre-training is designed to make use of the data to intelligently initialize the weights, instead of randomizing them prior to training with backpropagation (Dahl et al., 2012; Kilgour, 2015; Le, 2013; Seide et al., 2011). Typically pre-training is done unsupervised, using an un-annnotated data.

Two broad categories of neural networks are used for sequence modeling, recurrent neural networks (RNN) (Mikolov et al., 2010; Sundermeyer et al., 2012) where a hidden layer depends on the previous token's hidden layer and the feature of the current token to predict its label and feed forward neural networks (Bengio et al., 2006; Morin and Bengio, 2005) which have a fixed input context.

In any feed forward neural network topology, backpropagation algorithm is used to update weights for the network based on the back-propagated errors in the examples sent through the network.

Using softmax, a sigmoid neuron can saturate at its extremes. Since saturated neurons update very slowly, it is not optimal for output neurons. Because of this reason, a different error function has been tried. For error functions, mean square error and cross entropy error functions are used frequently (Bishop, 1995).

In this thesis, we use a feed-forward neural network due to its fast processing time. The neural networks are built using the *Theano* (Bergstra et al., 2010). Details on topology will be given in the relevant chapters.

3.5 Evaluation Metrics

In order to evaluate performance of models built for punctuation insertion and speech disfluency detection tasks, we used two evaluation metrics in this thesis. In this section, a brief description on them is given.

As an intrinsic evaluation metric, we used the F-score, or F-measure, which is defined as a weighted average of precision and recall. Precision, in classification tasks, is defined as

$$precision = \frac{tp}{tp + fp} \tag{3.13}$$

where tp denotes *true positive* and fp *false positive*. *true positive* represents correctly predicted condition positive, while *false positive* covers erroneous prediction on condition negative. Recall is defined as followings.

$$recall = \frac{tp}{tp + fn} \tag{3.14}$$

In this notation, fn denotes *false negative*, which represents erroneous prediction on condition positive.

F-score is then defined as following.

$$F = 2 \times \frac{precision \cdot recall}{precision + recall} \tag{3.15}$$

Therefore it represents how accurately the model performs on a given task. For punctuation insertion task, we measure how exactly each punctuation class is inserted. For disfluency detection task, we evaluate whether a token is correctly detected as a disfluency or not, regardless different sub-classes of the disfluency.

As an extrinsic evaluation metrics, we used the bilingual evaluation understudy (BLEU) (Papineni et al., 2002). Showing a relatively good correlation with human evaluation scores, BLEU is a widely-used metric in SMT tasks. The BLEU score is calculated by matching individual substrings within a string to a reference or multiple ones. Therefore, n-gram overlaps in translation output and reference translation are compared. Typically up to 4-grams are matched for the calculation. In order to avoid too short output, the penalty is applied to reduce the score.

The punctuation inserted and/or disfluency removed test data is translated into another language in order to demonstrate the impact of the two tasks on the machine translation performance. The MT performance in this case is measured in BLEU.

In this thesis, the metric BLEU is also used to calculate the similarity of texts after applying sentence reconstruction techniques. Favoring locally fluent outputs, BLEU can be a useful metric for this task.

4

Related Work

The importance of reformulating spontaneous speech has been emphasized throughout recent publications. In this chapter, we mainly focus on previous research on two major fields of this thesis. First, we will review previous research on segmentation and punctuation insertion for speech transcripts. The study on speech disfluency and sentence reconstruction, including its impact on the performance of machine translation will be given. Afterwards other research on spoken language translation tackling further issues will be reviewed.

4.1 Sentence Segmentation and Punctuation Insertion for Speech

The necessity of resegmentation of the ASR output was investigated in Rao et al. (2007b). In this work, the authors trained a sentence segmenter based on pause duration and language model probabilities. From the experiments, they emphasized that it is important to have commas in addition to periods within a sentence boundary, since commas can define independently translatable regions and eventually improve translation performance.

Ostendorf et al. (2008) gave a thorough analysis on speech segmentation and its impact on further downstream processes. Pointing out that it is impossible to process speech without some sort of segmentation, the authors reviewed types of segmentation in spoken language, traditional and recent models for modeling it, and the commonly used features in the models. It was also addressed that the definition of a sentence

boundary in spontaneous speech is less clearer than written text, due to its characteristics such as incomplete utterances and backchannel responses.

In previous work, the punctuation prediction problem was addressed to improve the readability as well as a subsequent step in the natural language processing (Huang and Zweig, 2002). In order to annotate ASR output with punctuation marks, they developed a maximum entropy (ME) based approach. In this approach the insertion of punctuation marks was considered a tagging task. An ME tagger using both lexical and prosodic features was applied and the model was used to combine the different features. Their work showed that it is hard to distinguish between commas and default tags, and periods and question marks, since there is little prosodic information (similarly short or similarly long pause durations) and the features can cover a span longer than bigrams. They achieved a good F-measure for both reference transcripts and transcripts produced by a speech recognition system.

The importance of sentence segmentation and punctuation was emphasized for users' readability again in Jones et al. (2005). They confirmed that sentence boundaries are essential for aiding human readability.

Lu and Ng (2010) presented a sentence boundary and punctuation prediction system using a sequential tagging tool. Their experiments were applied to transcribed conversational speech without relying on prosodic cues, on Chinese and English. From the experiments, it was shown that dynamic conditional random fields can outperform an approach based on linear-chain conditional random fields or a widely used appraoch based on a hidden event language model.

Segmentation within each sentence has been considered for better machine translation performance. In Wang and Waibel (1998), the authors improved the machine translation quality of spoken language by segmenting sentences into phrasal structures. This work made a contribution to emphasize the importance of defining separated zones for translation, in order to improve the translation quality.

Segmentation and punctuation issues are addressed together from an MT-driven perspective in Paulik et al. (2008). The authors modified phrase tables so that the target side contains commas, but the source side does not contain any. Thus, when this modified phrase table was applied during translation, it recovered commas on the target side. For the segmentation and periods after each new line, they used a sentence

segmenter based on a decision tree on the source side. They applied this method to three language pairs and achieved a significantly improved translation performance.

In Peitz et al. (2011) the authors made an extensive analysis on how to predict punctuation using a machine translation system. In this work, it was assumed that the ASR output already has the proper segmentation, which is sentence-like units. They investigated three different approaches to restore punctuation marks; prediction in the source language, implicit prediction, and prediction in the target language. Using a translation system to translate from unpunctuated to punctuated text, they showed significant improvements on the IWSLT evaluation campaign 2011.

The authors then extended this work using a hierarchical phrase-based translation system in Peitz et al. (2014). They show that long-range dependencies between words and punctuation marks can be captured robustly by the hierarchical phrase-based system. In addition, their monolingual translation system is tuned on F_2 rather than BLEU, which improved the accuracy and the following translation quality.

Recurrent neural networks are applied to the ASR output for punctuating it prior to translation in Kazi et al. (2015). The authors found that presenting each word in word vectors as well as having the recurrent state for the current word improved their punctuation prediction performance.

While the work mentioned above focused on enhancing punctuation accuracy or the machine translation performance when using the punctuated ASR output, some works are dedicated to research on MT performance for a given segment length, or latency. The empirical study on how utterance chunking influences machine translation performance is given in Fügen and Kolss (2007). In this work, machine translation performance is compared for each given segment length. From their experiments both on ASR hypotheses and reference transcripts, the authors show that even though chunking at the sentence boundaries is a good method, it is often not applicable for MT due to their length.

The authors in Sridhar et al. (2013) made an extensive study on different segmentation strategies and latency. They inserted segments based on various techniques into ASR output for real-time translation experiments. It was shown that a good performance can be achieved when they use the conjunction-based segmentation strategy along with a comma-based segmentation.

More algorithms to optimize segmentation strategies for simultaneous speech translation were proposed and compared in Oda et al. (2014). It was shown that the two methods based on greedy search and dynamic programming can effectively separate the source sentence into smaller segments, without harming the translation performance. This work is later further extended by Shavarani et al. (2015), where the authors present their investigation on segmentation in order to find the trade-off between latency and quality of spoken language translation. In order to address an issue of the previously suggested algorithm, Greedy-DP, where larger segments that can result in worse latency are preferred, the authors in this work suggested Pareto-optimality approach. In this approach they considered latency as an optimization parameter and achieved better/similar translation quality maintaining low latency.

Among different motivations for the sentence segmentation, Xu et al. (2005) split long sentence pairs in the bilingual training corpora to make full use of training data and improved model estimation for SMT. For the splitting they used the lexicon information to find splitting points. They showed that splitting sentences improved the performance for Chinese-English translation task. Similarly, to improve the performance of Example-based machine translation (EBMT) systems, Doi and Sumita (2004) suggested a method to split sentences using sentence similarity based on edit-distance.

4.2 Speech Disfluency Detection and Sentence Reconstruction

An early work in the automatic detection of speech disfluencies is based on statistical language model (Heeman and Allen, 1999). In their work, the speech recognition problem is redefined in a way that it includes the identification of POS tags, discourse markers, speech repairs, and intonational phrases.

Another early work (Levin et al., 1998) suggested to use an interlingua representation. Despite of a good performance in handling spontaneous speech, this approach was limited to only domain-specific dialogs.

The disfluency detection problem has been addressed using a noisy channel approach (Honal and Schultz, 2003). In this work it is assumed that fluent text, free of any disfluencies passed a noisy channel which adds disfluencies to the clean string. The authors use language model scores and five different models to retrieve the string,

where the two factors are controlled by weights. An in-depth analysis on disfluency removal using this system and its effect are provided in Rao et al. (2007a). They find that for the given news test set, an 8% improvement in BLEU is achieved when the disfluencies are removed.

In another noisy channel approach (Maskey et al., 2006), the disfluency detection problem is reformulated as a phrase-level statistical machine translation problem. Trained on 142K words of data, the translation system translates noisy tokens with disfluencies into clean tokens. The clean data contains new tags of classes such as repair, repeat, and filled pauses. Using this translation model based technique, they achieve their highest F-score of 97.6 for filled pauses and lowest F-score of 40.1 for repairs.

The noisy channel approach is combined with a tree-adjoining grammar to model speech repairs in Johnson and Charniak (2004). A syntactic parser is used for building a language model to improve the accuracy of repair detection. Same or similar words in roughly the same order, defined *rough copy*, are modeled using crossed word dependencies. Trained on the annotated Switchboard corpus, they achieve an F-score up to 79.7.

The automatic annotation generated in Johnson and Charniak (2004) is one of the features used for modeling disfluencies in Fitzgerald et al. (2009a), where they train a CRF model to detect speech disfluencies. In addition to the automatic identification by Johnson and Charniak (2004), they use lexical, language model, and parser information as features. The CRF model is trained, optimized and tested on around 150K words of annotated data, where disfluencies are to be classified into three different classes. Following this work, the authors offer an insightful analysis on syntactics and semantics of manually reconstructed spontaneous speech (Fitzgerald et al., 2009b).

In Liu et al. (2006), the authors explored different modeling techniques (i.e., the HMM, ME, and CRF approaches) on sentence unit and disfluency detection separately. It was shown that discriminative models are generally superior to the generative models for the given task, incorporating various features. The performance was measured in WER.

Though most of the progress has been focused on enhancing the performance of speech recognition via disfluency detection, authors of the work Wang et al. (2010) employ disfluency detection to achieve improved machine translation. They train three different systems. The first system combines hidden-event language models and

knowledge-based rules. The second system is a CRF model, which combines lexical features and shallow syntactic features. The final system is a rule-based filler-detecting system. Five classes are used in this task. The test sets for testing MT performance are generated by manually pulling out sentences with disfluencies from all sentences available. Thus, only the sentences containing disfluencies are selected and evaluated. There are two test sets built in this way, which contain 339 sentences and 242 sentences out of 1,134 sentences and 937 sentences respectively. Absolute improvements of 0.8 and 0.7 BLEU points are gained on the two selected test sets.

Sentence boundaries and speech disfluencies are studied together in other previous works. Combining prosodic and lexical information to detect sentence boundaries and disfluencies was demonstrated in the work of Stolcke et al. (1998), where decision trees are used to model prosodic cues and n-grams for the language model. The authors suggested that having large amounts of recognizer output as training data for the models can improve the prediction task as it lowers the mismatch between training data and test set.

In Wang et al. (2014) the authors presented an extensive study on various methods of combining punctuation prediction and disfluency removal. They applied their work to telephone speech data and evaluate it using F-score. Their models for punctuation prediction and disfluency removal are combined using either a cascade approach or a joint approach. Their results demonstrate clearly that both problems influence each other. The soft cascade system, where the decision label of the first prediction is embedded as a feature of the second step, outperforms the hard cascade approach where the second step is only performed on the output of the first step.

The impact of segmentation and disfluency removal on translation of conversational speech is investigated in Hassan et al. (2014). They separated the process into several steps. First they use a CRF model to detect sentence units. Based on these units they detect speech disfluencies, which are divided into two categories. After the simple disfluency is modeled using a CRF model, they use another CRF classifier to insert punctuation marks followed by a knowledge-based parser in order to remove more complicated disfluencies.

An in-depth analysis on automatic punctuation and disfluency detection in multi-part meetings has been made in Shriberg et al. (2001). In this work, the authors investigated the issue using prosodic cues, including duration, pitch features. Later

this work is extended to include and compare lexical clues (Baron et al., 2002). For lexical cues they also considered n-gram language models. They show that prosodic features can bring more robust performance, especially when recognition errors are considered.

The authors in Hough and Purver (2014) investigated on speech repair and edit term detection with minimal latency. In this work, they used information-theoretic measures from n-gram models as a principal decision features in order to detect different stages of repairs.

Speech reconstruction from a perspective of building grammatically correct sentences is discussed in Fitzgerald and Jelinek (2008). On the Fisher data (Cieri et al., 2004), they annotated 6K sentences in detail, in order to achieve more coherent and grammatical sentences.

Authors in Xu et al. (2012) showed how paraphrasing can be applied to change the style of a text. In this work, they paraphrased modern English into Shakespeare-styled English, and vice versa. Since they found out that BLEU tends to give an incomplete picture of system performance, the authors built and compared three different automatic metrics to measure the performance of paraphrasing. Based on cosine similarity, language models, and logistic regression, each of the three automatic metrics showed an improved correlation with human judgments. For the paraphrasing task itself, they used 31k aligned sentences to build the monolingual system to translate one to another.

This issue was discussed for speech in Neubig et al. (2012). In this work, a monotonic SMT-based model is used for creating clean transcripts from ASR transcripts based on in-depth analysis of the types of corrections. Implemented using weighted finite state transducers, their model could succesfully transform ASR transcripts into clean ones. The proposed method was able to perform deletions of redundant words, insertion of punctuation and omitted words, and correction of colloquial expressions.

Paraphrasing using a monolingual translation system (Quirk et al., 2004), within the noisy channel model in Brown et al. (1993), has shown its effectiveness. Trained on 139k sentences of large monolingual parallel data, the translation system showed a good performance to generate monolingual paraphrases.

4.3 Other Works on Spoken Language Translation

Apart from the two major points that this thesis tackles, many other works have shown further research related with spoken language translation.

The issue of quality of a translation of spoken language has been discussed extensively in previous research Kumar et al. (2014). Testing the SLT performance on the translation task of Spanish Fisher corpus (Post et al., 2013) (LDC2010S01 and LDC2010T04) to English, the authors showed that the ASR performance is negatively affected when recognizing conversational telephone speech. The inspiration from this work let the authors investigate the optimal coupling of ASR and SMT components later on in Kumar et al. (2015). In this work, authors compared different criteria to choose ASR hypothesis to translate for the Spanish-English Fisher translation task. Among other approaches, the authors could achieve the best performance when choosing the path which brings the best performance when it is translated monotone.

Tsvetkov et al. (2014) showed yet another approach to improve SLT. In this approach, they simulate likely mis-recognition errors and include them into the phrase table of a standard MT system. Their results demonstrated that using this technique brought a consistent improvement on several language pairs from English.

Inspired by simultaneous interpretation, authors in He et al. (2015) proposed to rewrite the reference translation. In order to support good translations while producing them promptly, they made an additional reference translation which is more monotone. By applying the rules that they generate upon linguistic knowledge, the word order of the newly created reference is closer to the source language. On Japanese to English translation, where there is a substantially big difference in word order, they could achieve better and faster translation.

There was another approach to learn from simultaneous interpretation (Shimizu et al., 2013). In their work, simultaneous interpretation data is collected, analyzed and incorporated into the learning process of the machine translation system. Results showed that this approach helped them to have a translation similar to that of experienced interpreters.

4.4 Comparison of this Work to Previous Works

There are notable differences between the techniques devised in this thesis and the previous works. First, the punctuation insertion system advised in Peitz et al. (2011) was limited to only punctuation marks within a given sentence boundary. As discussed earlier, the assumption in their work is that reliable sentence boundaries are already available. In this thesis, the monolingual translation system is used to predict sentence boundaries additionally. Thus, our model supports the actual end-to-end SLT scenario more closely, where sentence-like units are not offered by the ASR engine.

We extended the disfluency detection model shown in Fitzgerald et al. (2009a); Liu et al. (2006) further by using the novel features. The word representations from RNN and phrase table information for given source words and phrases are designed to capture deeper semantics.

While the performance of many of the previous works in punctuation insertion and disfluency detection is measured only in the domain of automatic speech recognition, namely in accuracy, in this thesis we measure the performance in terms of machine translation, in order to investigate the issues' significance and the developed models' impact in the further processes in NLP.

One of the most notable differences is that we integrated the disfluency detection scheme into SMT. Most of other works in this field leave the task as a separate, additional step, even if the output is used as an input of MT. By integrating the disfluency detection scheme into SMT, it is possible to choose clean words to translate based on weights of disfluency probability and MT components.

While the work in Hassan et al. (2014) showed an analysis on speech disfluency and punctuation issues in telephone speech of two speakers, this thesis addresses those in multi-party meeting in detail. Not only the multi-party meeting speech has an extensive amount of disfluency, but interactions between the multiple participants creates another challenge to build correct segment-like units. Also, in this thesis we show a novel scheme where two machine learning techniques are combined in order to detect speech disfluencies and augment sentence boundaries at the same time.

5

Spontaneous Data and Experimental Setup

Two different genres of in-house speech data, university lecture (Cho et al., 2014a) and multi-party meeting, are chosen to represent different degree of spontaneousness. In this chapter, details on the disfluency annotation, its categories and data statistics are described.

In the next part, we describe the automatic speech recognition we used in this work. The system is used to generate hypotheses for English. Also, the phrase-based machine translation systems for different language pairs that we used throughout in this thesis are also discussed in this chapter.

5.1 Spontaneous Speech Data

As discussed in Chapter 2, many issues arise when processing spontaneous speech due to its distinctive characteristics. The conventional method of modeling such linguistic phenomena is a supervised training, which requires annotated data.

There are several available spontaneous speech resources, such as Switchboard (SWBD) (Godfrey et al., 1992) and Fisher (Cieri et al., 2004). SWBD consists of telephone speech between two participants. Fisher data lacks detailed annotation on exact repair region, making it less efficient for our task, modeling speaker-generated errors. While this data is very well-resourced and already became a standard for the evaluation of automatic speech recognition and simple disfluency detection tasks, the

given prompts for speakers require a higher speaker cognitive load compared to other spontaneous speech corpora (Fitzgerald, 2009). Even though SWBD offers disfluency annotation, the spontaneousness we can model using this data is limited to English telephone speech between only two people. The AMI Meeting Corpus (McCowan et al., 2005) includes 135 sessions of multi-party interaction data, from both scenario-driven and real, spontaneous meetings, in which selected 28 sessions are annotated with disfluency (Germesin et al., 2008).

In order to model different degrees of spontaneous speech, we take two different genres of speech data. The university lecture data consists of German lectures given at the Karlsruhe Institute of Technology, which are transcribed and translated into English. The lecture data covers a broad range of topics in computer science. Another data set contains multi-party meetings held in English. Unlike the university lecture data, where there is usually a single speaker per each lecture, our multi-party meeting corpus involves 5 to 12 people in each meeting session. The interaction between the participants adds yet another degree of spontaneousness in the data.

In this section, we introduce these two data sets. Their characteristics as well as data statistics are also presented.

5.1.1 University Lectures

Most people who hold lectures tend to speak freely and do not read from a script. Compared to other styles of manuscript speech, such as political speeches or TED talks, which contain only very limited amounts of spontaneousness of the forms described in Section 5.3.1, university lectures express both more instances as well as varieties of spontaneousness.

The manual transcripts of the lecture data contain all the words, partial words, sounds and utterances of the speaker, including disfluencies. The disfluency annotation has been performed manually and on lectures that were previously recorded and transcribed. In this section, we describe how we annotated the disfluencies in the data and provide detailed statistics on the size of the corpus and the speakers. A special process applied to the English reference translation is also described.

5.1.1.1 Annotation

Prior to the annotation of the lecture corpus, we carefully examined the manual transcripts and explicitly chose lecture sets with a relatively high amount of disfluencies. In some rare cases, lectures showed characteristics of manuscript speech and thus had to be filtered out. The utterances of such lecturers were relatively clean and either lacked repetitions, corrections, filler words and so on, or showed very little of those.

Then, human annotators were asked to work on the data. Their first task was to read the transcripts in order to understand and follow the train of thought of the speaker. Afterwards, they marked disfluencies and characteristics of spontaneous speech by using the tags presented in the following section.

We aim to annotate starting and ending points of disfluent parts, so that removing the parts between the two points will generate cleaner and more readable sentences.

5.1.1.2 Disfluency Classes

The annotators distinguished several categories of disfluencies, namely repetitions and corrections, filler words and sounds, false starts, aborted sentences, and unfinished words.

Filler words and sounds often occur when a speaker hesitates. In our transcripts, we had a variety of filler words, for example *"uh"*, or *"uhm"* and also *"ähm"* and *"äh"*. In order to enhance the performance of the automatic processing, these various versions of fillers were unified into *"uh"*, or *"uhm"* respectively in our work. Words that only in some contexts are considered filler words remained unchanged. This class also includes discourse markers such as *"nun"* (*"now"*, *"well"*, in English) or *"ja"* (*"yes"*, *"right"* in English) in German.

Original transcript	äh das eine ist die ähm ist die Position, ja.
Disfluency annotation	<uh> das eine +/ist die/+ <uhm> ist die Position, <ja>.
English gloss	<uh> the one +/is the/+ <uhm> is the position <yeah>.
Reference	The one is the position.

Table 5.1: An example sentence of the filler and rough copy class in the lecture corpus. Filler words are unified.

Table 5.1 shows the unification of certain types of filler words along disfluency

tagging of another filler word and repetition. Since we observed mixed forms of the simple filler throughout our corpus, we changed *"ähm"* and *"äh"* into *"uhm"* and *"uh"*. Another filler word *"ja"* at the end of the sentence was also marked.

In spontaneous speech, repetitions and corrections occur when a speaker repeats her words. Repetitions can either be identical to the first utterance, or slightly different, because a certain part of a sentence is corrected. Such disfluencies are grouped together as **rough copy** in our work. Partial words can also occur in this class. An example of a repetition and a partial word is shown in Table 5.2, along with the literal translation and reference translation of the sentence. In this example, the verb and an additional word next to it *"werden da"* (engl. *"will be here"*) are forming an identical repetition. In the same sentence, a partial word *"Ka"* is also annotated as a rough copy, as it is a partial, but repetitive fragment of its next word *"Kapitel"* (engl. *"Chapter"*).

Disfluency annotation	... solche Dinge, die +/**werden da**/+ werden da vorgestellt, was ein ganz neues +/**Ka**=/+ Kapitel ist ...
English gloss	... such things, which +/**will be here**/+ will be here introduced, which a totally new +/**cha**=/+ chapter is ...
Reference	... things like that will be introduced there, which is a totally new chapter ...

Table 5.2: An example sentence of the rough copy class in the lecture corpus

Another class that we use in our work is **non-copy**, which is reserved for false starts or aborted sentences. This tag covers the case when a speech fragment is dropped and a new fragment is introduced, which is often observed at the start of a sentence. An example of this class is shown in Table 5.3. Here, we can observe that a different topic is introduced after the previous topic is dropped. The last token of the non-copy disfluency is tagged as a partial word as it is one from the full word *"rechteste"* (engl. *"furthest to the right"*).

Disfluency annotation	-/**Mit dem recht**=/- er würde wieder zurückgehen.
English gloss	-/**With the right**=/- it would again go back.
Reference	It would go back again.

Table 5.3: An example sentence of the non-copy class in the lecture corpus

As shown in the examples, our goal in this German lecture data annotation is to generate correct speech output. In our corpus, all disfluent parts that annotators believe to be deleted are marked as a disfluency in order to obtain clean, readable utterances. Partial sentences or phrases whose contents have been dropped by the speaker are also marked to be removed.

5.1.1.3 Sentence Reconstruction

From our manual analysis on the sentences where disfluency annotation was applied, we found out that even after disfluent words and filler sounds are removed, many of the spoken fragments are still grammatically imperfect. Many of them also include colloquial expressions that in general would rarely occur in written language. Some conjunctions, for example, are superfluous - they are not necessarily considered as a disfluency but can be removed in order to obtain a more formal-style sentence. As these properties affect user readability negatively, we desired another version of annotation offering a grammatically correct utterance.

So after the first version of disfluency annotation was done, the annotators corrected the sentences. They deleted repetitions, corrections and filler words and formed correct sentences. They were allowed to, if necessary, reorder words, and if there were no other possibilities to form a correct sentence, they could even leave out parts and change or add words. By doing so, we hope to get a grammatically correct version that is easier to understand and more fluent, while still preserving the content of the original, thus making it more suitable for use in subsequent automatic processes such as machine translation.

Similar attempts have been made for English conversational telephone speech. In Fitzgerald and Jelinek (2008), the authors published a small corpus consists of 6K sentence-like units chosen from the Fisher data (Cieri et al., 2004). In this corpus, each sentence is annotated in detail, how the sentence can be reconstructed in order to achieve better grammaticality.

These sentence reconstruction corpora are valuable resources to offer deeper insights into the structure of spontaneous speech. Despite its extreme difficulty, we believe this level of disfluency detection and correction will be potentially the future goal of the research in this area. This second version of annotation is therefore inevitable. As a result, we hence get two German versions: an unchanged one augmented with

disfluency tags, and another one considered to be a grammatically and linguistically correct German reproduction of the original sentence.

Manual transcript	Wenn Sie natürlich in der Vorlesung sitzen und *der Vorlesung* folgen, dann ist **Sprache**, die gesprochene Sprache, ein Problem.
English gloss	When you of course in the lecture sit and *the lecture* follow, then is **speech** the spoken speech a problem.
Disfluency annotation	Wenn Sie natürlich in der Vorlesung sitzen und der Vorlesung folgen, dann ist **+/Sprache/+** die gesprochene Sprache ein Problem.
English gloss	When you of course in the lecture sit and the lecture follow, then is **+/speech/+** the spoken speech a problem.
Disfluency annotation with reconstruction	Wenn Sie natürlich in der Vorlesung sitzen und *ihr* folgen, dann ist die gesprochene Sprache ein Problem.
English gloss	When you of course in the lecture sit and *it* follow, then is the spoken speech a problem.
Reference	Obviously, when you are sitting in the lecture and are following it, then spoken speech is a problem.

Table 5.4: An example sentence from the disfluency-annotated lecture corpus

Table 5.4 displays an excerpt of our annotated corpus, which shows the sentence reconstruction process described in this section as well as a reference translation. Words considered to be or causing disfluency are in bold letters. The first two rows show the original manual transcript of a German sentence along with its literal, gloss translation in English. It contains a repetition of the word *"Sprache"* (engl. *"speech"*). Therefore, in the next two rows representing the first annotated version, the word is marked with a repetition tag. Moreover, the fluency of this sentence can be clearly improved by replacing the words *"der Vorlesung"* (engl. *"the lecture"*) with a pronoun, as the noun is already used in the first part of the sentence. Finally, the last line offers a correct English reference translation, generated by human translators.

5.1.1.4 Reference Translation

The transcripts had been manually translated into English as described in Stüker et al. (2012b), prior to the disfluency annotation. Annotators, however, were also asked to check the English translation against the German source text, thereby completing their task. Although repetitions and other characteristics of spontaneous spoken language in the source sentence were not supposed to have been taken into account for the translation, and moreover are not needed for a readable reference translation, we found that sometimes the English translations still contained filler words or sounds, repetitions and corrections or unfinished or aborted sentences and words. In this case, we asked our annotators to also tag them, in order to make the reference more fluent.

No additional reference is created for the reconstructed sentences. The reference translation is based on the first version of the annotation. Therefore, it is possible that reference sentence does not exactly match the reconstructed sentences.

5.1.1.5 Corpus Details and Statistics

In this section, we will provide a detailed analysis of the disfluencies occurring in the lecture data. Relevant statistics on disfluencies will be given, including the amount of each sort of disfluency present in the corpus. Moreover, the proportions of different categories of disfluencies used by different speakers will be compared and discussed.

Table 5.5 shows the data statistics on disfluency classes for each speaker. Talk duration for each speaker is also shown, as well as the number of tokens of different disfluency classes and their proportions in each talk. Tokens include all words as well as punctuation marks. Therefore, one word or a punctuation mark is considered as one token.

Most of the talks are from computer science lectures held at our university. We have annotated 23 lectures from 17 speakers. Some of the talks are merged lectures from one speaker while some talks are only short excerpts from a lecture. Each talk has a different length, therefore we have largely varying numbers of tokens gathered. Statistics shows that the usage of certain types of disfluencies highly depends on the speaker.

Looking at the summed number, we have annotated around 130K tokens including punctuation marks, which correspond to 5,429 parallel sentences in German and En-

Speaker ID	Filler words		Rough copy		Non-copy		Non-disfluency		All tokens	(hh:mm:ss)
Speaker 1	4,782	11.50%	1,568	3.77%	458	1.10%	34,773	83.63%	41,581	04:05:46
Speaker 2	633	2.88%	504	2.29%	413	1.88%	20,465	92.96%	22,015	02:21:26
Speaker 3	550	3.97%	320	2.31%	97	0.70%	12,870	93.01%	13,837	01:27:04
Speaker 4	1,339	10.55%	789	6.22%	295	2.33%	10,264	80.90%	12,687	01:07:08
Speaker 5	607	6.14%	490	4.96%	76	0.77%	8,715	88.14%	9,888	00:59:29
Speaker 6	601	6.66%	308	3.41%	79	0.88%	8,040	89.06%	9,028	01:25:47
Speaker 7	126	2.28%	192	3.47%	64	1.16%	5,145	93.09%	5,527	00:46:53
Speaker 8	229	5.43%	33	0.78%	17	0.40%	3,938	93.38%	4,217	00:35:09
Speaker 9	418	12.67%	287	8.70%	83	2.52%	2,510	76.11%	3,298	01:13:52
Speaker 10	74	4.66%	34	2.14%	26	1.64%	1,455	91.57%	1,589	00:12:47
Speaker 12	41	4.27%	43	4.48%	7	0.73%	869	90.52%	960	00:05:19
Speaker 13	56	6.50%	71	8.25%	24	2.79%	710	82.46%	861	00:06:18
Speaker 14	15	1.82%	11	1.34%	14	1.70%	782	95.13%	822	00:05:47
Speaker 15	43	6.22%	8	1.16%	1	0.14%	639	92.47%	691	00:04:33
Speaker 16	26	4.01%	17	2.62%	2	0.31%	603	93.06%	648	00:04:56
Speaker 17	41	6.65%	48	7.78%	37	6.00%	491	79.58%	617	00:05:21
SUM	9,581	7.47%	4,713	3.67%	1,693	1.32%	112,269	87.53%	128,266	16:47:35

Table 5.5: Data statistics of lecture corpus, including classes of disfluency for each speaker

glish. The English reference consists of 113K tokens. The most common disfluencies are filler words and discourse markers, which represent around 7.5% of all tokens. Rough copy tokens correspond to 3.7% of all tokens. Non-copy disfluencies come to 1.3% of the whole corpus. More than 87.5% of the corpus are tokens without disfluencies.

5.1.2 Multi-party Meeting

In the previous section, we introduced our disfluency-annotated German lecture corpus, which is designed for modeling spontaneous speech phenomena of monologue speech.

NLP of multi-speaker speech presents unique research challenges. Various types of speech disfluencies have to be removed, while punctuation marks and sentence boundaries need to be inserted depending on the context or the speaker change. Similar to other spontaneous speech, multi-speaker speech contains a large number of disfluencies, including hesitations as well as repetitions, either exactly or vaguely the same, and speech fragments. In addition to these disfluencies, however, this genre of speech also includes interruptions between each other. Due to such interruptions, aborted speech fragments occur very often in multi-speaker speech.

A promising approach to use conventional state-of-the-art MT systems for translating multi-speaker speech is to transform it prior to the translation, so that the speech transcript from multiple speakers is closer in style to the training data of the MT systems. One of the difficulties of modeling spontaneous speech, however, is data sparsity, since it is usually modeled using manually annotated data.

For our second speech resource, we chose multi-party meeting data in order to support further research on speech phenomena of this genre. Compared to the previously introduced lecture data, this data will also stand as a different genre representing another degree of spontaneousness with its own distinctive characteristics.

In this section, our multi-party meeting corpus and its characteristics are described. Our corpus consists of project meetings between project participants on various topics. We use eight sessions, where each meeting session involves 5 to 12 different speakers. All meetings are held in English. As in real meeting scenarios, the meeting participants consist of native and non-native English speakers. The eight meeting sessions are transcribed and then disfluencies are manually annotated. In order to be able to evaluate our automatic models in a translation task, a certain portion of the data is chosen as test data and manually translated into French.

5.1.2.1 Annotation Process

The overall annotation process is applied in the same way as for the university lecture data. Once the disfluencies are annotated, the sentences are reconstructed as shown in Section 5.1.1.3. The reference translation in French, however, is generated only for selected parts of meetings due to time and cost constraints.

5.1.2.2 Speech Disfluencies

Disfluencies in the meeting data are annotated manually by human annotators. The same disfluency categories are used as in previous works on disfluencies (Fitzgerald et al., 2009a; Johnson and Charniak, 2004) and as the disfluency categorization for our German lecture data described in Section 5.1.1.

The class **filler** contains filler words as well as discourse markers, such as *"uh"*, *"you know"*, and *"well"*. As the class name suggests, **(rough)copy** includes an exact or rough repetition of words or phrases. In spontaneous speech, speakers may repeat what has already been spoken, as a stutter or a correction. For example, the sentence *"There is, there was an advantage"* has a (rough)copy tag on the phrase *"there is"*. Another class, **non-copy**, includes the cases where the speaker aborts previously spoken segments and starts a new segment. It can be rather moderate, so that the newly started fragment still has the same theme as the previously spoken segment. In a more extreme case, however, the speaker may introduce an entirely different topic in the new fragment. For example, in the following sentence from our meeting data the part before the comma is annotated as non-copy.

"I don't think it's the, the crucial thing is that we can compile with..."

After looking into the data, we decided that the disfluency annotations for the multispeaker speech task has to include an additional category, **interruption**. While the other three categories of disfluency can be used for other tasks such as monologue, the interruption class is devised for this new task. In multi-speaker speech, generally there are more than two speakers involved. Therefore, there are many parts of utterances which are interrupted by other speakers. Those segments which are interrupted and therefore could not be finished were classified as interruption.

Example Table 5.6 shows an excerpt from the meeting data. Filler tokens are marked with <>, and (rough)copy tokens are marked with +//+. non-copy tokens are tagged with −//−, and finally interruption are marked with #//#.

In this excerpt, the first speaker tried to start a new fragment (starting *"what"*), then a filler word is occurred (*"uh"*), and then the fragment is aborted, then yet another fragment is started (*"how far"*). But this last fragment is interrupted by the next speaker. We can also observe repetition.

A: I haven't heard anything, so I don't know -/what/-
<uh> #/how far/#
B: I will check for that.
C: Why is the API so hard?
We're waiting for a month now for this.
D: I don't know +/the last/+ the last meeting outcome <uh>
he said he could give us API at the end of the month.
C: Okay.

Table 5.6: Meeting data example with disfluency annotation

5.1.2.3 Corpus Details and Statistics

The number of tokens of each class of disfluencies and its proportion are shown in Table 5.7. Out of eight meeting sessions, five of them are taken as training data and three of them as testing data. The numbers do not include punctuation marks, but only words.

Class	Training		Testing	
filler	2,666	6.9%	999	6.7%
(rough)copy	2,232	5.8%	1,017	6.8%
non-copy	802	2.1%	331	2.2%
interruption	1,350	3.5%	864	5.8%
clean	31,507	81.7%	11,660	78.4%
SUM	38,557	100%	14,871	100%

Table 5.7: Meeting data statistics

Both the training and test data have a disfluency rate of around 20%, which is much

higher than the rate reported in Section 5.1.1, where lecture data has a disfluency rate of roughly 10%. Around 7% of the word tokens in the meeting data are simple disfluencies, or filler words, while the other 11 to 15% are more complicated disfluencies.

Analysis on Segment Length The training data shown in Table 5.7 consists of $4.6k$ sentences, while the test data has around $2.1k$ sentences. We found that the multi-party meeting data has the characteristic that each segment is rather short. On average, for all the meeting data we have, there are around 8 words per segment. This is quite short compared to, for example, the lecture data, which has around 15 words per segment. We also compared the number of segments to the training data of our MT system, which consists of mainly parliamentary proceedings and news text. This data has around 24 words per segment.

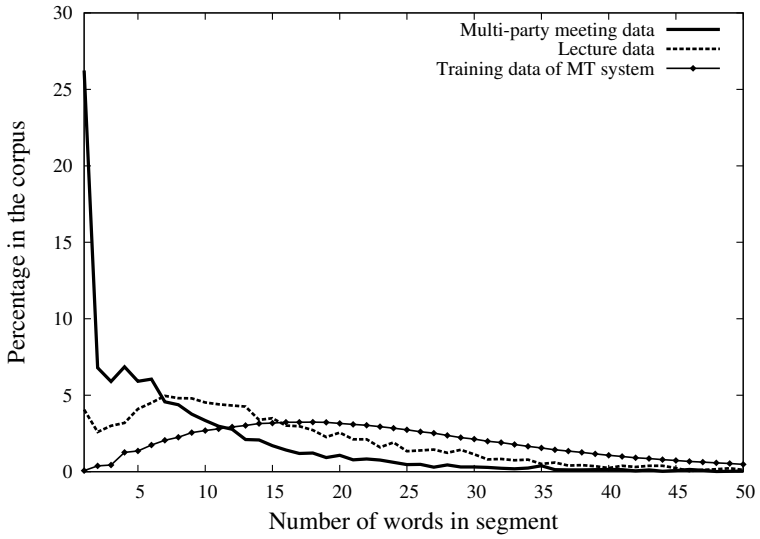

Figure 5.1: Statistics on number of words in segment

Figure 5.1 depicts the distribution of the segment length for every corpus. In the meeting data short segments are the majority, especially one word segments. There are many segments which only consist of a single word, such as *"yes"* or *"okay"*. Although some of them are discourse markers and therefore annotated with the filler disfluency, some of them are also left intact when those tokens are actually used to convey meaning.

This results in another challenge when detecting disfluencies in meeting data. Another cause of the short segments is that there are also many short segments which are interrupted by other speakers and are therefore aborted. The lecture data, which also consists of spoken language, also has a higher frequency of shorter segments, compared to the conventional MT training data which has more segments whose length is longer than 15 words.

5.1.3 Summary

In this thesis, we aim to model two different degrees of spontaneousness by using both university lecture and multi-party meeting data. In this section, we introduced the disfluency-annotated KIT lecture corpus and the multi-party meeting corpus designed for spoken language processing and translation. The goal of building the corpora is to give insights of spoken language phenomena in different genres. These corpora were used to model and test our models built in this thesis in terms of performance of subsequent applications, such as machine translation systems.

The largest part of our lecture corpus covers diverse topics related to computer science, and contains various speaking styles from 17 different speakers. The multi-party meeting data consists of project meetings between 5 to 12 participants. The speech disfluencies and the characteristics of the two data sets were discussed in this section.

5.2 English Automatic Speech Recognition System

In this section, we discuss the ASR system we use throughout this work. The English audio data is decoded using our ASR system.

The speech recognition is performed using the Janus decoder (Soltau et al., 2001) in an online setup. Using a framesize of 32ms and a frameshift of 10ms the audio stream is converted in a stream of 40 dimensional lMel feature vectors.

The hybrid DNN/HMM acoustic model uses a context dependent quinphone setup with three states per phoneme, and a left-to-right HMM topology without skip states. The neural network has an input window of ± 6 frames leading to an input layer size of 520 neurons, this is followed by 4 layers of 2k neurons and a finial classification output layer containing just over 8k neurons.

The neural network is pretrained layerwise using denoising autoencoders with a 20 million mini batches. After pretraining the final layer is added, with the output layer using the softmax activation function. The full DNN is then fine-tuned using the newbob learning rate schedule (Senior et al., 2013). All training is performed using Theano (Bergstra et al., 2010) on the TED (Cettolo et al., 2013) and Quaero data (Stüker et al., 2012a).

For the language model training texts from various sources such as webdumps, scraped newspapers and transcripts are used. The 120k vocabulary is selected by building a Witten-Bell smoothed unigram language model using the union of all the text sources' vocabulary as the language models' vocabulary (global vocabulary). With the help of the maximum likelihood count estimation method described in Venkataraman and Wang (2003) we found the best mixture weights for representing the tuning set's vocabulary as a weighted mixture of the sources' word counts thereby giving us a ranking of all the words in the global model by their relevance to the tuning set.

Using this vocabulary, language models are built from each of the sources and interpolated using the SRILM toolkit (Stolcke) so as to maximally reduce the perplexity of the tuning set.

5.3 Machine Translation Systems

This section aims to give the reader a brief review on SMT systems we used in this thesis. The detailed description on data sets used for building the translation models is also given.

In order to build the phrase table, we use the Moses package (Koehn et al., 2007). Unless it is stated separately, the alignment is obtained from the IBM-4 model using GIZA++ (Gao and Vogel, 2008; Och and Ney, 2003). We used the SRILM Toolkit (Stolcke) for building the LMs.

Word reordering variants are encoded in our lattice. This lattice, afterwards, then is used as input to the decoder. The different criteria on word reorderings for each language pair will be given in the following sections.

In all our SMT systems, our models are optimized using the minimum error rate training (MERT) (Och, 2003) implemented in Venugopal et al. (2005) so that we can

reach the best BLEU score. This process is repeated for several iterations. The optimized weights are applied to translate the test data. Our translations are generated by our in-house phrase-based decoder (Vogel, 2003).

5.3.1 Training Data

Most of the conventional MT systems are built using manuscript-style data as their main training data. Manuscript-style data has well-defined sentence boundaries and few speech disfluencies. While our MT components are also utilizing the manuscript-style data for training, we use some parts of data from speech as in-domain data, in order to adapt our models into speech translation. In this section, we briefly introduce different sources and styles of data.

In this thesis, we use two different test sets for our experiments of translating spontaneous speech. University lecture data is in German, and we evaluate our techniques by translating it into English. On the other hand, multi-party meeting data is in English and the performance is measured by translating it into French. Details of the two spontaneous data sets are given in Chapter 2.

One large parallel corpus available for building a large-scale SMT system is the European parliament proceedings parallel corpus (EPPS) (Koehn, 2005), which consists of proceedings from the European parliament. It is available for many language pairs in European Union. We use the German-English and English-French parallel corpus for training data of SMT systems.

Another corpus used for training translation models is the News commentary corpus (NC). This data contains mainly opinions and commentaries about politics and economics and is translated into different languages.

For our MT systems, we use some part of the openly-available spoken-style data as our in-domain data.

TED TED[1] is a online platform, where talks in various topics are shared. The presentations are given by invited speakers, and translated into different languages by volunteers. Since the TED data consists of audio, manual transcript and translation of each talk, it has been a valuable resource for many different NLP tasks.

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

Although TED data is stemmed from speech, the talks are scripted compared to other spoken-style data, such as lectures or meetings. Therefore, TED talks in general contain rather limited spontaneousness in the speech. The characteristics of spontaneous speech, such as stutters or repetitions, are observed much less frequently in TED talks.

5.3.2 German to English System

The translation system is trained on 1.76 million sentences of German-English parallel data including the European Parliament data and the News Commentary corpus. We also use the parallel TED data as in-domain data to adapt our models to the lecture domain. Preprocessing which consists of text normalization, tokenization, and smartcasing is applied before the training. For the German side, compound splitting (Koehn and Knight, 2003) and conversion of words written according to the old spelling conventions into the new form of spelling are applied additionally.

As development data, manual transcripts of lecture data collected internally at our university are used. The talks are 14K parallel sentences from university classes and events.

A 4-gram language model is trained on 462 million words from the English side of the data using the SRILM toolkit Stolcke. A bilingual language model (BiLM) (Niehues et al., 2011) is used to extend source word context. In order to address the different word orders between German and English, the POS-based reordering model as described in Rottmann and Vogel (2007) is applied. The POS information for this reordering is learned from Schmid (1994). The reordering model is further extended as described in Niehues and Kolss (2009) to cover long-range reorderings.

5.3.3 German to English Lecture Translation System

We build yet another German to English MT system in addition to the system discussed in Section 5.3.2. In order to make use of the lecture data described in Section 5.1.1, we train another SMT system whose in-domain data is the lecture data. Therefore, when only the partial lecture data is needed for testing and the whole lecture data is not required for modeling of other tasks beforehand, we use this system to translate test sets.

We use the parallel TED data and manual transcripts of lecture data containing 63k sentences as indomain data and adapt our models at the domain. To better cope with domain-specific terminologies in university lectures, Wikipedia[1] title information is used as presented in Niehues and Waibel (2011).

The translation system is trained on 1.8 million sentences of German-English parallel data including the European Parliament data and News Commentary corpus. Before the training, the data is preprocessed and compound splitting for the German side is applied. Preprocessing consists of text normalization, tokenization, smartcasing, conversion of German words written according to the old spelling conventions into the new form of spelling.

The 4-gram language model is trained on the 425 million words. The BiLM and reordering models are applied in the same as in Section 5.3.2.

5.3.4 English to French System

The English-to-French translation system is built on 2.3 million parallel sentences. The training data includes the European Parliament data and the News Corpus data. The noise-cleaned common crawl data is also utilized. The system also includes a small amount of spoken-style data such as TED, which is used as in-domain data on which the models are adapted. Manual transcripts of some of the TED data are used as development data for the translation system.

We use a 4-gram language model built as well as a BiLM (Niehues et al., 2011). The POS-based reordering model is using only short-range-based reorderings.

5.3.5 English to German Online System

In order to evaluate our online punctuation insertion schemes, we translate the test sets with different segmentation and punctuation marks into German. For the translation, we use our online English to German phrase-based translation system. The system is trained on the parallel corpus of Europarl, News commentary, TED, and the noise-filtered common crawl data. For the monolingual data we take the News Shuffle corpus. Detailed statistics on corpus can be found in Slawik et al. (2014).

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

We build a 4-gram language model on the German side of TED data which is used as an in-domain language model. In addition to this language model, we used a BiLM on all available parallel data as described in Niehues et al. (2011). Also, we used a 4-gram language model on a data that is sampled based on cross entropy with the development data. For the in-domain TED data, we applied the cluster algorithm (Och, 1999). Once the TED data is clustered into $1,000$ classes, we build a 9-gram language model and used it as an additional model.

In order to address the word order difference between English and German, we use the POS-based reordering (Rottmann and Vogel, 2007) along with the tree-based (Herrmann et al., 2013) and lexicalized reordering rules.

For evaluating differently segmented test sets, we use the Levenshtein minimum edit distance algorithm (Matusov et al., 2005) in order to align hypothesis against the reference translation.

6

Segmentation and Punctuation Insertion

In spoken language translation, finding proper segmentation and reconstructing punctuation marks are not only significant but also challenging tasks. Previous research on segmentation and punctuation insertion (Paulik et al., 2008) emphasized the importance of the task in order to improve the MT performance. In this chapter, we discuss different techniques to insert segmentation and punctuation marks into speech transcripts and measure their performance from the perspective of MT quality.

Oracle experiments and their scores show how important it is to have proper punctuation marks on ASR transcripts as well as manual transcripts. Also, the oracle experiments show the upper bound of this task, establishing up to which point we can improve the translation quality by inserting punctuation marks. The description on which the experiments are conducted and which system is taken for evaluating the impact of segmentation and punctuation will be given.

Machine translation-based approach (Peitz et al., 2011) to insert punctuation marks within each sentence boundary showed good potential for MT of speech transcripts. Inspired by this, we extended this approach so that proper sentence boundaries can be predicted using a monolingual translation system (Cho et al., 2012). Thus, a technique to insert punctuation marks and sentence boundaries using a monolingual translation system is described in the following section. The technique is applied to our German lecture data and the performance is measured by translating it into English.

Another crucial aspect of segmentation and punctuation on speech transcripts is the

latency issue. While a longer context can boost the accuracy of inserted punctuation marks, it drastically increases the delay in the spoken language translation system. In the following section, we will study punctuation insertion system from the perspective of latency (Cho et al., 2015b). We investigate the impact of shorter context in punctuation insertion task on simultaneous speech translation system.

An empirical study (Fügen and Kolss, 2007) showed how the machine translation performance is affected by choosing different segment lengths. In Sridhar et al. (2013), the authors tried a grammar-oriented segmentation. On the other hand, greedy search and dynamic programming (Oda et al., 2014; Shavarani et al., 2015) have shown a good performance to maintain MT performance while decreasing the latency.

In this thesis, we suggest a new scheme within stream decoding where the time delay consumed on punctuation prediction is avoided. The scheme is built and tested for English TED talks. The performance of the punctuation insertion system is measured in terms of MT quality by translating it into German using our online system. Our evaluations show that our suggested scheme can be used as an efficient method to punctuate recognized streams in real-time scenarios. While outperforming a conventional language model and prosody based punctuation prediction system, our model maintains performance comparable to systems that require longer contexts.

6.1 Oracle Experiments

In order to investigate the impact of segmentation and punctuation marks on the translation quality, we conduct two experiments.

In the first experiment, we apply human-transcribed segments and punctuation marks to the output of the speech recognition system. Thus, words are still from an ASR system, but the segments and punctuation marks are reused from a human-generated transcript. In the second experiment, the segments in the output of the speech recognition system are applied to the human-generated transcripts. In this case, words are transcribed by human transcribers, but segmentation and punctuation are from an ASR system.

From these experiments we can observe how much impact the better segmentation and punctuation have for the performance of ASR output translation. We can also find how the segmentation according to an ASR system affects manual transcripts.

6.1.1 Genre and System

For the oracle experiments, we choose parts of the German lecture data for development and testing introduced in Section 5.1.1. As removing simple filler words such as *uh* and *uhm* is trivial and the existence of them differs greatly from the training data of the models, we simply removed the filler words from the lecture data.

For development and testing, we use the lecture data from different speakers. These are also collected internally from university classes and events. They consist of talks of 30 to 45 minutes and the topic varies from one talk to the other. For the development set we use manual transcripts of lectures, while for testing we use the transcripts generated by an ASR system. The development set consists of 14K parallel sentences, with 30K words on the source side and 33K words on the target side including punctuation marks. Detailed information on the source side of the test set, including the word error rate (WER) of the recognition output, can be found in Table 6.1.

ASR output	Sentences	2,393
	Words without punctuation marks	27,173
	WER	20.79%
Manual Transcript	Sentences	1,241
	Words	29,795
	Words without punctuation marks	26,718
	Periods	1,186
	Commas	1,834
	Question marks	55

Table 6.1: Information on the preprocessed source side of the test set

For translating different test sets, we use the MT system described in Section 5.3.3.

6.1.2 Oracle 1: Insertion of Manual Segments and Punctuation Marks into ASR Output

Applying manual segments to the output of an ASR system requires the time stamp information for each utterance. We use this information from manual transcripts and segment the output stream generated by the ASR system according to it. The alignment information between the ASR test sets and their manual transcripts is learned in

order to insert punctuation marks. As punctuation marks, we consider period, comma, question mark, and exclamation mark. Punctuation marks such as period, question mark, and exclamation mark are usually followed by a new segment in manual transcripts, and commas are useful to define independently translatable regions (Rao et al., 2007b).

Depending on which punctuation marks are inserted, three hypotheses are considered in this experiment.

- MTSegment: correct segments from a manual transcript are applied to the ASR test set.
- MTSegmentFullStop: correct segments and ".?!" from a manual transcript are applied to the ASR test set.
- MTSegmentAllPunct: correct segments and ".,?!" from a manual transcript, including commas, are applied to the ASR test set.

Therefore, the results in the hypothesis MTSegment show the upper bound of performance improvement when the proper segmentation is given, while the hypothesis MTSegmentAllPunct shows the scenario when we also have good punctuation marks additionally. With the hypothesis MTSegmentFullStop, we intend to investigate how helpful it is for the translation quality to have commas or not.

To show the impact of the different segmentations according to the ASR system and according to the hypothesis MTSegmentAllPunct, several consecutive segments are extracted from our test set. The original ASR output for this excerpt with its simple language model based segmentation and reference translation of the segments are given in Table 6.2. The manual transcript for this excerpt is also given together. The ASR system generated no word errors for this excerpt, but the sentence boundary of the manually created transcript differs greatly from the automatically inserted ones.

The translations of the ASR output with different segmentations are presented in Table 6.3. The two source texts contain the same recognized words from the ASR system, but different segmentation and punctuation are applied. We can observe that when the text uses the manual transcripts' segmentation, the translated text conveys the meaning of the sentence substantially better. It also provides improved readability enormously. For example, the German participle *"gesprochen"*, which was translated

Manual Transcript	wir sehen hier ein Beispiel aus dem Europäischen Parlament. Europäischen Parlament werden zwanzig Sprachen gesprochen, und man versucht durch Hilfe menschlicher Übersetzer, Simultanübersetzer die Reden der Sprecher jeweils in andere Sprachen hinein zu übersetzen. ist es möglich, Computer zu bauen die ähnliche Übersetzungsdienste leisten?
ASR Output	wir sehen hier ein Beispiel aus dem Europäischen Parlament Europäischen Parlament werden zwanzig Sprachen gesprochen und man versucht durch Hilfe menschlicher Übersetzer Simultanübersetzer die Reden der Sprecher jeweils in andere Sprachen hinein zu übersetzen ist es möglich Computer zu bauen die ähnliche Übersetzungsdienste leisten
Reference	Here we see an example from the European Parliament. There are twenty languages spoken in the European Parliament, and people have tried to translate the talks of the speakers to the other languages respectively, by means of human translators, simultaneous interpreters. Is it possible to build computers that perform similar services?

Table 6.2: ASR output and reference translation of the excerpts

into *"spoken"* using MTSegmenatAllPunct, is lost in the first segment in the ASR system and segmented into the next line. This leads to the loss of the information about this participle during the translation. An article and its following noun, *"die Reden"*, are also split using the original segmentation of the ASR system. It becomes the reason why the more suitable word *"(the) speeches"* in this context is not chosen, but *"Talk"*.

6.1.3 Oracle 2: Insertion of ASR output segments into manual transcripts

In addition to the insertion of proper segmentation and punctuation into the output of the ASR system, we perform another experiment where the segmentation in the output of the ASR system is applied to manual transcripts.

Although the segmentation from ASR output is obtained by incorporating language

Segmentation	Translation
ASR	We see here is an example from the European Parliament, the European Parliament 20 languages And you try simultaneously by help human translator translators the Talk to each of the speaker in other languages to translate it is possible to build computers The similar to provide translation services
MTSegment-AllPunct	We see here is an example from the European Parliament. The European Parliament 20 languages are spoken, and you try by help human translator to translate simultaneously translators the speeches of the speaker in each case in other languages. It is possible to build computers that are similar to provide translation services?

Table 6.3: Translation using different segmentation according to ASR output and MT-SegmentAllPunct hypothesis

model probability and prosodic information such as pause duration, it is often not the best segmentation especially for spontaneous speech. One explanation is that defining a sentence boundary is less clearer in spontaneous speech than in written text (Ostendorf et al., 2008), due to distinctive phenomena of spontaneous speech. Also, while conversation speech may contain unique information different from other sources, historically models and most research concern broadcast news as a main data source, which is a fairly clean and closely resembles written documents (Ostendorf et al., 2008).

6.1.3.1 Language Model and Prosody Based Segmentation

In this section, we briefly describe our language model and prosody based segmentation model. The language model and prosody based segmenter employs a 4-gram language model trained on punctuated text. In order to predict punctuation marks a context of four words, two prior and two after the possible punctuation mark, is taken into consideration.

The language model is used to calculate three scores. The first one is the score without an inserted punctuation mark as

$$P(w_{i-1}, w_i, w_{i+1}, w_{i+2}) \tag{6.1}$$

while the second one is the score with a comma.

$$P(w_{i-1}, w_i, \texttt{@COMMA}, w_{i+1}, w_{i+2}) \tag{6.2}$$

The last one is calculated by followings.

$$P(w_{i-1}, w_i, \texttt{@STOP}, w_{i+1}, w_{i+2}) \tag{6.3}$$

The similar approach has been applied for speech disfluency processing in Stolcke and Shriberg (1996). The authors suggested a hidden-events language model to predict disfluencies probabilistically. Their language model, though, is developed to be used for speech decoding, lowering perplexity, while our language model-based segmentation is applied to our 1-best hypothesis.

A dynamic scaling factor is applied to the punctuation mark scores in order to prevent both very short sentences and very long sentences. In parallel to the language model a prosody component searches for pauses over t_θ seconds and then force terminates any sentences.

ASR output	wir haben somit also auch ein drittes Standbein in Asien in in chinesischen Raum in Hongkong
Reference	wir haben somit also auch ein drittes Standbein in Asien, im chinesischen Raum, in Hongkong.

Table 6.4: Disfluency and its affect on the automatic segmentation (Reference translation: Thus we consequently also have a third foot hold in Asia, in the Chinese region, in Hong Kong.)

Table 6.4 depicts an example of incorrect automatic segmentation caused by disfluencies. As the speaker stutters, the automatic segmenter of the ASR system based on pause duration and a language model trained on clean texts inserts a new line.

6.1.3.2 Experimental Setup

In this experiment, we analyze the following three scenarios.

- ASRSegment: a manual transcript was segmented according to the segmentation of the ASR output.

- ASRSegmentComma: a manual transcript was segmented according to the segmentation of the ASR output, and commas are removed.
- ASRSegmentAllPunct: a manual transcript was segmented according to the segmentation of the ASR output, and all four punctuation marks are removed.

The four punctuation marks correspond to ".,?!" as in the first oracle experiment. To segment a manual transcript as in the ASR output, we use an algorithm which is commonly used for evaluating machine translation output with automatic sentence segmentation (Matusov et al., 2005). This method is based on the Levenshtein edit distance algorithm (Levenshtein, 1966). By backtracing the decisions of the Levenshtein edit distance algorithm, we can find the Levenshtein alignment between the reference words and the words in the ASR output.

In this work, the ASR output plays the role of a reference and using this algorithm we are able to find a resegmentation of the human reference transcript based on the original segmentation of the ASR output.

6.1.4 Results

Table 6.5 depicts the results of the two oracle experiments in numbers. The scores are reported as case-insensitive BLEU scores, without considering punctuation marks. This aims at analyzing the impact of the segmentation and punctuation solely on the translation quality.

System		BLEU
ASR		**20.70**
Oracle 1	MTSegment	21.42
	MTSegmentFullStop	22.18
	MTSegmentAllPunct	22.48
Transcripts		**27.99**
Oracle 2	ASRSegment	26.38
	ASRSegmentComma	26.36
	ASRSegmentAllPunct	25.54

Table 6.5: Influence of oracle segmentation and punctuation on the speech translation quality

For the hypotheses MTSegment, ASRSegmentAllPunct and tests on the ASR output, we create phrase tables removing punctuation marks on the source side in order to make a better match between the test set and the phrase table. To evaluate the translation hypotheses of ASR output and the ASRSegmentation experiments, we re-segmented our translation hypotheses to have the same number of segments as the reference as shown in Matusov et al. (2005).

From this table we observe that having the correct segmentation and punctuation improves the translation quality significantly. When the human-transcribed segmentation and punctuation are available, an improvement of 1.78 BLEU is observable on the test set.

Another interesting point is when we compare MTSegmentAllPunct to MTSegmentFullStop, we see the steady improvement of 0.3 BLEU in translation from having commas on the source side. This is congruent with the findings in Rao et al. (2007b), that inserting commas in addition to periods improves translation quality. In our case, the scores are evaluated ignoring punctuation marks. Thus, the improvement on BLEU means that by having proper punctuation marks the translation quality itself can be improved.

On the other hand, we can observe from Table 6.5 that by simply changing the segmentation of the transcripts we lose 1.6 BLEU scores in translation performance. As shown in Table 6.1, there are almost twice as many segments in the ASR output compared to the manual transcript. This can be one reason for the drastic drop in translation quality. We also observed from this translation that incorrect reordering of words occasionally happens within a segment, when the segment is not a sentence-like unit but a part of a sentence.

Removing commas from ASRSegment does not result in a big performance drop in ASRSegmentComma. Often, the segments from the ASR system do not match with the phrase boundaries learned in the text translation system, which results in having fewer independently translatable regions separated by commas. In addition to this, losing all punctuation information leads to a further performance drop of 0.84 BLEU scores.

6.2 Monolingual Translation System

The first approach to be described is a monolingual translation system. It is an MT system, which translates a non-punctuated text into a punctuated and properly segmented text. We build a monolingual translation system from German to German implementing segmentation and punctuation prediction as a machine translation task. When using the monolingual translation system to punctuate the German lecture data before translating it into English, we get an improvement of 1.53 BLEU points on the lecture test set. This is a comparable performance to the upper bound drawn by the oracle experiments.

6.2.1 Model

Inspired by Peitz et al. (2011), we build a monolingual translation system to predict segmentation and punctuation marks in the translation process. This monolingual translation system translates non-punctuated German into punctuated German. Using this system we predict punctuation marks as well as segmentation before the actual translation of the test sets. The output of this system becomes the input to our regular text translation system which is trained using training data with punctuation marks.

When translating the output of the monolingual translation system, no preprocessing is applied as the test set is already preprocessed before going through the monolingual translation system. The monolingual translation system neither alters any words nor reorders them, but it is used solely for changing segments and inserting punctuation marks.

In order to build this system, we first process the training data to make the source side not contain any punctuation marks, while the target side contain all punctuation marks. The training data statistics on the target side is shown in Table 6.6.

Words	46.32M
Periods	1.76M
Commas	2.88M
Question marks	0.10M
Exclamation marks	0.07M

Table 6.6: Information on the preprocessed punctuated German side of the training data

For a language model, we use 4-gram and it is trained on the punctuated German data. Also, no reordering model is used as we use the monotone alignment.

The difference of our monolingual translation system to the work in Peitz et al. (2011) is that in our work the monolingual translation system is used to predict sentence segmentation additionally. In their work, it was assumed that the segmentation of the speech recognition output was given and corresponded to at least sentence-like units. Therefore, their monolingual translation system was used to reconstruct punctuation marks only with using three different strategies.

It was shown in Section 6.1 that the segmentation generated from an ASR system using a language model and prosody is not necessarily the best segmentation, especially when the recognized text is spontaneous speech with less grammatical sentences and more disfluencies. In this work, we aim at improving segmentation in addition to inserting punctuation marks using this monolingual translation system. Performing this requires a modification to the training data as well as development and test sets.

6.2.1.1 Data Preparation

Usually training data for conventional text translation systems is segmented by human transcribers so that it has punctuation such as a full stop, a question mark, or an exclamation mark at the end of each line. Therefore, if we use this training data to translate the ASR test sets, translation models would more likely insert a punctuation mark at the end of every line of the ASR test set during translation. From this observation, we resegment training corpora randomly so that every segment is not necessarily one proper sentence-like unit. The development set is modified in the same way.

The test sets for this monolingual translation system are also prepared differently, using the idea of a sliding window. Exemplary sentences from our test set are shown in Table 6.7. In this table, each line contains 8 words. The first line starts with a word *"der"*, and in the second line, we have the next starting word *"bildet"*, which was the second word in the first line. At the same time, we encounter a new word *"gesehen"* at the end of the line.

When the length of a sliding window is l, each line consists of l-1 words from the previous line and 1 new word. Thus, the nth line contains the nth to $n+l-1$th word of a test set. The test set prepared in this way has the same length as the number of words in the original test set. In this way we can have up to l spaces between words.

For those spaces we want to investigate how probable it is to have a punctuation mark in that word space. In this experiment, we constrain the length of sliding window l to 10.

This differently formatted test set enters the monolingual translation process in a normal way, line by line. The translation of the test set shown in Table 6.7 using our monolingual translation system is illustrated in Table 6.8. We see that words such as *"Normalform"* and *"gesehen"* are followed by certain punctuation marks.

der	bildet	die	sogenannte	konjunktive	Normalform
bildet	die	sogenannte	konjunktive	Normalform	wir
die	sogenannte	konjunktive	Normalform	wir	haben
sogenannte	konjunktive	Normalform	wir	haben	gesehen
konjunktive	Normalform	wir	haben	gesehen	dass
Normalform	wir	haben	gesehen	dass	wir
⋮	⋮	⋮	⋮	⋮	⋮

Table 6.7: Test data preparation for the monolingual translation system. The excerpt corresponds to in English: *it forms the so-called conjunctive normal form we have seen that we.*

der	bildet	die	sogenannte	konjunktive	Normalform.
bildet	die	sogenannte	konjunktive	Normalform.	Wir
die	sogenannte	konjunktive	Normalform.	Wir	haben
sogenannte	konjunktive	Normalform.	Wir	haben	gesehen,
konjunktive	Normalform.	Wir	haben	gesehen,	dass
Normalform.	Wir	haben	gesehen,	dass	wir
⋮	⋮	⋮	⋮	⋮	⋮

Table 6.8: Test data punctuated using the monolingual translation system

6.2.1.2 Punctuation Prediction Criteria

A punctuation mark is chosen if the same punctuation mark is found same or more often than a given threshold. If more than one punctuation mark appears more than the threshold in the same word space, the most frequent one is chosen. For example,

we can observe that the word *Normalform* is followed by a full stop in multiple lines, as shown in Table 6.8. At the same time, *gesehen* is often followed by a comma. We examine how often a certain punctuation mark is following which word, and apply a threshold to decide whether we should extract this punctuation mark. There are some cases where we have the same frequency for multiple punctuation marks; in this case we put a different priority on punctuation marks. For example, in this experiment we put higher priority for a period over a comma.

In this experiment, we evaluate the translation quality over a varying threshold, from 1 to 9. We exempt the case when the threshold is 10, the length of the sliding window. In this case, one punctuation mark has to appear all the 10 word spaces after a word in order to be inserted. This condition is so restrictive that only few full stops are generated, which causes unaffordable computational time consumption for the translation procedure.

In the same way as in the oracle experiment, we consider four punctuation marks here: period, comma, question mark, and exclamation mark. A new segment is introduced when either a period, question mark, or exclamation mark is predicted, in order to have congruence with the manual transcripts.

In order to make the hypotheses comparable with the oracle experiments shown in Section 6.1, we considered three different hypotheses of reconstructing segmentation and punctuation.

- MonoTrans-Segment: monolingual translation system is used for segmentation prediction only.
- MonoTrans-FullStop: monolingual translation system is used for segmentation and full stop prediction.
- MonoTrans-AllPunct: monolingual translation system is used for segmentation and all punctuation marks prediction.

6.2.2 Experiments and Results

For consistency, we applied the techniques to the data and system described in Section 6.1.1.

In order to analyze the effect of the varying threshold for the monolingual translation system, first we use the same threshold value for all punctuation marks. The number of punctuation marks predicted using the same threshold are shown in Table 6.9. As shown in the table we could predict periods and commas, but we could not generate question marks or exclamation marks. A reason might be that question mark and exclamation mark are already rare in the manual transcript. In addition, we do not have many of them appearing in the training corpora, compared to the frequency of the other punctuation marks. The number of periods in Table 6.9, therefore, is the same as the number of segments predicted.

Figure 6.1 presents the translation performance of the three hypotheses in BLEU over different threshold values. In this experiment as well, the same threshold value is used for all the different punctuation marks. Even though we obtain more segments the lower we set the threshold value, each hypothesis still outperforms the translation of ASR output (20.70 in BLEU). The threshold value can go down to 1 without any significant loss in BLEU. As shown by the curve of MonoTrans-FullStop, the performance is already good when having segments from periods only. When we compare MonoTrans-AllPunct and MonoTrans-FullStop, the performance of MonoTrans-AllPunct fluctuates relatively more while that of MonoTrans-FullStop stays more stagnant. From this observation we notice the necessity of another experiment where different threshold values for period and commas are used, as the performance can be improved with fewer commas when there are more segments.

Table **??** presents how close we can get toward the oracle experiments shown in 6.1 when using the segmentation and punctuation predicted output from the monolingual translation system. The numbers from an oracle experiment and ASR output are also shown for comparison. The condition Test1 represents the results where the threshold 6 was used for both period and comma.

As depicted in this table, all three hypotheses of our monolingual translation system beat the translation quality using the ASR output with a significant difference. When both segmentation and punctuation are predicted using our monolingual translation system, we gain 1.53 BLEU points on our test set, which is only 0.25 BLEU points less than a result from the oracle experiment.

In order to maintain a similar number of segments to the manual transcript, but still have the "helpful" number of commas for translation, we separate the threshold

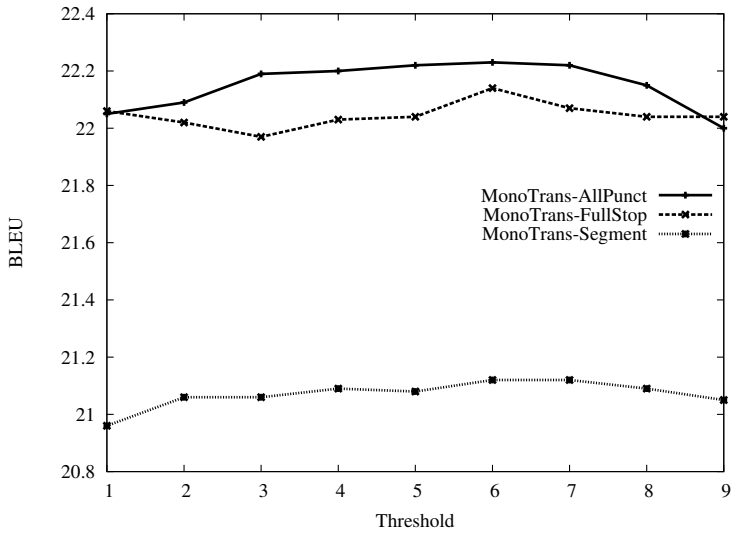

Figure 6.1: Translation performance with varying threshold values

Threshold	1	2	3	4	5	6	7	8	9	Manual Transcript
Periods	1,273	970	881	861	851	841	817	736	464	1,186
Commas	2,741	2,190	1,973	1,915	1,904	1,889	1,857	1,773	1,486	1,834

Table 6.9: The impact of threshold on punctuation marks. The number of punctuation marks predicted using the monolingual translation system with a different threshold are shown. The number of punctuation marks in the manual transcript is also given as a comparison.

value for period and comma. Test2 in Table **??** depicts the translation performance when we use the threshold value 1 for period and 6 for comma. Thus, a comma is chosen when it is found more than 5 times at the space between words. Compared to the case where the same threshold value of 6 for both punctuation marks is used, we obtain more than 150% of the original number of segments. However, we can still maintain a similar translation performance, showing only a drop of 0.06 BLEU points in the hypothesis MonoTrans-AllPunct.

Predicting a new line only after a period performs well for the translation. However, the numbers shown in Table 6.1 indicate that inserting a new line only after a period provides half of the number of segments that our ASR system produced for the test set. Therefore, to compare the performance of the ASR segmenter in a fair condition, we conduct another experiment where a new line is inserted whenever a punctuation mark, including comma, is predicted. For this experiment we use the same threshold 8 for all punctuation marks, so that we can have similar number of segments as in the ASR output. By doing so we could obtain 2,509 segments, which is nearly 200 segments more than the ASR output. From this we gained 21.67 BLEU points for the MonoTrans-AllPunct hypothesis. Although the score of the hypothesis MonoTrans-AllPunct is 0.5 BLEU points lower than previous two tests, the score is still around 1 BLEU point higher than the translation quality of raw ASR output.

6.3 Punctuation Insertion for Real-time Spoken Language Translation

The importance of inserting reliable punctuation marks and sentence segmentation into automatically recognized transcripts was emphasized in previous sections. Since many of the conventional ASR systems generate either no or only unreliable punctuation marks, many techniques have been applied for punctuation insertion task.

One indisputably crucial aspect to consider when inserting punctuation marks for real-time speech translation is the time delay. A longer context is preferred for better prediction performance but it causes more delay. The impact of longer contexts to the punctuation prediction performance is studied and shown in Appendix A. Since speed is important when presenting results by text (Mieno et al., 2015), we need to punctuate

incoming recognitions quickly for display and further processes as well, such as machine translation.

One of the commonly used real-time methods for inserting punctuation marks into the ASR output is the LM and prosody based scheme discussed in Cho et al. (2013a). It has the advantage that it incorporates acoustic features keeping the process relatively fast.

As shown in Section 6.2, a monolingual translation systems can be very effective at improving the performance of MT systems when they are applied to the ASR output. The conventional monolingual translation system suggested in Section 6.2 uses overlapping window for input. Since it uses a comparatively long context, a great performance improvement on the MT for ASR outputs can be achieved using this technique. Overlapping windows, however, make the system difficult to be used in real-time scenarios without long latencies.

Although the monolingual translation system in Section 6.2 shows a good performance in the subsequent application, adopting this system for the real-time speech translation system causes an unacceptable amount of latency due to its long shifting window of 10 words. This component alone would introduce more latency into the whole system than the desired total average latency.

In this section, we suggest an efficient punctuation insertion scheme for real-time SLT systems, using the monolingual translation system. Our punctuation insertion and sentence segmentation system is designed to take the output of a stream decoding ASR system. The input to the monolingual translation system is modified so that latency can be decreased while maintaining a similar translation performance. We performed experiments both on audio streams as well as manual transcripts, in order to give an in-depth analysis on the impact of different lengths of context in the punctuation insertion scheme.

6.3.1 Model

In order to decrease the delay in the real-time speech translation system, we use a streaming input scheme instead of the overlapping window, along with the resending ASR. In this section, we describe how the streaming input scheme works in detail.

6.3.1.1 Resending of ASR

As discussed in Section 2.1.1, one approach to reduce the apparent latency of the speech to speech translation system is to use the ASR with a resending function. In this method, the ASR components continually outputs its current best hypothesis, e.g., once a second. The hypothesis can be updated by newer, possibly better ones when more contexts are available. This approach has an advantage that it can offer higher user acceptance, since users can see the hypothesis right away.

An example excerpt shows how the resending of ASR works.

> *in this planet you would have to* ***prove*** *...*
> *in this planet you would have to* ***provide*** *36 million translation ...*

The current best hypotheses of the ASR component contains an ASR error at the verb *prove*, which is updated into *provide* based on the further recognized context.

6.3.1.2 Streaming Input

Our in-house stream decoding ASR system stores its recognition in two separate stacks. In one stack it saves its final 1-best list for words $w = \{w_l, \ldots, w_m\}$. Their following words are stored in another stack $v = \{v_{m+1}, \ldots, v_n\}$, which is not the final recognition yet. Since this stack v is flexible depending on the upcoming context, it is updated based on the context and whenever it is updated, the changes are shown to users.

In the following example, we are showing recognized words are updated using the flexible stack for a segment *"... would not exist in one hundred years why because they look at the curve and say if the population keeps growing at this rate"*. The flexible stack v is marked in a red box.

> *... would not exist in one* *hundred years* ***one***
> *... hundred years why because they look* *at the* ***curb*** *its*
> *... why because they look* *at the* ***curve*** *and say if*
> *... the curve and say if the population* *keeps growing at* ***these***
> *... keeps growing* *at* ***this*** *rate*

For this given stream of words, we can observe that occasionally words in the flexible stack v are updated when more contexts are available. In our punctuation insertion setup, we introduce another stack for recognized words before w, in order to consider

more context. The history stack h is defined as:

$$h = \{h_{l-c}, \ldots, h_{l-1}\} \tag{6.4}$$

The context c is chosen as four throughout this work. When there are fewer previous words available in the initial part of the recognition, only up to the available context is used. Thus, after the history stack h we have the finalized words w, which is followed by an updating stack v.

The newly punctuated string is then obtained by

$$w' = m(h + w + v) \tag{6.5}$$

where m denotes the monolingual translation system. Its scheme is basically same as described in Section 6.2.1. Therefore, we can obtain longer contexts by using the history stack h and the future stack v. Even though the future stack v is still unstable, it can give us an advantage that it offers more, approximately correct contexts.

Parts of the generated output are used as the final string.

$$s = \{w'_{l-c}, \ldots, w'_{m-4}\} \tag{6.6}$$

At the same time the history stack is updated.

$$h = \{w'_{m-3}, \ldots, w'_{m}\} \tag{6.7}$$

This results in us inputing punctuated text into the monolingual translation system and repunctuating it. Although this leads to a slight mismatch between the training and test data, using this approach we can guarantee that punctuation can be predicted using the longest context available.

For the non-final ASR recognition stack v, we generate the possible output string and show it to users.

The example segment in this punctuation scheme is depicted in Table 6.10. The history stack is in a yellow box, while the flexible stack is in a red box. The punctuated string to be sent to the MT module is marked in a blue box. For the first input line, we can observe that no word in the blue box is punctuated but a word in the flexible stack is. This punctuation is also shown to the users until the flexible stack is updated. For the second line, however, the punctuation module inserted a final period and a question mark around *why*.

Input	city of New York would not exist in one hundred years one
Output	city of New York would not exist in one hundred years. One
Input	not exist in one hundred years why because they look at the curb its
Output	not exist in one hundred years. Why? Because they look at the curb its

Table 6.10: Punctuation module using the streaming input

An advantage of this model is that while longer history is utilized, the decision on punctuation insertion on the current window can be made instantly, minimizing the time delay consumed on sentence segmentation. Also, by supporting the stream decoding, users can see the updated recognition as well as its most probable punctuation marks fast.

Using the overlapping input, it is required to observe the long context till its 9th next word in order to predict a potential punctuation mark after a word. This causes a structural latency in the speech to speech translation system. On the other hand, using the streaming input it is possible to use relatively longer contexts while removing the structural latency.

6.3.1.3 Phrase Table Preparation

For online translation systems, it is impossible to generate a perfectly fitting phrase table for all possible each inputs. Therefore, we build a phrase table for English to German translation, based on the vocabulary in the training data. In order to decrease the size of the model for online scenario, we first filtered out words which occurred in the corpus less than four times. Phrases that are longer than 4-grams are filtered out as well.

6.3.2 Experiments and Results

In order to measure the impact of different segmentation methods and models on MT, we experiment on the official test set of IWSLT evaluation campaign 2013. The English manual transcript of this test data has 993 sentences, or 17.8K tokens. The audio is 2h and 16m long.

The proposed streaming punctuating prediction (StreamingInput) system is compared to both a low latency baseline language model and prosody based punctuation

prediction (LM, Prosody) system as well the high latency but highly accurate monolingual translation (OverlappingInput) system using a 10 word moving window. For the LM and prosody based model we used the same segmenter described in Section 6.1.3.1. Table 6.11 presents these systems' translation performance of the test data. Not only the ASR outputs, but also the manual transcript of the corresponding talks are tested in order to give better insights of the impact from the ASR errors. All numbers are reported using case-sensitive BLEU.

Punctuation	ASR Output	Manual Transcript
LM, Prosody	9.74	-
OverlappingInput	11.18	**19.57**
StreamingInput	**11.55**	19.41

Table 6.11: Results of the punctuation scheme using streaming input. Translation performance of the proposed system is shown, compared to a fast LM, prosody based model as well as a high latency, but highly performant monolingual system using an overlapping window.

In the first row, we first show the translation performance when using the simple LM and prosody based segmentation, available only for the ASR output, as the manual transcript has no pause information. In the *OverlappingInput* system, both ASR output and manual transcript are punctuated using the conventional monolingual translation system, using overlapping windows, as suggested in Section 6.2. The shift window is applied so that each word is translated ten times. When using the *OverlappingInput* system, the phrase table is also generated upon the knowledge of the each test data as it is not for online scenario.

We can see that when we use the suggested punctuation insertion scheme, we achieve 11.55 BLEU points in the ASR test data, beating the conventional LM and prosody based model by 1.8 BLEU points. Even though this system is using relatively shorter context and the less-fitting phrase table than the traditional monolingual translation system, the translation performance is comparable with the one of the monolingual translation system with the long overlapping window.

Table 6.12 presents several segments from the ASR output, punctuated using the LM and prosody model and the suggested streaming input system. We can observe

Punctuation	Examples
LM, Prosody	I also ask myself does not really work can they really store all. This information about us and every time I use my mobile phone. So I ask my phone company Deutsche Telekom which was at that time the largest phone company. In Germany and they ask them please send me all the information you have started. About me. And there is some one thousand against and I got no real...
	... the city of New York would. Not exist in one hundred years. Why because they look at the curve and say if the population keeps growing at. This rate to move the population of New York year round they would have needed.
StreamingInput	I also ask myself, does not really work? Can they really store. All this information about us. And every time I use my mobile phone. So I ask my phone company, Deutsche Telekom, which was at that time the largest phone company in Germany, and they ask them, please send me all the information you have started about me. And there is some one thousand against, and I got no real, ...
	The city of New York would not exist in one hundred years. Why? Because they look at the curve, and say, if the population keeps growing at this rate to move the population of New York year round. They would have needed ...

Table 6.12: Segmentation improvement using the streaming input. Differences in punctuation and segmentation are shown, when using LM and prosody based model and the monolingual translation system with the streaming input.

that not only the following MT performance was improved, but the readability was greatly improved when we are using the punctuation model with the streaming input.

Due to the small model footprint and the use of an efficient MT decoder the stream-based punctuation prediction setup incurs only minimal computational cost, comparable to the punctuation model based on LM and prosody without having much future context requirements. This fast system also allows for updated punctuation when new data is received. As this component does not add further communication overhead, the total latency of the real-time speech translation system is not negatively impacted. Recent development in our framework allows outputting current best hypothesis at any time. With this, users can always access to the hypothesis with very low latency.

6.4 Summary

In this chapter, we first presented the impact of segmentation and punctuation on the output of speech recognition systems by implementing oracle experiments. Experiments have shown that we can gain up to 1.78 BLEU points of improvement on the translation quality if we apply the manual segmentation and punctuation to the ASR output. On the other hand, when we apply the segmentation and punctuation of speech recognition output to the manual transcripts, we have an overall loss of 2.45 BLEU points on the translation quality. Therefore we show that the segmentation produced by ASR systems may not assure the best translation performance, and that a separate process to segment the ASR stream before the translation can help the translation performance.

In the second part of the chapter, the monolingual translation system is used to predict segmentation and punctuation in ASR output. In order to implement this system, we change the format of the training corpora as well as the development and test set. By using the monolingual translation system, we gain more than 1.5 BLEU points on the ASR test set.

It is followed by a new punctuation insertion scheme for real-time spoken language translation system. Taking streamed input from an ASR decoder, the suggested scheme can improve the output of the speech translation without negatively impacting the speech translation system's latency. The experiments show that our low-latency real-time punctuation insertion system can achieve a comparable performance to an offline system requiring a large context window.

7

Speech Disfluency Detection

In speech, speakers occasionally talk with disfluencies such as repetitions, stuttering, or filler words. These speech disfluencies inhibit proper processing other subsequent applications, such as MT systems.

MT systems are generally trained using well-structured, cleanly written texts. The mismatch between this training data and the actual test data, in this case spontaneous speech, causes a performance drop. A system which reconstructs the non-fluent output from an ASR system into the proper form for subsequent applications will increase the performance of the application (Rao et al., 2007a).

A considerable number of works on this task such as Johnson and Charniak (2004) and Fitzgerald et al. (2009a) focus on English, from the point of view of the ASR systems. One of our goals is to extend this work to German, and also apply it to the MT task, in order to analyze the effect of speech disfluencies on MT and be able to make it applicable for lecture translator system of KIT.

In this chapter, we discuss an approach for speech disfluency detection based on conditional random fields (Cho et al., 2013b), which is a sequential modeling technique used broadly for various tasks in NLP. Later we show how this approach can be integrated into our SMT system, in order to achieve better performance in MT (Cho et al., 2014b).

7.1 Conditional Random Fields-based Approach

This section presents our disfluency detection system developed on German to improve spoken language translation performance.

Inspired by previous works Fitzgerald et al. (2009a); Liu et al. (2006), we used conditional random fields with extended features engineered for this task. In order to detect speech disfluencies considering syntactics and semantics of speech utterances, we extended this CRF-based approach using information learned from the word representation and the phrase table used for machine translation. The word representation is gained using recurrent neural networks and projected words are clustered using the k-means algorithm. The details will be given in the following sections.

Using the output from the model trained with the word representations and phrase table information, we achieve an improvement of 1.96 BLEU points on the lecture test set. By keeping or removing human-annotated disfluencies, we show an upper bound and lower bound on translation quality. In an oracle experiment we gain 3.16 BLEU points of improvement on the lecture test set, compared to the same set with all disfluencies.

7.1.1 Semantics and Disfluency Detection

Detecting obvious filler words and simple repetitions can be more feasible than other sorts of disfluencies for automatic modeling techniques, using lexical patterns such as typical filler word tokens and repetitive POS tokens as in previous work Fitzgerald et al. (2009a); Wang et al. (2010). In the Table 2.1, for example, we discussed different copy patterns in the spontaneous speech. The repetition pattern of the first sentence can be captured by analyzing the word tokens. On the other hand, the second sentence does not exhibit the exact copy pattern on their word tokens. Instead, we can detect such disfluencies easily by examining the POS patterns. Such disfluencies can be relatively less problematic to detect.

Although it is the case for obvious disfluencies (i.e. “uh”, “uhm”, same repetitive tokens, and so on) as well as limited types of disfluencies, we are confronted with many other cases where it is hard to recognize or decide whether the token is a disfluency or not via automatic means. This issue can be consistent even when the disfluency is filler words or repetitive tokens. Table 7.1 contains a sentence from the annotated data,

which exemplifies this issue for repetition. In the German source sentence, the word *üblicherweise*, meaning 'customarily' is annotated as a disfluency, as it was the speaker's intention to change the utterance into the next word *traditionell*, which means 'traditionally'. Such disfluencies are more difficult to capture than other simple repetitions, as they do not show any repetitive pattern on their surface level.

Source	Die Kommunikation zwischen Mensch und Maschine, die wir so **üblicherweise** traditionell immer sehen, ist die...
Engl. gloss	The communication between man and machine, which we **customarily** traditionally always see, is the...

Table 7.1: Difficulty in detecting repetitions

Discourse markers can be hard to capture, as they occasionally convey meaning in a sentence. In the same way as it is with English discourse markers such as "I mean", "actually", and "like", for example, German discourse markers, as shown in Table 7.2, can sometimes be used as a discourse marker and sometimes as normal tokens. In this table it is shown that a German word *nun* means 'now' as shown in the upper part, but occasionally is used as a discourse marker like in the lower part and does not need to be translated. In the lower row, the word *nun* appears with another discourse marker *ja*, which can also mean 'yes' in English, depending on the context.

Source	Sie sehen hier unseren Simultanübersetzer, der **nun** meinen Vortrag transkribiert.
Reference	Here you see our simultaneous translator, which **now** transcribes my presentation.
Source	An einer Universität haben wir **ja nun** viele Vorlesungen.
Reference	In a university, we have many lectures.

Table 7.2: Difficulty in detecting discourse markers

These examples suggest that disfluency detection requires an analysis of syntactics as well as semantics. Detecting restarted fragments especially requires semantic label-

ing, as in some cases the restarted new fragment does not contain the same content as the aborted utterances.

In this thesis we aim to analyze and improve the machine translation performance by detecting and removing the disfluencies in a preprocessing step before translation. For this we adopt a CRF-based approach, in which the characteristics of disfluencies can be modeled using various features. In order to consider the issues discussed previously, we devised features learned from word representations and phrase tables used for the MT process in addition to lexical and language model features. The MT performance of CRF-detected output is evaluated and compared to the result of an oracle experiment, where the test data without all annotated disfluencies is translated.

7.1.2 Model

We used the GRMM package (Sutton, 2006) implementation of the CRF model. We used bi-gram features, in order to model first-order dependencies between words with a disfluency. The CRF model was trained using L-BFGS, with the default parameters of the toolkit.

7.1.2.1 Training

For training and testing our CRF disfluency detection model, we use parts of the in-house German lecture data from different speakers, which is transcribed, annotated, and translated into English as introduced in Section 5.1.1.

Disfluencies are annotated manually on a word or phrase level. There are subcategories of annotation such as filler words, repetitions, deletions, partial words, and so on. These subcategories are very fine-grained, so we later re-classify them for the CRF tagging task according to our aims. Inspired by the classes defined in previous works (Fitzgerald et al., 2009a; Johnson and Charniak, 2004), we classified these annotations into three categories; `filler`, `(rough)copy`, and `non-copy`.

The disfluency classification is consistent as shown in the Section 5.1.1. Therefore, the class `filler` includes simple disfluencies such as *uhm*, *uh*, *like*, *you know* in English. If source words are discourse words or do not necessarily convey meaning and are not required for correct grammar, they are also classified as filler words. Words or phrases are grouped into `(rough)copy` when the same or similar tokens reoccur. Words are tagged as `non-copy` when the speaker changes their mind about how or what to

say. Contrary to previous work shown by Fitzgerald et al. (2009a), extreme cases of `non-copy`, in which the restarted fragments are considered to have new contexts after aborted utterances, are not excluded from the modeling target but are also taken into account.

Table 7.3 shows the detailed statistics of the annotated data used in this task, which is now a part of the data described in Section 5.1.1.

	Tokens	Percentage in the corpus
`Filler`	3,304	5.35%
`(rough)Copy`	1,518	2.46%
`Non-copy`	620	1.00%
Non-disfluency	56,264	91.18%

Table 7.3: Disfluency annotated data for CRF-based detection model

In order to make use of all annotated data and to enable cross validation, we divided the 61K words of annotated data as well as its translation in English into three parts, such that each part has around 20K words in the German source. For testing one corpus part out of three, the other two parts, which are around 40K words, are used as training data for the CRF model.

7.1.2.2 Features

The CRF-based modeling utilizes lexical, language model, word representation, and phrase table information features. Word representation and phrase table information features are devised in order to capture more syntactic and semantic characteristics of speech disfluencies.

The features are structured as followings.

- Previous and next two word/POS tokens
- Previous word/POS token with a current word/POS token[1]
- Distance to the next equal word/POS token
- Whether current word is a partial word

[1]expanded upto with previous two tokens or next two tokens

- Distance to the next word which contains the same initial letters
- Normalized word position in a sentence
- Word/POS token distance pattern
- Language model scores[1]
- RNN cluster code and pattern
- Phrase table information

Our lexical and language model features are based on the ones described in Fitzgerald et al. (2009a). We extend the language model features on words and POS tags up to 4-grams. Parser information and JC-04 Edit results as shown in Johnson and Charniak (2004) are not available in German, and therefore not used in this thesis. Furthermore, we add two new pattern features at the lexical level.

In Table 7.4, several selected features are shown for the rough repetition sentence from Table 2.1. The sentence *Da gibt es da gab es in uh gab es nur eins.* suffers from repetitions as well as using a filler word.

The 'Word/POS-Dist' feature means the distance of a token to its next appearance. Therefore, a low 'Word/POS-Dist' number indicates that this token occurs again shortly thereafter. If two or more neighboring tokens have the same 'Word/POS-Dist', the 'Word/POS-Patt' feature of the corresponding tokens is set to 1. For example, the first three tokens have the same 'POS-Dist' number, therefore their 'POS-Patt' has a value of 1. This feature enables us to efficiently detect such blocks of repetition, where the same or roughly the same words are repeated. We use a 1 of k encoding for features. Since binary features are supported better for the toolkit, we quantize the numeric features. For example, language model scores are quantized using the equal-sized bins in the log space. The POS tags are automatically generated using Schmid (1994).

With the mentioned features, we can find syntactic clues for disfluency detection. For example, POS tokens and their patterns can help to figure out repetitive `(rough)copy` occurrences. However, as discussed earlier, in the annotated data we observe that in many cases it is required to include a semantic level information as well. In addition to the mentioned features, we devised a new strategy of including word embedding features derived from an RNN and phrase table information.

[1] unigram, 4-gram, and devision of them

Source	Da	gibt	es	da	gab	es	in	uh	gab	es	nur	eins	.
Engl. gloss.		There is			there was		in	uh	there was		only	one	.
Word	Da	gibt	es	da	gab	es	in	uh	gab	es	nur	eins	.
POS	ADV	VVFIN	PPER	ADV	VVFIN	PPER	APPR	ITJ	VVFIN	PPER	ADV	PIS	$.
Word-Dist	3	365	3	47	4	4	259	9	218	821	115	933	27
POS-Dist	3	3	3	7	4	4	12	9	6	80	3	21	27
Word-Patt	0	0	0	0	1	1	0	0	0	0	0	0	0
POS-Patt	1	1	1	0	1	1	0	0	0	0	0	0	0
Annotation	-	RC	RC	RC	RC	RC	RC	FL	-	-	-	-	-

Table 7.4: Sample features on the lexical level

7.1.2.3 Word Representation using RNN

Word representations have gained a great deal of attention for various NLP tasks. Especially word representation using RNNs have been proved to be able to capture meaningful syntactic and semantic regularities efficiently (Mikolov et al., 2013b). RNNs are similar to feed-forward neural networks, but an RNN has a backwards directed loop, where the output of hidden layers becomes additional input. This allows the network to effectively capture longer history compared to other feed-forward-based n-gram models.

Word embedding is a distributed word representation, where words are represented as multi-dimensional vectors. The word vectors syntactically and semantically relating to each other will be close to each other in that representation space. Thus, words within certain semantic and syntactic relations have similar vector values. Conventionally, word embeddings of a textual corpus are obtained using certain types of neural networks.

In the hope that word representation can offer insights on semantics and syntactis, in this work we use word embedding features learned from an RNN for the CRF model. We use RNNLM (Mikolov et al., 2010) with 100 dimensions for word representations. In order to ensure an appropriate coverage of the representation, we use the preprocessed training data of the MT system, which contains various domains such as news and lectures. This data consists of 462 million tokens with 150K unique tokens.

Word Projection and Cosine Distance Figure 7.1 depicts the 2-dimensional word projection from the 100-dimensional real-valued vectors representations using the RNN, where we can observe word clusters being formed. This visualization is obtained using t-Distributed Stochastic Neighbor Embedding (Van der Maaten and Hinton, 2008). Due to memory consumption, only the most frequent 10K words are projected.

Analyzing the details of this projection, we observe that words with the same syntactic role are projected closely to each other. For example, possessive cases corresponding to 'my', 'his', and 'our' in English are projected closely to each other as shown in Figure 7.2. We can observe that in the top left corner the possessive cases such as *meinen* and *deinen* are gathered together. In the figure we can also find *welchem*, which means 'whose'. The figure also depicts that some of the prepositions such as *im*, *zum*, *am*, for example, are gathered in one spot.

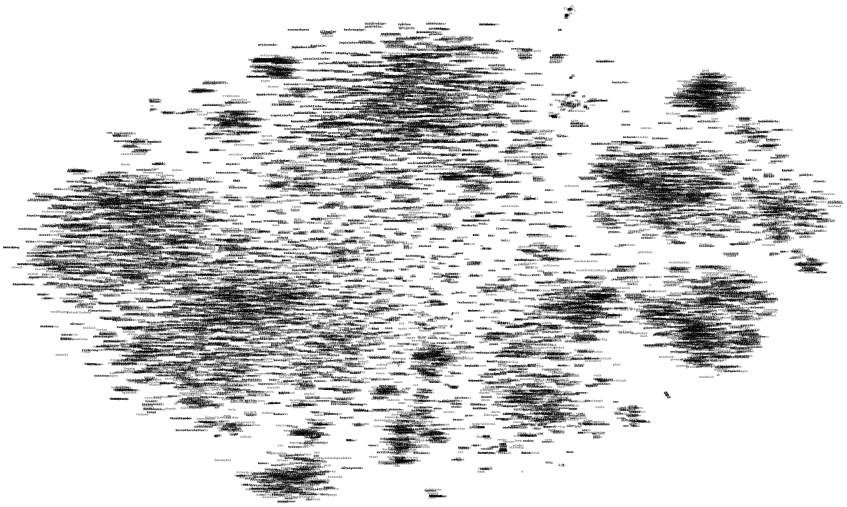

Figure 7.1: Word projection of training data, with word representation obtained with an RNN

Figure 7.2: Syntactic information encoded in the word representation

Throughout the figure, the cluster seems to be able to express morphological inflections as well. While the possessive accusative masculine cases (with *en* ending) are clustered together, all prepositions seem to have the dative ending (represented with *(e)m* ending). This is consistent with other grammatical components of a sentence, such as personal pronouns or relative pronouns. We observe clusters for dates, months and times.

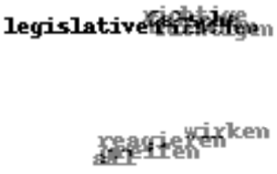

Figure 7.3: Semantic information encoded in the word representation

The projection seems to convey semantic relations to some extent. Figure 7.3 presents the semantic information encoded by using the RNNs. We can see that the

verbs *wirken*, *reagieren*, and *greifen* are projected closely. These verbs have a common meaning of 'to act/function/be effective'. In the upper part of the figure several adjectives are depicted. *legislative* means 'legislative', whereas *richtig* means 'right' or 'correct'. When it comes to adjectives, we often observed that they are projected according to their stem and occasionally also their meanings. Verbs are clustered with other verbs with the same tense or stem.

In order to compare the closeness of words numerically, we calculate their cosine similarity. Cosine similarity for two words is calculated by

$$similarity = cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} \tag{7.1}$$

where A and B are vectors of them.

Word in German	Meaning in English	Cosine Distance
schnell	fast, quick	1
rasch	quick, rapid	0.8394
bald	soon, shortly	0.6245
effektiv	effective	0.6092
zügig	efficient, speedy	0.6088
wahrscheinlich	probable	1
vermutlich	probably	0.9066
möglicherweise	maybe, possibly	0.8938
sicherlich	certainly	0.8937
vielleicht	maybe, possibly	0.8827

Table 7.5: Cosine similarity of words in word representations

Table 7.5 depicts a couple of examples. For each bold-lettered word, the four words with the highest cosine similarity are presented. Evidently, these four words are sharing a high semantic closeness with each given word, which will provide a quality feature for the task of disfluency detection. From this analysis, we conclude that RNNs can offer syntactic and semantic clues for disfluency detection.

Word Clustering In order to use the word representation vectors as features in the CRF model more efficiently, we cluster the word representations with the k-means algorithm. From preliminary experiments, the number of clusters k is chosen to be 100.

Every word of the RNN training data falls into the 100 clusters. For every word in the test data, our preprocessing system checks whether this word has been observed in the word representations. If it has been observed, the word is assigned with the corresponding cluster code as a binary feature. If it has not been observed, the cluster code 0 is assigned. Also, the distance to the next identical cluster code and the repetitive pattern of it are also used as CRF model features, as shown in Table 7.4 for word and POS tokens.

7.1.2.4 Phrase Table Information

One of the common effects of disfluencies on the MT process is that often the translation contains repetitive words or phrases. When identical tokens in the source sentence are the reason for this, the original source sentence can be corrected using lexical features. However, often we observe other cases where two words, which are different on the lexical level, generate two identical translated words. Table 7.6 depicts one example of this from our data.

Source	Diese Vorlesungen sind natürlich **jetzt inzwischen** alle abgespeichert, die liegen auf unserem Server.
Engl. gloss	These lectures are of course **now meantime** all stored, they lie on our server.
MT output	This lecture series are, of course, **now now** all stored, which lie on our server.
Reference	These lectures have of course all been saved in the meantime, they are on our server.

Table 7.6: Necessity of using phrase table information for disfluency detection

In this example, the German word *jetzt* (Engl. gloss. 'now') is annotated as a disfluency, followed by a word *inzwischen* (Engl. gloss. 'meantime', 'now'). Translating this source sentence as it is generates the translation containing two identical tokens in a row in English. We expect to solve this problem by examining the meaning of the source words in a phrase table. Thus, the target words for given source words in a phrase table are examined.

An advantage from using phrase table information is that we can detect semantic closeness of words or phrases in a source sentence independent from their syntactic roles.

As shown in Table 7.5, word representation tends to group those words together which are syntactically and semantically closely related. However, using the phrase table information, words which are only semantically related, but not necessarily syntactically related, can also be grouped together. Considering that many of the repetitions also have different POS tags in a sentence, this phrase table feature is expected to capture such disfluencies.

In order to derive this feature, we examine the bilingual language model (Niehues et al., 2011) tokens in the phrase table. The bilingual language model tokens consist of target words and their aligned source words. Using this information, we count how often a given source word is aligned to a certain target word and list the three most frequently used target words. For example, for a German word *normalerweise*, its frequently aligned target words are *normally, usually, typically, ordinarily* and *generally.* For another German word *üblicherweise*, English target words such as *traditionally, typically, usually*, and *normally* are frequently aligned to it. Therefore, by comparing the frequently aligned words in the target language, we can extract semantic relations between the two words that are far in the surface form. If the same target word(s) appears in both lists, the current word is given a phrase table feature.

An equivalent feature is introduced for the phrase level, so that we can cover the case where multiple words are translated into one or multiple word(s). As an example, we can consider consecutive source words f_1, f_2, and f_3 in one phrase. This phrase is aligned to a target token e_1. If the next source token f_4 is also aligned to the target token e_1, the first three tokens, namely f_1, f_2, and f_3, are given the phrase level phrase table feature. The coverage of the phrase level feature can be expanded up to three consecutive words as a single phrase on the source side. Thus, the source tokens f_1, f_2, and f_3 are examined as one phrase, and this can be also narrowed down to f_1 and f_2 only. The target token(s) aligned to the source phrase, consists of upto f_1, f_2, and f_3, is compared to the target token(s) aligned to the potential repetitive phrase, which can consist of also up to next three tokens f_4, f_5, and f_6. The German source words with split compounds are also considered in this way.

In our phrase table the word *inzwischen* in Table 7.6 is aligned to 'now' most frequently, followed by 'meantime' and 'meanwhile'. The most frequently appeared translation for the next appearing word *jetzt* is 'now', followed by 'currently', and

'just'. Thus, by using the phrase table features, it will be indicated that the first word *jetzt* is aligned to a same target word with its next appearing word.

7.1.3 Experiments and Results

To investigate the impact of disfluencies in speech translation quality, we conduct four experiments.

In the first experiment, the whole data, including annotated disfluencies, is passed through our SMT system. Throughout these experiments, we used the German to English SMT system described in Section 5.3.2.

For the second experiment, we remove the obvious filler words *uh* and *uhm* manually in order to study the impact of the filler words which can be captured systematically. Although there are a great number of other filler words, many of these filler words are not removed in this experiment, since they are not always disfluencies.

In the third experiment, we use the output from the CRF model without the features from words representations and phrase table information, which will be noted as CRF-Baseline. The one trained with features from word representations and phrase table information will be noted as CRF-Extended. If the CRF models detect a token as either of the three classes, `filler`, `(rough)copy`, or `non-copy`, the word token is assumed to be a disfluency and is removed. The three classes are trained in the same model together. As mentioned previously, training and testing the CRF model is done with three-fold cross-validation. Thus, both of the CRF models are trained on around 40K annotated words, and tested on around 20K annotated words. The performance is evaluated on the joined three sub-test sets.

In the last experiment, all disfluency-annotated words are removed manually. As all annotation marks are generated manually, this experiment shows as an oracle experiment the maximum possible improvement we could achieve.

All experiments are conducted on manually transcribed texts, in order to disambiguate the effects from errors of an ASR system. The experiments considers all available data, which is 61K words, or 3K sentences.

7.1.3.1 Results

Table 7.7 depicts the results of our experiments. The scores are reported as case-sensitive BLEU scores, including punctuation marks.

System	BLEU
Baseline	19.98
+ no *uh*	21.28
CRF-Baseline	21.92
CRF-Extended	21.94
Oracle	23.14

Table 7.7: Impact of disfluency removal using the CRF-based model, prior to translation. We can observe the influence of disfluency in speech translation.

The result of the first experiment is presented as the Baseline system, where all disfluencies are kept in the source text. When we remove all *uh*s and *uhm*s in the source text manually, we gain 1.3 BLEU points.

Apart from this, we use the output of the CRF-Extended as an input to our machine translation system. Words tagged as disfluencies are all removed. The translation score using the CRF-Extended is almost 2 BLEU points better than translating the text with all disfluencies. Compared to the second experiment where we remove *uh* and *uhm*, the performance is improved by around 0.7 BLEU points. The improvement by using the extended features in the CRF model was not captured by the BLEU, yielding only a minimal difference. An in-depth analysis of the impact of the two systems will be given in the following chapter.

7.1.3.2 Analysis

The detection results for all models are given in Table 7.8. In total, there are 5,432 speech disfluencies annotated by human annotators, and among them, 3,012 speech disfluencies are detected by CRF-Extended.

Compared to the case where the obvious filler words are removed, 1,025 more speech disfluencies are detected and removed. Compared to CRF-Baseline, where the features obtained from the word representations and phrase table information are not used, 103 more disfluencies are detected using CRF-Extended, while also a higher number of tokens are falsely detected.

In order to analyze the difference between the translations produced by CRF-Baseline and CRF-Extended, we score the two test sets resulted from each of the

System	Correct	Wrong
Baseline	0	0
+ no *uh*	1,987	0
CRF-Baseline	2,909	489
CRF-Extended	3,012	552
Oracle	5,432	0

Table 7.8: Performance of disfluency detection in accuracy

CRF model sentence by sentence and rank them according to their difference in BLEU scores. Differences appear in 223 sentences.

One notable difference is that the CRF-Extended system detects a higher number of repetitions. Table 7.9 shows a sentence from the test set, where a longer phrase of repetition is captured using CRF-Extended. Words which represent a disfluency are marked in bold letters. Both systems can catch the obvious filler word *uh* and the simple repetition *als als*. In addition to this detection, the CRF-Extended system captures the whole disfluency region, in spite of the considerably complicated sentence structure and repetitive patterns. In this sentence the repeated words appear with varying frequencies and with a different distance to the next identical token. In order to detect such disfluencies, the correct phrase boundary needs to be recognized. As a result of this detection, the MT output using the CRF-Extended system is much more fluent than the one using the CRF-Baseline system.

Table 7.10 shows a sentence from the test set, where the CRF-Extended system does not perform better than the CRF-Baseline system for the given reference. The only disfluency shown in the original sentence *der*, marked with bold letters, is removed using both techniques. The CRF-Extended system additionally detects *einen Umschwung* as a disfluency. However, this deletion harms neither the structure nor meaning of the sentence, as *einen Umschwung* means 'a turnaround', or 'a change', which conveys practically the same meaning as the next following tokens.

It is an interesting point that using the semantic features we could detect that *einen Umschwung* is semantically closely related with *eine veränderte*, despite their distance in tokens and different syntactic roles in the sentence. This is an example that even though the CRF-Extended output does not match the human-generated annotation in this case, the CRF-Extended still provides a good criteria to detect semantically related

Source	Man kann das natürlich sowohl **als Links- als auch als** als Links- als auch als Rechtshänder **uh** verwenden.
Engl. gloss	You can this of course both **as left- as also as** as left- as also as right-handed **uh** use.
CRF-Baseline	Man kann das natürlich sowohl **als Links- als auch** als Links- als auch als Rechtshänder verwenden.
MT output	You can use this, of course, both as a left- as well as on the left- as well as a right-handed.
CRF-Extended	Man kann das natürlich sowohl als Links- als auch als Rechtshänder verwenden.
MT output	You can use this, of course, both as a left- as well as a right-handed.
Reference	You can of course use this as left- as well as also as a right-handed person.

Table 7.9: Disfluency detected using the CRF-based model with semantic features. Syntactically complicated, long phrase with a disfluency is captured using CRF-Extended.

words.

The CRF-Extended system also performs better with regard to distinguishing between discourse markers and the normal usages of the words. 59% of difference in correctly classified disfluencies between the CRF-Baseline and CRF-Extended stems from filler words. The rest is achieved from detecting a higher number of correct repetitions.

Source	Die Ausrufung des totalen Kriegs markierte eigentlich *einen Umschwung*, **der** *eine veränderte* Form der Politik.
Engl. gloss	The proclamation of total war marked actually *a turnaround*, **of** *a change* form of politics.
CRF-Baseline	Die Ausrufung des totalen Kriegs markierte eigentlich *einen Umschwung, eine veränderte* Form der Politik.
MT output	The proclamation of the total war was collared actually *a turnaround, a changed* form of politics.
CRF-Extended	Die Ausrufung des totalen Kriegs markierte eigentlich *eine veränderte* Form der Politik.
MT output	The proclamation of the total war was collared actually *a changed* form of politics.
Reference	The call for total war in fact marked *a turnaround*, and *a changed* form of politics.

Table 7.10: Semantically related words detected using the CRF-based model with semantic features (CRF-Extended)

7.2 Integration into an SMT System

In previous section, we discussed how speech disfluencies can be detected using a sequential tagging model. Such disfluency detection systems deploy a hard decision, which can have a negative influence on subsequent applications such as machine translation. In this section we show a novel approach in which disfluency detection is integrated into the translation process.

We train a CRF model to obtain a disfluency probability for each word. The SMT decoder will then skip the potentially disfluent word based on its disfluency probability. Using the suggested scheme, the translation score of both the manual transcript and ASR output is improved by around 0.35 BLEU points compared to the CRF hard decision system.

7.2.1 Motivation

One of the advantages of detecting and removing speech disfluencies is to increase the accuracy of recognition and sequentially read the readability. Previous works shown by Johnson and Charniak (2004) and Fitzgerald et al. (2009a) focus on this aspect, for example. Other works, such as Wang et al. (2010) and Cho et al. (2013b), extend the

point of view and aim to improve the following application systems, such as machine translation.

The approaches suggested by the previous works have a potential drawback, that the decision whether a token is a disfluency or not is a hard decision. For an MT system, especially, this can pose a severe problem if the removed token was not in fact a disfluency and should have been kept for the correct translation. Therefore, we pass the decision whether a word is part of a disfluency or not on to the translation system, so that we can use the additional knowledge available in the translation system to make a more reliable decision. In order to limit the complexity, the search space is pruned prior to decoding and represented in a word lattice.

In this section, we show a novel scheme where the disfluency removal process is integrated into an MT system. Unlike previous works, our work is not limited to the preprocessing step of MT, instead we use the translation model to detect and remove disfluencies. Contrary to other systems where detection is limited on manual transcripts only, our system shows translation performance improvements on the ASR output as well.

7.2.1.1 Word Lattices in NLP

While ASR systems use lattices to encode hypotheses, lattices have been used for MT systems with various purposes. Rottmann and Vogel (2007) constructed a lattice, which contains all word reorderings according to the reordering rules learned from the POS. This is later extended to cover long-range reorderings in Niehues and Kolss (2009), as well as to include tree-based word reordering in Herrmann et al. (2013). Lattices have also been used as a segmentation tactic for compound words (Dyer, 2009), where the segmentation is encoded as input in the lattice. The authors use a maximum entropy model for the segmentation and encode it in the lattice as input into an MT system.

7.2.2 Tight Integration using Lattices

In this section, we explain how the disfluency removal is integrated into the MT process.

7.2.2.1 Model

The conventional translation of texts from spontaneous speech can be formulated as

$$\hat{e} = \arg\max_{e} p(e|\arg\max_{f_c} p(f_c|f)) \tag{7.2}$$

with

$$p(f_c|f) = \prod_{i=1}^{I} p(c_i|f_i) \tag{7.3}$$

where f_c denotes the clean string

$$f_c = \{f_i \mid c_i = \texttt{clean}\} \tag{7.4}$$

for the disfluency decision class $c \in \{\text{clean, disfluent}\}$ of each token. f_i is ordered according to the word occurrence in the sentence. Thus, using the conventional models, disfluency removal is applied to the original, potentially noisy string in order to obtain the cleaned string first. This clean string is then translated.

The potential drawback of a conventional speech translation system is caused by the rough estimation in Equation 7.2, as disfluency removal is not depending on maximizing the translation quality itself. For example, we can consider an exemplary sentence *Use what you build, build what you use.* Due to its repetitive pattern in words and structure, often the first clause is detected as a disfluency using automatic means. To deal with the issue, we can change the scheme how the clean string is chosen as following,

$$\hat{e} = \arg\max_{e}(p(e|f_c) \cdot p(f_c|f)) \tag{7.5}$$

This way a clean string which maximizes the translation quality is chosen. Thus, in this scheme no instant decision is made whether a token is a disfluency or not. The disfluency probability of the token, however, will be taken into the process of MT, by taking the log linear combination of the probabilities as shown in Equation 7.5.

For this task as well, we use a CRF (Lafferty et al., 2001) model to obtain the disfluency probability of each token.

Since there are two possible classes for each token, the number of possible clean sentences is exponential with regard to the sentence length. Thus, we restrict the search space by representing only the most probable clean source sentences in a word lattice.

7.2.2.2 CRF Model Training

In order to build the CRF model, we used the open source toolkit CRF++ (Kudoh, 2007). The unigram features for CRF modeling are same as the ones used in Section 7.1.2.2. The disfluency classes are also following the ones in Section 7.1.2.1.

For training and testing the CRF model, we use 61k annotated words of parts of the manual transcripts of university lectures in German as shown in Section 7.3. Thus, this scheme is directly comparable to the baseline model shown in Section 7.1. For developing and testing the MT system, the same data is used along with its English reference translation. In the same way in the Section 7.1, we split the data into three parts and perform three-fold cross validation. Therefore, the train/development data consists of around 40k words, or 2k sentences, while the test data consists of around 20k words, or 1k sentences.

7.2.2.3 Lattice Implementation

We construct a word lattice which encodes long-range reordering variants (Niehues and Kolss, 2009; Rottmann and Vogel, 2007). For translation we extend this so that potentially disfluent words can be skipped.

Let us consider an example sentence.

Das sind die Vorteile, die sie uh die sie haben.

(En.gls: *These are the advantages, that you uh that you have.*)

This sentence experiences repetition *die sie* as well as filler word *uh*. Its reordering lattice is shown in Figure 7.4, where words representing a disfluency are marked in bold letters. In this sentence, the part *die sie uh* was manually annotated as a disfluency, due to repetition and usage of a filler word.

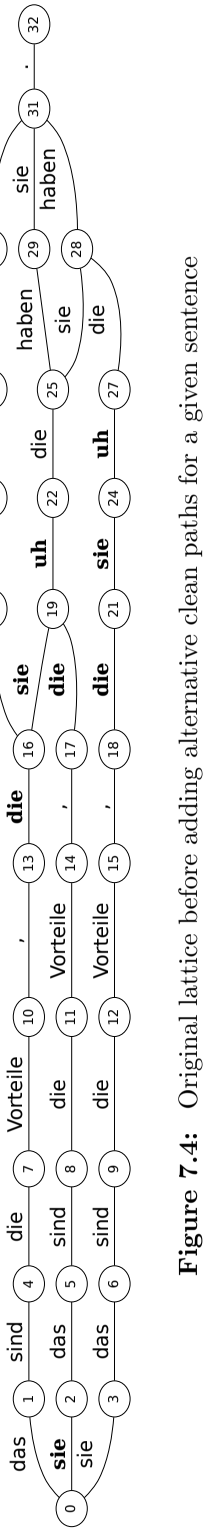

Figure 7.4: Original lattice before adding alternative clean paths for a given sentence

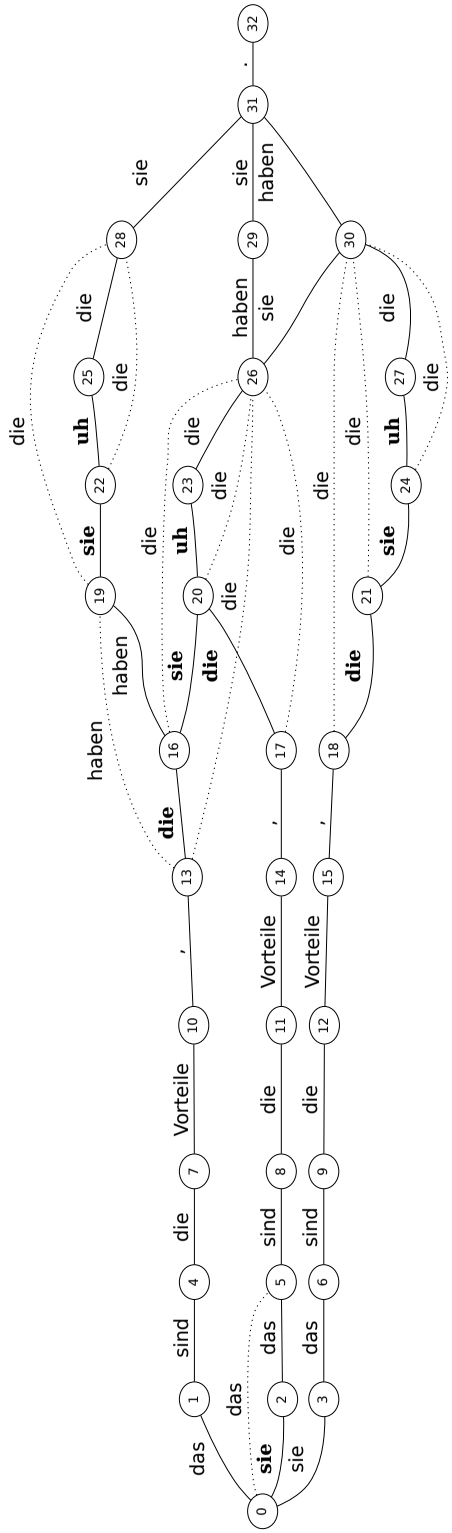

Figure 7.5: Extended lattice with alternative clean paths for an exemplary sentence

The disfluency probability P_d of each token is calculated as the sum of probabilities of each class.

$$P_d = P_{FL} + P_{RC} + P_{NC} \tag{7.6}$$

Table 7.11 shows the disfluency probability P_d obtained from the CRF model for each token. As expected, the words *die sie uh* obtain a high P_d from the CRF model.

das	0.000732	**sie**	**0.953126**
sind	0.004445	**uh**	**0.999579**
die	0.013451	die	0.029010
Vorteile	0.008183	sie	0.001426
,	0.035408	haben	0.000108
die	**0.651642**	.	0.000033

Table 7.11: Disfluency probability of each word

In order to provide an option to avoid translating a disfluent word, a new edge which skips the word is introduced into the lattice when the word has a higher P_d than a threshold θ. During decoding the importance of this newly introduced edge is optimized by weights based on the disfluency probability and transition probability.

The extended lattice for the given sentence with $\theta = 0.5$ is shown in Figure 7.5, with alternative paths marked by a dotted line. We can observe that compared to the original lattice, the new lattice contains a lot of edges. The previously discussed word with a disfluency *sie* now has an edge skipping over it, connecting node 0 and 5. From node 22 to 28, now we have an option of taking the edge *die*, instead of going over the node 25, which would have generated the path *uh die*.

Search Space On selecting the candidate words to jump over, we conduct experiments with the altered search space by changing θ. The optimal value of θ was manually tuned on the development set. Apart from the search space, we also set restrictions in order to avoid memory and time consumption. For example, when P_d is smaller than 0.2, the corresponding word does not get an edge jumping over it. Also, when the next word of the current skipping word has P_d higher than 0.9, the expansion covers directly the second next edge.

For the testing, we choose the scaling factor from the best performing optimization.

Weights of Edge Each edge of a lattice has a probability. Every edge of an original lattice has the weight. When a new edge is introduced, the weight on this new edge is obtained by getting product of the weight of two edges. For example, in Figure 7.5, the weight on the edge between the node 0 and 5 is obtained by multiplying the weight on the edge between the node 0 and 2 and the edge between the node 2 and 5.

In addition to this weight that each edge originally has, we use another weight to encode the probability of keeping the corresponding word. The weight on this new edge is product of the weight of the edge between 0 and 2 and the next edge between 2 and 6. Therefore, this second weight of the edge between the node 0 and 5 in Figure 7.5 is obtained by multiplying P_d of the word *sie* and P_k of the word *das*.

7.2.3 Experiments

In order to compare the effect of the tight integration with other disfluency removal strategies, we conduct different experiments on manual transcripts as well as on the ASR output. All translations are generated using the system introduced in Section 5.3.2. While the system uses same translation and language models, it is re-optimized using our three-fold training data scheme as discussed in Section 7.2.2.2.

7.2.3.1 Manual Transcripts

As a baseline for manual transcripts, we use the whole uncleaned data for development and test. For "No *uh*", we remove the obvious filler words *uh* and *uhm* manually. In the CRF-hard experiment, the token is removed if the label output of the CRF model is a disfluency class. This scheme is based on the CRF-based model described in Section 7.1. The fourth experiment uses the tight integration scheme, where new source paths which jump over the potentially noisy words are inserted based on the disfluency probabilities assigned by the CRF model. In the next experiments, this method is combined with other aforementioned approaches. First, we apply the tight integration scheme after we remove all obvious filler words. In the next experiment, we first remove all words whose P_d is higher than 0.9 as early pruning and then apply the tight integration scheme. In a final experiment, we conduct an oracle experiment, where all words annotated as a disfluency are removed.

7.2.3.2 ASR Output

The same experiments are applied to the ASR output. Since the ASR output does not contain reliable punctuation marks, there is a mismatch between the training data of the CRF model, which is manual transcripts with all punctuation marks, and the test data. Thus, we insert punctuation marks and augment sentence boundaries in the ASR output using the monolingual translation system as introduced in Chapter 6. As the sentence boundaries differ from the reference translation, we use the Levenshtein minimum edit distance algorithm (Matusov et al., 2005) to align hypothesis for evaluation. No optimization is conducted, but the scaling factors obtained when using the corresponding setup of manual transcripts are used for testing.

7.2.3.3 Results and Analysis

Table 7.12 shows the results of our experiments. The scores are reported in case-sensitive BLEU.

System	Dev	Text	ASR
Baseline	23.45	22.70	14.50
No *uh*	25.09	24.04	15.10
CRF-hard	25.32	24.50	15.15
Tight int.	25.30	24.59	15.19
No *uh* + Tight int.	25.41	24.68	15.33
Pruning + Tight int.	25.38	**24.84**	**15.51**
Oracle	25.57	24.87	-

Table 7.12: Results of the tight integration of a disfluency detection model into SMT. Translation results for the investigated disfluency removal strategies are presented.

Compared to the baseline where all disfluencies are kept, the translation quality is improved by 1.34 BLEU points for manual transcripts by simply removing all obvious filler words. When we take the output of the CRF as a hard decision, the performance is further improved by 0.46 BLEU points. This system and CRF-Extendend in Table 7.7 are in the same condition, using the same method. The score difference is from using different development data, due to three-fold system using this integration scheme.

When using the tight integration scheme, we improve the translation quality around 0.1 BLEU points compared to the CRF-hard decision. The performance is further improved by removing *uh* and *uhm* before applying the tight integration scheme. Finally the best score is achieved by using the early pruning coupled with the tight integration scheme. The translation score is 0.34 BLEU points higher than the CRF-hard decision. This score is only 0.03 BLEU points less than the oracle case, without all disfluencies. One explanation for this improvement can be that the removing of the words with a high probability simplifies the task of selecting the remaining disfluent words and therefore the log-linear model.

Experiments on the ASR output also showed a considerable improvement despite word errors and consequently decreased accuracy of the CRF detection. Compared to using only the CRF-hard decision, using the coupled approach improved the performance by 0.36 BLEU points, which is 1.0 BLEU point higher than the baseline.

System	**Precision**	**Recall**
CRF-hard	0.898	0.544
Pruning + Tight int.	0.937	0.521

Table 7.13: Detection performance comparison

Table 7.13 shows a comparison of the disfluency detection performance on word tokens. While recall is slightly worse for the coupled approach, precision is improved by 4% over the hard decision, indicating that the tight integration scheme decides more accurately. Since deletions made by a hard decision can not be recovered and losing a meaningful word on the source side can be very critical, we believe that precision is more important for this task. Consequently we retain more words on the source side with the tight integration scheme, but the numbers of word tokens on the translated target side are similar. The translation model is able to leave out unnecessary words during translation.

7.3 Summary

In this chapter, we presented a CRF-based disfluency detection technique with extended features from word representations and a phrase table. These features are designed to

capture deeper semantic aspects of the tokens. Using the predicted results from the CRF model, we gain around 2 BLEU points on manual transcripts of lectures. From the detailed analysis, we show that usage of the extended features provides a good means to detect semantically related disfluencies. The oracle experiment suggests that the machine translation of spontaneous speech can be improved significantly by detecting more disfluencies correctly.

Later on we presented a novel scheme to integrate this disfluency removal system based on CRF model into the MT process. Using this scheme, it is possible to consider disfluency probabilities during decoding and therefore to choose words which can lead to better translation performance. The disfluency probability of each token is obtained from a CRF model, and is encoded in the word lattice. Additional edges are added in the word lattice, to bypass the words potentially representing speech disfluencies.

We achieve the best performance using the tight integration method coupled with early pruning. This method yields an improvement of 2.1 BLEU points for manual transcripts and 1.0 BLEU point improvement over the baseline for ASR output.

Although the translation of ASR output is improved using the suggested scheme, there is still room to improve.

8

Modeling Punctuation and Disfluency for Multi-Party Meeting Data

Multi-party meeting data is a speech resource where another degree of spontaneousness can be observed compared to previously discussed university lecture data. Meetings involve multiple participants, which increase speech disfluencies such as interruptions drastically. Translating such meetings, therefore, presents a big challenge.

Previous research to deal with disfluency and lack of punctuation in multi-party meetings was focused on using prosody (Baron et al., 2002; Shriberg et al., 2001). Another approach was using multi-stage classifiers to detect disfluencies (Mieskes and Strube, 2008).

In this chapter, we investigate the importance of transforming speech transcripts of multi-part meetings into well-written input, prior to the translation process. Therefore, our first goal is to improve machine translation performance of it. For this transformation, we modeled punctuation prediction and speech disfluencies.

As shown in previous chapters, both tasks are essential to improve machine translation quality of speech. In this chapter, we explore two different ways of modeling the two tasks, cascaded model (Cho et al., 2014c) and the joint model (Cho et al., 2015a), motivated by their different advantages. In the cascaded model, where two tasks are applied sequentially, we can fully use all available data for the punctuation prediction. In the joint model, on the other hand, there is an advantage that the two tasks can be

handled in a single process.

8.1 Cascaded Model based on Conditional Random Fields

In the cascaded model, disfluency removal and punctuation prediction are trained separately and applied to the test data one by one. This scheme is motivated by the amount of data available for each model. Speech disfluencies are normally modeled based on human-generated annotation on the speech data. This process is therefore very expensive and time consuming. The data resource for modeling punctuation and segmentation, however, is more abundant. Monolingual data with proper punctuation marks can be a useful data for this task. In order to make use of the big data resource for punctuation modeling, we investigate the performance of the cascaded system.

In the cascaded model, different schemes of punctuation insertion model are applied to the test set after its disfluencies are detected by the CRF-based disfluency removal system as described in Section 7.1. Disfluencies are removed by a CRF model trained on in-domain and out-of-domain data. By doing so, we are going to explore the genre-portability of this task. Sentence segmentation and punctuation are performed in three different ways and their performance is compared. The first method is based on a language model. As described in Section 6.3, this method is one of the most frequently used methods for real-time segmentation due to its fast processing speed. The second criterion is based on turn information and the third one is the monolingual translation system as described in Section 6.2.

For comparison, we build a joint CRF model for punctuation insertion and disfluency removal. By applying these models, multi-party meetings are transformed into fluent input for machine translation.

We evaluate the models with regard to translation performance and are able to achieve an improvement of 2.1 to 4.9 BLEU points depending on the availability of turn information.

8.1.1 System Architecture

In this work, we chose a work scheme where the output stream from an ASR system passes first through an automatic disfluency detection system. Based on this cleaned-up stream, punctuation and segmentation insertion is performed. Once the disfluencies

in the ASR output are removed and punctuation marks are inserted, the cleaned, punctuated data goes through the MT system like normal input data.

For the disfluency removal model, we use data of two different domains: multi-party meeting and lecture. The multi-party meeting data is split into train and test data as shown in Table 5.7. For training, we use five meeting sessions, which sum up to $38.6k$ annotated words. In order to model the case where we have no in-domain data, we train the second model using lecture data. We use web-based seminar lecture data given in English as well as parts of the annotated English reference translation of the German lecture data shown in Section 5.1.1. The $41.8k$ tokens of web-based seminar lecture data is obtained within the project EU-BRIDGE internally. The lecture data contains altogether $104k$ annotated words, and shows a moderate level of disfluency.

Once the models are built, they are applied to the remaining three meeting sessions. The test data consists of $2.1k$ segments with $14.9k$ English words and $11.4k$ French words. After cleaning up the disfluencies manually, the source side contains $11.7k$ English words.

8.1.1.1 Turn Information

For MT of multi-party meetings, turn information can play a big role, since knowing who spoke when can provide basic segmentation. However, turn information is not always available. For example, a good diarization system can be missing in small group meeting sessions.

In order to compare and study the impact of turn information on our models, we assume two scenarios: in the first scenario turn information is available while in the second one it is not available. With the turn information, basic segment information according to speaker changes is available. Even though this may not be the exact sentence segmentation, it can offer a reasonable baseline for segmentation and punctuation insertion. It can also offer additional features for disfluency detection. As it is possible to know which segment is started by which speaker, we can obtain a cue that the previous segments' last tokens could have been interrupted by the new speaker, given the fact that meetings contain a lot of interruptions.

When the turn information is not available, there is no basic segmentation. Therefore it is required to chunk the stream of ASR output into segments. Different tactics on segmentation and punctuation insertion will be described in Section 8.1.3.

8.1.2 Disfluency Detection

Speech disfluencies in the multi-party meeting data is modeled as a sequence labeling task. For the task, we use the conditional random fields-based model as introduced in Section 7.1. Same as for the lecture data, we used the GRMM package (Sutton, 2006). Disfluency classes follow the meeting data description in Section 5.1.2.2.

8.1.2.1 In-domain vs. Out-of-domain Data

In the ideal case, disfluency annotated in-domain data is available for training the CRF model. However, the annotation of speech for different domains can be very time-consuming. As in-house disfluency annotated lecture data is available, we use this data as our out-of-domain training data for the CRF model. As in-domain training data we use the in-house English meeting data. This will show whether the disfluency removal model is portable across different domains.

Compared to the meeting data, lecture data has different characteristics. Although it still provides general speech disfluencies such as repetitions or filler words, lecture data in general contains a moderate level of speech disfluencies compared to the quite noisy meeting data. Especially, unlike meeting data, lecture data does not contain interruptions by other speakers. Therefore, for testing the CRF model using lecture data, we mapped `interruption` onto the `non-copy` class.

8.1.2.2 Features

In order to capture speech disfluencies, we use the features introduced in Section 7.1.2.2. Word vectors for each word is obtained using Mikolov et al. (2013a), due to its efficiency in training.

As mentioned earlier, we assumed two scenarios about turn information availability. In the scenario where the turn information is available, we extracted the word position within the turn. We expect that disfluencies can be more prominent in the initial part of each turn, because many stutters as well as corrections occur within the first several words. In addition, as interruptions between speakers occur at end of each turn, we encoded whether the current token is one of the first or final 5 words of the turn in order to incorporate this information for the training.

The CRF model is trained with a bigram feature, so that first-order dependencies between words with a disfluency can be modeled.

8.1.3 Segmentation and Punctuation Insertion

After removing disfluencies, the main difference between written text and the disfluency-removed speech is the lack of punctuation marks. In recent work Peitz et al. (2011), it has been shown that a promising approach to translate unpunctuated text is to automatically insert punctuation marks and segmentation prior to translation. Therefore, we analyzed three different methods to segment and punctuate the multi-party meeting data: simple LM-based segmentation, turn segmentation, and monolingual translation system.

8.1.3.1 Simple LM-based Segmentation

Assuming there is no information about different speakers and their turns available, ASR of such a talk would generate a stream of words. For translation, it is necessary to segment the stream of words. As a baseline system, we segmented based on a hard threshold of word-based LM scores. First we concatenated the test data into a single line without any punctuation marks, in order to mimic the ASR output. We use a 4-gram LM trained on the punctuated English side of the MT training corpus in Section 5.3.4 and measure the probability of a final period given the previous words. When the probability exceeded an empirically chosen threshold, we inserted a final period and started a new segment. The output of this baseline system consists of segments where each segment ends with a final period.

8.1.3.2 Turn Segmentation

If we have access to turn information, we can exploit this information in order to obtain a better baseline segmentation. We inserted a final period and began a new segment whenever the speaker changed. Each segment of this system may contain more than one actual sentence, with no further punctuation marks within the segment.

8.1.3.3 Monolingual Translation System

In Chapter 6, we showed that a monolingual translation system can be used successfully for inserting punctuation marks into non-punctuated German lecture data. Following this approach, we built a monolingual translation system from non-punctuated English to punctuated English. While the previous two methods insert only final periods, this system can insert all punctuation marks appeared in the training data. As training data we used the English side of the MT training corpus. This MT training corpus is ideally segmented and contains all punctuation marks, including a final period at the end of each sentence. In order to learn where segment breaks should be inserted, we throw away the segmentation and randomly cut the English side of the data. Aiming to generate data that is similar to the test data, we limit the length of segments to 22 words. The test data goes through the monolingual translation system with a sliding window of 10 words.

For the scenario where turn information is available, we build an additional, slightly different monolingual translation system. When we have the turn information, several segments uttered by a speaker are concatenated. Therefore, in order to make the training data similar to the test data, we concatenated one to three sentences randomly into one sentence. Punctuation marks between sentences are removed, and only a final period is added at the end of each line of the source side data. The target side contains all punctuation marks.

8.1.4 Experiments

In this section, the results of the oracle experiments are given, followed by results of different punctuation insertion and disfluency detection techniques.

In the oracle experiments, human-generated segmentation and punctuation is inserted into the test data. Also, disfluencies are removed according to the manual annotation. The oracle experiments, therefore, will give us an insight on the upper bound of this experiment. As described earlier, for all experiments we are applying two different scenarios, depending on whether the turn information is available.

In the following section, we present the impact of the different segmentation and punctuation approaches in machine translation. For these experiments, we control the disfluency condition into two cases. In the first case, all disfluencies are kept and in

the another case all manually annotated disfluencies are removed. By doing so, we can evaluate the impact of the segmentation and punctuation approaches distinguished from the disfluencies.

After that, the results of disfluency removal are analyzed. Here, by building the models using either in-domain or out-of-domain data only, we investigate the genre portability of the disfluency modeling task. For the disfluency models, we use two segmentation and punctuation schemes, the monolingual translation system and the oracle punctuation.

Finally, the overview of our system is given in the end. The performance of each technique is measured by translating the multi-party meeting data into French. All translations are generated by using the En-Fr system described in Section 5.3.4.

8.1.4.1 Oracle Experiments

Table 8.1 shows the translation performance for oracle punctuation marks and oracle disfluency removal on the multi-party meeting data.

System	No turns	Turns
Baseline	9.53	12.93
Oracle segmentation	13.96	
Oracle punctuation	15.64	
Oracle disfluency	12.21	15.72
Oracle all	20.93	

Table 8.1: Translation performance of oracle experiments for multi-party meeting data. Punctuation and speech disfluency are conditioned separately and also jointly.

In the first system, all disfluencies are kept and baseline segmentations are used. As the baseline segments, we use two different segmentation methods. When there is no turn information available, segmentation and final periods are inserted using the simple LM-based method as described in Section 8.1.3.1. On the other hand, when we have access to the turn information, a new segment and a final period are inserted whenever the speaker changes as described in Section 8.1.3.2. We can observe that using the turn information is very helpful in achieving better performance.

Then we insert oracle segmentation and a final period at the end of segment. When we also inserted all other punctuation marks from the reference transcript, the translation performance is improved up to 15.64 BLEU points even though it still contains all disfluencies. We can observe that nearly 1.7 BLEU points are achieved by inserting all other punctuation marks, on top of we have the ideal reference segmentation and a final period.

In the next experiment, we keep the punctuation and segmentation the same as in the baseline system, but remove all of the manually annotated disfluencies. By doing so, translation performance is improved by around 3 BLEU points compared to the baseline system. Finally, we achieved a BLEU score of 20.93 when we have the oracle for both punctuation and disfluency. This is the upper bound of the performance we can get for this test set when we have both perfect segmentation/punctuation and disfluency removal.

As shown by these numbers, the performance can be improved by more than 10 BLEU points if the ideal punctuation and disfluency detection are applied. Therefore, modeling these two problems in a translation system of multi-speaker speech is essential to reach a good translation quality.

8.1.4.2 Segmentation and Punctuation Insertion

In this section, we look into the performance of the segmentation and punctuation in a realistic approach (all disfluencies kept) and perfect conditions (remove all disfluencies using the manual annotation). The experiments are conducted for the two cases, depending on the turn-information availability.

System	Keep disfluency	Oracle disfluency
Baseline	9.53	12.21
Monolingual translation system	12.44	16.34
Oracle punctuation	15.64	20.93

Table 8.2: The impact of segmentation and punctuation on translation performance when no turn information is available

Table 8.2 shows the results under the assumption that no turn information is available. The baseline system has punctuation and segmentation inserted using the sim-

ple LM-based method. When punctuation marks are inserted using the monolingual translation system, we achieved an improvement of 3 to 4 BLEU points for both disfluency conditions. This improvement reaches almost half of the difference between the baseline systems and oracle scores. We can also observe that when segmentation and punctuation are improved, the impact of disfluencies increases. There is bigger room of improvement which can be achieved by removing correct disfluencies, when we have better segmentation and punctuation. The same phenomena can be observed in the experiments with turn information, as shown in Table 8.3.

System	Keep disfluency	Oracle disfluency
Baseline	12.93	15.72
Monolingual translation system	13.25	17.71
Oracle punctuation	15.64	20.93

Table 8.3: The impact of segmentation and punctuation on translation performance when turn information is available

We can observe that the baseline scores in this case have already improved a lot over the experiments without turn information. Since the baseline segmentation is already better, the improvements are smaller, but there are still consistent improvements when inserting punctuation marks using the monolingual translation system. It is shown that when a better disfluency modeling technique is available, our segmentation modeling technique can also show a better performance, emphasizing the importance of the accuracy of the disfluency detection model.

8.1.4.3 Disfluency Removal

This section presents translation performance when we apply the disfluency removal models trained either on in-domain or out-of-domain data. Punctuation and segmentation are inserted not only by the monolingual translation system for the realistic case, but also oracle punctuation is used for comparison.

Table 8.4 shows the scores under the assumption that there is no turn information available. In the first experiment, we keep all disfluencies. Then we show the scores when we use the disfluency removal model trained only on the in-domain data, multiparty meeting data. These scores are compared with the scores when we use the

System	Monolingual translation system	Oracle punctuation
Keep disfluency	12.44	15.64
CRF in-domain	14.41	17.26
CRF out-of-domain	14.24	16.95
Oracle disfluency	16.34	20.93

Table 8.4: The impact of disfluency removal on translation performance when no turn information is available

model trained only on the out-of-domain data, which is lecture data. Finally, we show the scores removing all disfluencies annotated. An interesting point is that using lecture data for training the CRF model yields similar performance to training using the meeting data. Even though using the lecture data is slightly worse than using the meeting data, the difference is minimal.

Our preliminary experiments showed that when we use the in-domain data for training the disfluency removal model, we have around 8 points better F-scores, compared to the case when we train the model using out-of-domain data. However, such differences are not pronounced in terms of BLEU. It shows that the disfluency modeling technique shown in this work can be transfered into a new domain without causing a big loss of performance in MT.

System	Monolingual translation system	Oracle punctuation
Keep disfluency	13.25	15.64
CRF in-domain	15.01	17.10
CRF out-of-domain	14.90	17.03
Oracle disfluency	16.34	20.93

Table 8.5: The impact of disfluency removal on translation performance when turn information is available

This result is also observable when the models are trained with turn information, as shown in Table 8.5. The disfluency removal model trained on meeting data performs only slightly better than the lecture data. In all listed conditions, it is shown that we can improve the translation quality by 1.5 to 2 BLEU points by removing disfluencies.

8.1.4.4 Combined Modeling of Punctuation Insertion and Disfluency Removal

As an additional experiment, we model punctuation marks and disfluencies in one model. This way of modeling has an advantage that it is not necessary for ASR output to pass through two different steps. We also hope that this experiment can provide the first insight on MT performance when modeling these two in one model for the given task. In this scheme, both the punctuation marks as well as disfluencies are predicted given the potentially disfluent, and unpunctuated ASR output. For modeling we use the same features as for the disfluency removal as in Section 8.1.2.2.

Punctuation and disfluencies are trained using the data with speech disfluencies. For the modeling, we use the same CRF tool, but with two decision labels: one with disfluency classes and another one with punctuation marks.

System	No turn	Turn
Baseline	9.53	12.93
Combined CRF in-domain	13.92	14.45
CRF in-domain + Monolingual translation system	14.41	15.01
Combined CRF out-of-domain	13.99	14.58
CRF out-of-domain + Monolingual translation system	14.24	14.90
Oracle all	20.93	

Table 8.6: Punctuation insertion and disfluency removal in one CRF model

Table 8.6 presents translation performance when using one CRF model for both punctuation and disfluency. The CRF model is built again under the two genre-matching conditions: in-domain and out-of-domain. For comparison, we also give the number when punctuation and disfluency are modeled separately using the monolingual translation system and a CRF model respectively. In all experiments, we apply two baseline segmentation conditions depending on the turn information availability.

When modeling punctuation marks and disfluency removal together in one model, it still provides a big improvement over the baseline, where all disfluencies are kept. Same as in the previous experiments, training the models on in-domain or out-of-domain data does not cause a big performance difference in MT. Comparing the scores of training the models separately for disfluencies and punctuation marks, however, the

scores are generally around 0.3 to 0.5 BLEU points worse. The F-score of disfluency removal does not get affected significantly even when we are modeling it along with punctuation marks. However, as the monolingual translation system is trained using much more data, the performance of segmentation and punctuation insertion is affected and therefore degrades the overall performance.

8.1.4.5 Overview

Finally, Table 8.7 shows the best scores achieved in this work.

System	No turn	Turn
Baseline	9.53	12.93
Best system	14.41	15.01
Oracle	20.93	

Table 8.7: Overview of the translation performance improvement when using the cascaded approach

The baseline system corresponds to the same systems in Table 8.1, where no punctuation or disfluency schemes are applied. In our best system we first remove disfluencies using a CRF model trained on the in-domain data, and then insert proper segmentation and punctuation marks using the monolingual translation system. When there is no turn information, we achieve around 4.9 BLEU points of improvement. With turn information, we improve the system by around 2.1 BLEU points. In the oracle condition, we have 20.93 BLEU points, showing a considerable difference from the best performance we could achieve.

One explanation of this difference can trace back to the exceptionally high rate of speech disfluencies in the multi-party meeting data. Comparing Table 5.7 and Table 5.5, we can observe that the overall disfluency rate of the multi-party meeting data is significantly higher than the one of the lecture data. In the multi-party meeting data, the disfluency rate is around 19.2%, while the one of the lecture data is around 12.5%.

8.2 Joint Model based on the Combination of Conditional Random Fields and Neural Networks

As shown in the previous section, inserting proper punctuation marks and deleting speech disfluencies can improve the following machine translation performance greatly. This challenging task has prompted extensive research using various techniques, such as conditional random fields. Neural networks, however, are relatively under-explored for this task.

Combining different modeling techniques with different advantages has the potential to lead to improvements. In this chapter, we first establish the performance of joint modeling of punctuation prediction and disfluency detection using neural networks. We then combine a conditional random fields based model and a neural networks based model log-linearly, and show that the combined approach outperforms both individual models, by 2.7% and 3.5% in F-score for speech disfluency and punctuation detection, respectively. When used as a preprocessing step to machine translation this also results in an improved translation quality of 2.5 BLEU points compared to the baseline and of 0.6 BLEU points compared to the non-combined model.

8.2.1 Motivation

Modeling punctuation marks and speech disfluency together can result in positive synergistic effects. Punctuation prediction, for example, can heavily depend on the existence of a speech disfluency, since disfluencies are good predictors for punctuation marks. Also, information regarding sentence boundaries can be beneficial for disfluency detection.

CRF and NN have been used extensively for various NLP tasks, showing different advantages. CRFs are successfully used in sequence labeling tasks, detection task, due to their ability to model first order dependencies. For example, in Chapter 7, we showed that speech disfluencies can be modeled effectively using the CRFs. They were applied to the same task of multi-party meeting data in the cascaded model in Section 8.1. They were devoted to model punctuation marks for the international evaluation campaign, which will be described in Section 10.2.4.3. Even though the model is built on much smaller data, it offered a comparable performance to the monolingual translation model

which is built on all available data. NNs have proven themselves to be very useful at classification tasks and are therefore a sound choice for NLP tasks of this nature.

The different strengths and weaknesses of both CRF models and NN models suggest that they can complement each other when jointly applied to task of detecting punctuation and disfluencies in spontaneous speech. Despite the potential advantages they can offer when applied together, combining the two modeling techniques for punctuation and disfluency detection has not been investigated yet. Although RNNs can offer the context information that feed-forward neural networks (FFNN) can not easily provide, training RNNs bears a disadvantage of an expensive computation. In this work, therefore, we aim to explore the potential of using FFNNs combined with other ML frameworks for their synergistic effects.

In this chapter, we present a punctuation and disfluency detection scheme using a combination of both CRF and NN models. We propose a multi-tasking learning NN, which is designed to exploit the above mentioned synergistic effects by jointly modeling both punctuation and disfluencies in a single network with multiple parallel output layers. One output layer is devoted to detecting speech disfluencies while the other output layer is concerned with predicting punctuation marks. The CRF also models punctuation and disfluency detection using two output labels, where the first label covers disfluency and the second one punctuation marks. The predictions of the models are extracted in probabilities and used as features in a log-linear combination.

8.2.2 Model

In this section we describe the two modeling techniques for generating probabilities of punctuation and disfluency as well as the features they use.

Our features for the models include lexical and language model features, as well as word cluster and phrase table features as introduced in Section 7.1.2.2. The same set of features is used for both CRF and NN training in order to make them comparable.

Disfluency classes are following the study in Section 5.1.2.2. The class `FL` covers filler words and discourse markers, while identical and rough copies in the class `RC`. Restart fragments and aborted sentences are categorized in the class `NC`. Finally, the interrupted segments are grouped in the class `IR`. The tokens without any disfluency are given the class `clean`.

Punctuation marks are grouped into four classes. As their names suggest, `Comma` takes commas and `QuMark` covers question marks. Due to its rare occurrence, exclamation marks are gathered together with final periods and grouped into `Period`. Words that are not followed by any punctuation marks are assigned with `none`.

8.2.2.1 Conditional Random Fields

In this work, the GRMM package (Sutton, 2006) is used for the linear chain CRF model. As there are two output labels for each token, one for disfluency and another for punctuation, we use one linear chain edge across disfluency labels, another one across punctuation labels, and another for the in-between edges. The model is trained using L-BFGS, with default parameters.

8.2.2.2 Neural Networks

Due to its shorter training time we use a five layer FFNN in this work. It is trained to jointly predict both the punctuation and disfluency labels. As can be seen in Figure 8.1 the input consists of a 907 dimensional feature vector encoding the features described in Section 8.2.2, followed by three hidden layers containing 500 neurons each, and two parallel output layers. The hidden layers use the sigmoid activation function and the output layers use the softmax activation function. The parallel output layers are devised for the joint detection of disfluency and punctuation marks. Each output layer is considered to be a separate softmax group which results in the network generating a separate probability distribution for punctuation and disfluency labels.

The network is pretrained layer-wise using denoising auto-encoders (Vincent et al., 2010) which enable us to also make use of the 400K unannotated examples as well as the 140k labeled examples. After pretraining the network is fine-tuned using mini-batch gradient decent. The learning rate is updated according to the newbob schedule, where it remains constant until improvements between epochs on the a cross-validation set drop below a threshold, after which the learning rate is decreased exponentially. Training is terminated when improvements between epochs on the a cross-validation set drop below a threshold again. Pretraining and fine-tuning were implemented using *Theano* (Bergstra et al., 2010).

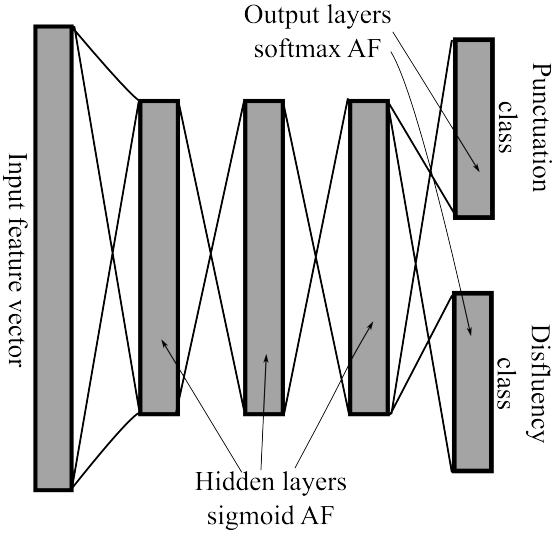

Figure 8.1: Proposed joint punctuation and disfluency prediction neural network.

Unlike CRFs, feed-forward NN only base their label predictions on the provided features which can lead to predictions that contradict each other. This problem can be overcome by integrating the NN with an LM into a decoder.

8.2.3 Log-linear Combination

In MT, combining different translation models and LMs log-linearly in a decoder can greatly improve the translation quality (Och and Ney, 2002). The individual models are encoded as separate features and weighted using interpolation coefficients optimized on a validation set. Inspired by this, we combine our CRF based punctuation and disfluency prediction with our NN based one log-linearly using a label LM. For this task, we perform the search for the best label sequence as well as the optimization of the log-linear weights using an MT decoder (Vogel, 2003). The models are optimized on BLEU.

For a given word sequence $w = w_1 \dots w_n$ we wish to find the best sequence of punctuation and disfluency labels $(p, d) = (p_1, d_1) \dots (p_n, d_n)$:

$$\underset{(p,d)}{\operatorname{argmax}} \sum_{i=1}^{m} \lambda_i \cdot f_i(w, p, d) \tag{8.1}$$

where m is the number of features and

$$p_i \in \{\texttt{Period}, \texttt{Comma}, \texttt{QuMark}, \texttt{none}\}$$
$$d_i \in \{\texttt{filler}, \texttt{rough-copy}, \texttt{non-copy}, \texttt{interruption}, \texttt{clean}\}.$$

following the classes described in Section 5.1.2.

We define input features from the two models ($M \in \{CRF, NN\}$) for each of the punctuation labels by:

$$f_{\hat{p}}^{M} = \sum_{j=1}^{n} \delta_{\hat{p},p_j} \cdot \log P_M(p_j|w) \tag{8.2}$$

and for each of the disfluency labels by:

$$f_{\hat{d}}^{M} = \sum_{j=1}^{n} \delta_{\hat{d},d_j} \cdot \log P_M(d_j|w) \tag{8.3}$$

The final input feature is derived from a 9-gram LM trained on the output labels of the training data. The LM is built using the SRILM Toolkit (Stolcke).

This formulation of the problem not only allows us to find the optimal label sequence, it can also be easily extended to incorporate further models.

8.2.4 Experiments and Results

In this section we present the results using F-score and BLEU for translation of the test data into French.

8.2.4.1 Results

Our first experiment measures the quality of disfluency detection and punctuation insertion using precision, recall and the standard F-score metric. The scores presented in Table 8.8 measure whether a word was labeled as one of the disfluency classes or not. The results of the individual CRF and NN models, which are found in the first two rows of the table, show that the CRF model detects more disfluencies and therefore has a better recall performance. On the other hand, the NN model outperforms it on precision leading to fewer false detections. Their log-linear combination improves the F-score by 2.7% and seems to strike a balance between precision and recall.

System	F-score	Precision	Recall
CRF	53.90	68.83	44.29
NN	49.31	81.08	35.43
Log-linear combination	56.56	72.77	46.26

Table 8.8: Results of the disfluency detection. Performance is measured in F-score for all systems.

Similarly, the evaluation of our models' punctuation prediction capabilities show that while the NN model is the most precise at detecting punctuation marks, it is more conservative, and therefore has as a lower recall, than the CRF model. As can be seen in Table 8.9, we achieve our best performance on both F-score and recall when the models are combined. Both metrics are noticeably improved by the combination, F-score by 3.5% and recall by 5.5%.

System	F-score	Precision	Recall
CRF	58.22	60.23	56.34
NN	52.82	65.31	44.35
Log-linear combination	61.76	61.64	61.87

Table 8.9: Results of the punctuation prediction. Performance of the CRF, NN and combined systems for punctuation prediction are measured in F-score.

As an additional experiment, we measured the effect of multi-task (disfluency and punctuation) learning. In this experiment, we build two separate CRF models and two NN models. For each technique, one model is dedicated for disfluency and another for punctuation. In this way, we can measure the impact of modeling two speech phenomena jointly. Shown in Table 8.10, the results show that the multi-task learning does not bring a great difference to the detection accuracy itself.

System	Joint-disf	Joint-punc	Sep-disf	Sep-punc
CRF	53.90	58.22	53.12	57.06
NN	49.31	52.82	50.45	52.16
Log-linear combination	56.56	61.76	56.34	62.05

Table 8.10: Evaluation of the multi-task learning.

The purpose of using LM in this architecture is to provide more context information

that the NN model might miss. From our preliminary experiment, it was shown that the disfluency detection using NN, as a single-task learning, benefits from the extra context information. For this task, the NN model for disfluency detection achieved 50.45 F-score, as shown in Table 8.10. Adding an LM on this configuration improved the F-score up to 55.33. We applied the same experiment for the joint detection for disfluency and punctuation using the NN model. The result is shown in Table 8.11. The numbers prove that using an additional LM can provide the missing context information of the NN model.

System	F-score-disf	F-score-punc
NN	49.31	52.82
+ LM	53.68	56.78

Table 8.11: Effectiveness of using an LM for the FFNN model

In order to evaluate not only the raw detection accuracy, but also its impact on an MT system, we use the punctuation-predicted, disfluency-removed test data as input data for the MT system described in Section 5.3.4. The MT system translates the test data into French. It is then evaluated against a human translation of the oracle text where all annotated disfluent words were removed and reference punctuation marks are inserted.

Table 8.12 shows a comparison of translation quality using the various punctuation and disfluency prediction methods presented in this work. In order to ensure a fair comparison, we use consistent segments for translation and evaluation in all tests; they span all tokens between speaker changes. In the baseline system, all disfluent words are kept and only just prior to the speaker change is a single sentence ending period inserted. The test data generated by our systems and the reference may also contain punctuation marks within these segments. A trivial rule-based disfluency removal system that only removes simple filler words such as *uh* or *uhm* is also listed in order to demonstrate the additional capabilities of our models.

Removing disfluent words and inserting punctuation marks using only the CRF model improves the translation quality of the baseline system by 1.90 BLEU points. With a BLEU score of 16.32 this approach also compares favorably to our trivial system, beating it by around 1.38 BLEU points. The NN model achieves a BLEU score of 16.18,

System	BLEU
Baseline	14.42
+ No *uh*	14.94
CRF	16.32
NN	16.18
Log-linear combination	16.93
Oracle	22.76

Table 8.12: Translation performance using the combined model. Translation scores after disfluency removal and punctuation insertion using various systems are shown in BLEU.

which still outperforms the trivial rule-based system by 1.24 BLEU points. Using both models in a decoder results in our best score of 16.93 BLEU. The oracle score shows the upper bound of this experiment.

8.2.5 Analysis

In this section, in-depth analysis on disfluency and punctuation detection performance is given, comparing the individual models and the combined model. The performance comparison is given for each class of disfluency and punctuation depending on the technique used. Analysis shows that the improvement in punctuation and disfluency detection has a positive impact on readability.

8.2.5.1 Readability

Two segments from our test data, showing the synergistic effect of the log-linear combination, are presented in Table 8.13. The raw input contains a repetition, marked in bold letters, and is missing proper punctuation marks. In the manually cleaned version of this excerpt, the repeated part is removed and punctuation marks are inserted, which makes it notably easier to understand. The CRF model was able to successfully detect the repetition in the first segment. In the second segment however, it deletes too much, leading to the ungrammatical sentence *"for what are these recordings for"*. This false labeling of disfluencies is probably due to the repetitive nature of that segment. The CRF also fails to insert any sentence boundaries. Although the NN based model was unable to remove the repetitive part in the first segment, it correctly detected the sentence boundary after the first segment.

Raw input	**do you use do you have** digits as a class what are these for what are these recordings for
Manually cleaned	Do you have digits as a class? What are these for? What are these recordings for?
CRF	do you have digits as a class *for what are these recordings for*
NN	**do you use do you have** digits as a class. What are these for what these recordings for
Log-linear combination	do you have digits as a class? What are these for? What are these recordings for

Table 8.13: Synergistic effect of the combined model. Excerpt from the test data showing that the CRF and NN models can complement themselves is presented.

Using the combined model we were able to remove the speech disfluency detected by the CRF model while at the same time inserting the correct sentence boundaries. This sequence generated using the combined model shows that even when the two separated models perform imperfectly, we can benefit from their synergistic effects. It is also notable that while the location of the sentence boundaries was correctly predicted by the NN it predicted the wrong punctuation class. In the model combination though both the location of the sentence boundaries and the fact that they were question marks and not periods were correct. These effects are observable throughout the test data. It suggests that combining the models provides an opportunity to optimize on relative importance of the features.

Another impressive outcome of the combined model is that it can improve readability even in cases where it does not match the human annotation. An example segment that demonstrates this point is given in Table 8.14, where a very disfluent segment is cleaned and punctuated using different techniques. Speech disfluencies according to the annotators are marked in bold letters. Although the result of the combined model does not match disfluency and punctuation marks of the annotation, thereby lowering the F-score, its readability is comparable to the annotated sentence.

8.2.5.2 Synergistic Effect

As shown in Table 8.8 and 8.9, using the combined model we can achieve improved F-scores both on disfluency and punctuation detection.

Raw input	yeah **you** you mean this okay **right right** good **yeah** it's an at sign
Manually cleaned	Yeah. You mean this. Okay. Good. It's an at sign.
CRF	yeah, this okay **right, right** good **yeah,** it's an at sign.
NN	yeah, you mean this okay, **right, right** good, **yeah,** it's an at sign.
Log-linear combination	yeah, this okay, **right?** Good, **yeah,** it's an at sign.

Table 8.14: Improved readability using the combined model. Excerpt demonstrating the improved readability of the combined model despite a prediction that is very dissimilar to manually cleaned text.

Speech Disfluency Detection Detailed performance in detecting different disfluency classes using the CRF model is given in Table 8.15. For simplicity, we are going to notate `filler` class as `FL`, `rough-copy` as `RC`, `non-copy` as `NC`, and `interruption` as `IR` in this section. It is clearly observable, that detecting disfluency classes as `NC` and `IR` is a much harder task compared to detecting other classes. While the model can classify 60.1% and 48.7% of filler words and (rough) repetitions into their exact class correctly, detecting other classes is possible less than 10%. The disfluency removal rate can be further higher, as even when a word is categorized into a different disfluency class, the word is removed from the transcript. Thus, in this test set 63.4% of filler words and 53.8% of (rough) repetitions are removed.

Hyp \ Ref	FL	RC	NC	IR	clean
FL	600	14	16	37	190
RC	14	495	50	19	253
NC	8	22	16	14	63
IR	11	16	2	86	137
clean	366	470	246	704	11,006

Table 8.15: Disfluency detection performance using CRF for different classes

This phenomenon is observed more strongly in the experiments using only NNs.

Table 8.16 demonstrates that disfluency detection based on neural networks is more conservative compared to the one based on CRFs, which also fits with our findings in the previous section. While it can detect slightly more filler words into the correct class, the number of overall detection ratio itself is much lower than the CRF model. Especially the rare occurrence of non-copy and interruption tokens in the training data, which is less than 1.8% and 0.9% respectively, becomes the driving source that the NN based model maintains the high precision.

Hyp \ Ref	FL	RC	NC	IR	clean
FL	621	13	13	35	159
RC	5	415	20	14	106
NC	0	0	0	0	0
IR	0	0	0	0	0
clean	373	589	297	811	11,384

Table 8.16: Disfluency detection performance using NN for different classes

Table 8.17 shows the synergistic effects achieved when combining the two models. Compared to the two individual models, the combined model can detect a notably higher number of FL and RC disfluencies. While the CRF based model falsely detected and removed 200 clean tokens into disfluency class NC and IR, this model makes the false detection into the two classes only on 26 tokens.

Hyp \ Ref	FL	RC	NC	IR	clean
FL	674	25	23	53	266
RC	8	568	52	43	263
NC	0	1	0	0	0
IR	0	2	1	33	26
clean	317	421	254	731	11,094

Table 8.17: Disfluency detection performance using the combined model for different classes

The comparison on number of speech disfluencies detected and missed is given in Table 8.18. When a word is classified as one of the disfluencies and this word

is in fact annotated as one of the speech disfluency classes, it is counted as *Deleted disfluency*. A large portion of them is actually classified into the exact disfluency class, which is counted in *Deleted disfluency (correct)*. In the criterion *Missing disfluency* the disfluency tokens which are classified as clean tokens are concerned, in which the disfluencies are removed correctly. In all three criteria the combined model outperforms other two individual models, which proves the synergistic effects.

	Deleted disf.	Deleted disf. (corr)	Missing disf.
CRF	1,420	1,197	1,786
NN	1,136	1,036	2,070
Log-linear Combination	1,483	1,275	1,723

Table 8.18: Performance comparison of different techniques for disfluency detection. The comparison on number of speech disfluencies detected and missed using different techniques is given.

Punctuation and Segmentation Insertion The detailed performance of punctuation and segmentation insertion using the CRF model only is given in Table 8.19. Among 1,811 sentence boundaries in the reference, we could detect 1,048 correctly. For other 283 sentence boundaries, commas are inserted instead. The detection rate for commas is slightly lower, down to 31.5%. Out of 963 commas in the references, 93 of them are inserted as sentence boundaries. While it misses 1,047 punctuation marks to detect, the model inserted 576 of false punctuation marks.

Hyp \ Ref	`PC`	`CM`	`null`
`PC`	1,048	93	239
`CM`	283	303	277
`null`	480	567	11,565

Table 8.19: Punctuation prediction performance for each class using CRF. Detailed punctuation and segmentation insertion performance is given for each class using the CRF model.

Same as for speech disfluencies, the detection using the neural network based model is more conservative and it inserted fewer number of punctuation marks. In Table 8.20,

it can be observed that the detection rate for final period and comma is down to 45.9% and 27.3% respectively. At the same time, however, the number of false punctuation marks is also decreased to 274 tokens.

Hyp \ Ref	PC	CM	null
PC	831	9	117
CM	298	263	157
null	682	691	11,807

Table 8.20: Punctuation prediction performance for each class using NN. Detailed punctuation and segmentation insertion performance is given for each class using the NN model.

By using the combined model, 66.5% of sentence boundaries are detected correctly as shown in Table 8.21. The summary of punctuation and segmentation detection using different detection techniques is given in Table 8.22. Being able to insert the highest number of correct punctuation marks and minimize the missing punctuation marks, the combined model shows its sound performance.

Hyp \ Ref	PC	CM	null
PC	1,204	149	324
CM	246	268	197
null	361	546	11,560

Table 8.21: Punctuation prediction performance for each class using the combined model. Detailed punctuation and segmentation insertion performance is given for each class using the combined model.

	Correct punctuation	Missing punctuation
CRF	1,351	1,047
NN	1,094	1,373
Log-linear Combination	1,472	907

Table 8.22: Performance comparison of different techniques for punctuation prediction. The comparison on number of punctuation marks correctly inserted and missed using different techniques is given.

We also investigated the number of commas, sentence boundaries and kept word tokens of each test set, shown in Table 8.23. Note that the number of segments may differ from the number of sentence segmentations inserted by the model. The numbers reported in Table 8.21 are the number of punctuation marks detected using the model. It is often the case, however, that such punctuation marks surround a word that is classified into a disfluency using the same model. In this case, we remove the word that is classified into a disfluency and its following punctuation mark.

System	SB	Comma	Word
Baseline	1,255	0	14,855
+ No *uh*	1,158	0	14,445
CRF	1,083	827	12,792
NN	828	684	13,454
Log-linear combination	1,340	691	12,860
Oracle	1,236	772	11,832

Table 8.23: Test data statistics before/after the prediction process

In Table 8.23, both the baseline system and the trivial rule based system do not contain any commas, and only whenever the speaker changes do they contain a period. Over 400 simple filler words are removed, by applying the trivial rule based system. Compared to the NN system, the CRF system inserts more punctuation marks as well as deleting more words for disfluency.

8.3 Summary

In this chapter, we showed how machine translation performance is affected when different techniques for segmentation, punctuation insertion and disfluency removal are applied to multi-speaker speech. The characteristics and differences of multi-speaker speech compared to other data were described.

Punctuation insertion and disfluency removal are modeled in two approaches: cascaded modeling and the joint modeling. In the cascaded model, first the disfluency removal is applied and then punctuation and segmentation are inserted. For the disfluencies, we built two separate disfluency removal systems using in-domain and out-of-domain data and their performances are compared in terms of translation quality. We

showed that our disfluency removal technique presented in this work can be transfered to a new domain. Depending on the availability of turn information, two scenarios are modeled for the segmentation and punctuation.

The best system of disfluency removal and punctuation detection models achieves a gain of 4.9 BLEU points when there is no turn information and 2.1 BLEU points when turn information is available over the baseline. As an additional experiment, a sequence tagging model which models both segmentation, punctuation insertion and disfluency removal is built and the performance is compared to our best automatic systems.

In the joint modeling, we build a combined model which detects speech disfluencies and predicts punctuation marks jointly. We showed that multiple models with complementary advantages can be combined in order to improve the performance of joint disfluency and punctuation labeling. We present both conditional random fields based and neural networks based models and explain how they can be combined in a log-linear decoder, in order to achieve better performance. Both models and their combination are tested on conventional meeting data and intrinsically evaluated with F-score as well as extrinsically by using them as a precursory step to an MT system.

The results demonstrate that our combination outperforms the two individual models on both F-score and BLEU. Compared to the best single model it boosts the disfluency detection F-score by 2.7% and the punctuation prediction F-score by 3.5%. While both the CRF and NN models improve the translation quality of the baseline system by 1.90 and 1.76 BLEU points respectively, the combined approach gives us an improvement of 2.51 BLEU points.

An analysis of our proposed model indicates that these improvements stem from synergies between the models. We go on to show that it can also noticeably increase the readability of the spoken language input even when the model's output does not conform to the human annotation.

9

Reconstruction of Spoken-style Sentences

In previous chapters, we discussed the two big differences between spoken language and written language and showed how the lack of proper punctuation marks and the existence of speech disfluencies can be remedied by using different techniques to achieve better quality of machine translation.

Apart from those two challenges, spoken language also contains colloquial expressions and ungrammatical phrases. The necessity of building more coherent and grammatical sentences is emphasized in Fitzgerald and Jelinek (2008), while paraphrasing or changing the styles of the text has been discussed in Neubig et al. (2012); Quirk et al. (2004); Xu et al. (2012).

Inspired by this, we analyze the difference between spoken and written language in its structure and word usage. Based on the analysis, we transform the spoken-style sentences into more formal sentences. Several previous approaches (Quirk et al., 2004; Xu et al., 2012) relied on using a machine translation system. Using this technique, however, reasonable performance can be achieved when we have a considerable amount of parallel data. While Neubig et al. (2012) investigated this problem using weighted finite state tranducers on a large quantity of data. In this task, we investigate sentence reconstruction of German lecture data. In order to overcome the data sparsity, our strategy is automatically model the differences between formal language and speech-style sentences separately step by step.

9.1 Motivation

Altering a sentences' style into a different one has attracted the attention of several researches due to the various applications. In Lee and Seneff (2006), a stylistic correction was deployed in order to check the grammar of non-native English speakers. The transformation of certain dialects into a standard language (Al-Gaphari and Al-Yadoumi, 2012) has been also investigated from this perspective. In this work, we explore sentence reconstruction as a stylistic correction for spontaneous speech. We aim to give helpful insights for developing further applications related to natural language understanding and paragraphsing, by providing an automatic means of reconstruction.

In spontaneous speech, it is often the case that sentences still contain flaws even after oracle disfluency removal. The sentences suffer from less grammatical structure, or the usage of colloquial words. As these are not necessarily disfluencies, they are not annotated as such and are not removed using techniques introduced in the previous chapters. Such differences, however, still can pose an issue when applying subsequent NLP applications built using a well-written text data. Since the well-written, formal sentences are often preferred in certain domains, such as news or politics, we hope that the analysis and the models developed in this work can be valuable for further research.

As described in section 5.1.1.3, two steps of annotation were applied to the data. In the first step, all disfluencies are removed. In the following second step, further corrections were made. For a more formal format, unnecessary or colloquial expressions are removed. New words are introduced when they can improve the fluency. Word reordering is also allowed and some expressions are replaced with formal ones.

One German example sentence from our lecture data is shown in Table 9.1. The sentence's gloss translation in English and its proper reference translation are also given.

In this example, a sentence is depicted, where both the modal verb *kann* and the reflexive verb *sich ändern* are in wrong positions. In German, the modal verb (*kann*) and the reflexive pronoun (*sich*) must be located in the second position in the main clause. Afterwards the verb itself (*ändern*) should be located at the end of the main clause. In the reconstructed version, we can observe that this part is corrected.

Sentence reconstruction is not limited to reordering, but can be also expanded to word/phrase replacement. Table 9.2 shows another sentence excerpted from our German lecture data. In this sentence, four words, that are used colloquially, are

Disfluency removed	Im Gegensatz dazu, bei dem Mealy-Automaten, die Ausgabe **kann sich ändern** bei einer Zustandsänderung, ...
English gloss	In contrary, with the Mealy-automaton, the output **can itself vary** with a status change, ...
Reconstructed	Im Gegensatz dazu **kann sich** bei dem Mealy-Automaten, die Ausgabe bei einer Zustandsänderung **ändern**, ...
English gloss	In contray, **can itself** with the Mealy-automaton, the output with a status change **vary**, ...
Reference	In contrary, the output can vary when a status changes with the Mealy-automaton, ...

Table 9.1: Example of a sentence requiring verb reordering

replaced with different words. *mal* is replaced by *einmal*, *was* is replaced once with *etwas* and one with *das*. This gives us another insight; replacements are not always homogeneous. There is also reordering involved in this sentence. A phrase *auf den Seiten* is reordered to be located in front of *'rum*, which is again replaced with *herum*.

Disfluency removed	Surfen sie **mal** ein bisschen **'rum auf den Seiten**, die ich Ihnen gegeben habe, vielleicht fällt Ihnen **was** auf, **was** sie gerne machen wollen.
English gloss	Search you **once** a little **aruond on the websites**, that I you given have, maybe like you **something** on, **something** you gladly do would like to.
Reconstructed	Surfen sie **einmal** ein bisschen **auf den Seiten herum**, die ich Ihnen gegeben habe, vielleicht fällt Ihnen **etwas** auf, **das** sie gerne machen wollen.
English gloss	Search you **once** a little **on the websites around**, that I you given have, maybe like you **something** on, **that** you gladly do would like to.
Reference	Search the websites I gave you a little, maybe you will find something which you would like to do.

Table 9.2: Example of a sentence requiring word replacement

9.2 System Architecture

In this thesis, we suggest the initial approach to build the written styled, formal-speech styled sentences from a given set of spontaneous German lecture data.

In this work, we restrict the scope of the problem to deletion and replacement of words and phrases. We notice that sometimes words, which are not necessarily speech disfluencies, and are therefore not removed by the annotators in the first step of annotation, are deleted in the reconstruction annotation. The words which are most frequently removed by the human annotators are modeled using a binary maximum entropy classifier. The second step is replacement. In this step, words or phrases that can be replaced with more formal expression are handled. Since we have a problem of sparse features and small data, we build an artificial data out of the big EPPS data. The patterns of replacements in the annotated data are learned and applied back to the EPPS data according to their frequency.

9.2.1 Deletion

In order to obtain the statistics, we align the disfluency-removed data to the manually reconstructed data using GIZA++ (Och and Ney, 2003). The manually annotated

5,4k sentences are split into 4k training sentences and 1,4k test sentences. Based on the alignment, we get statistics on the deletion, in both train and test data as shown in Table 9.3. The numbers in this table are given in tokens.

We can observe that a relatively few number of words are deleted from train and test data. Only 1.45% and 1.34% of all words occur in the train and test data are deleted by the annotators, respectively. Based on this, we get a list of the most frequently deleted words. The most frequently deleted words are as followings: *","*, *"die"*, *"das"*, *"dann"*, *"wir"*, *"ist"*, *"da"*, *"und"*, *"der"*, *"so"*, ..., where most of them are articles and conjunctions.

	Train	Test
Number of words	79,032	32,747
Number of deleted words	1,148	438
Number of deleted unique words	285	181

Table 9.3: Statistics in annotation of deletion

9.2.1.1 Maximum Entropy-based Model

ME modeling has been used extensively in different classifying tasks in NLP, including punctuation detection tasks as in Huang and Zweig (2002). In this work, we model the n frequently deleted words using the binary ME model[1]. Therefore, each word is assigned its own ME model.

We start by creating a list of frequently deleted words in the training data as in Section 9.2.1. For each word in the list, we observe training data and whenever there is the word of the list we extract relevant features. For features we used followings:

- Current word w_0 and its POS
- Adjacent words $(w_{-3}, ..., w_3)$ and their POS
- Whether there is a phrase boundary where reordering occurs

For part-of-speech, we used both Tree Tagger, introduced in Schmid (1994), and RF Tagger, described in Schmid and Laws (2008), in order to model more fine-grained

[1] http://www.umiacs.umd.edu/∼hal/megam/

information. We choose the window size of seven. The word and its six adjacent words are considered as feature. In addition to the features from words and their POSs, we also used the information whether there is a reordering occurred around the word.

Experiments and Results For the deletion task, we choose 10 for the number of words considered. Table 9.4 shows the accuracy of deletion detection based on ME model in F-score.

Precision	Recall	F-score
35.25	28.10	31.27

Table 9.4: Performance of deletion detection using ME model (in F-score)

The score reported considers all n words. We extract the decision label from each ME model and apply to the test data sequentially. Thus, if a word is one of the n-most frequently deleted words in the training data and labeled as one to be deleted, we extracted this label and removed the word from the test data. The test data is then measured in BLEU against the second step of the annotation, which already includes all reconstructions, such as rephrase, reordering, deletions and insertions. By measuring the performance in BLEU, we aim to evaluate the similarity between the automatically deleted test data and the manually reconstructed one. Table 9.5 shows the result.

System	BLEU
Baseline	80.92
Deletion by ME model	81.26

Table 9.5: Performance of deletion detection using ME model (in BLEU). The score is obtained by evaluating the test data against the human-generated reconstruction.

The baseline is the test set without deletion detection applied, but its all disfluencies are removed according to the annotation. When removing the words to be deleted by the ME model, we achieve a 0.34 BLEU point increase against the manually reconstructed test data.

9.2.2 Replacement

As presented in table 9.2, annotators often replace certain words with other ones, in order to avoid colloquial expressions or build more formal sentence structures. In this section, we aim to model the replacement of words using ME models.

Similar to the deletion step, most frequently replaced words are assigned with ME models. When the same replacement is applied more than θ times for a certain word throughout the training data, the word is assigned with an ME model. The threshold θ is empirically chosen.

When there is more than one possible replacements for a word, a multi-class ME model is trained for the word. Otherwise a binary ME model is trained. As features, we use the adjacent words and POSs as described in Section 9.2.1.1. In addition, we use the parser features extracted from Rafferty and Manning (2008). The parser features are:

- Whether this word is the head of the sentence
- The head word of the sentence
- POS of the head word of the sentence

For training, we take the same part of the annotated data used in the previous step. Assuming the oracle condition, all manually annotated deletion words are removed from the training data. For testing, we use the output data automatically generated from the deletion detection step.

9.2.2.1 Data Sparsity

A challenge in this task is data sparsity. Not only is the annotated training data itself very limited (4,000 sentences), the annotation itself is not always consistent. For example, the word *was* in the training data is replaced with *das* for 18 times, while it is replaced with *etwas* 32 times. The word remains unaltered for 367 times. Another word, *Mal* is replaced with *einmal* for 23.6% of the occurrence.

The negative impact of this data sparsity issue can be observed in Table 9.6, where the results of the experiments using only the annotated data are shown. The performance is measured in the same way as in the previous section, by measuring the BLEU between the automatically generated test set and the manually reconstructed test data.

BLEU	81.30
Correct decisions	7,101/7,745
Correct keeping decisions	7,074/7,088
Correct changing decisions	27/657
No. replacements	47
Precision	57.45
Recall	4.12
F-score	7.69

Table 9.6: Results of replacement step using the annotated data only

Throughout the replacement experiments, we choose 5 for the threshold θ based on preliminary experiments. An operation of replacement is added to the ME models when it occurred more than five times in the annotated training data. Altogether, we see that there are 7,745 decisions to make based on this threshold, either to keep the word or to change it into one of its candidates.

Although the model did not change many words that were already correct, we can observe that many of the words that should have been replaced were not changed. The relatively high precision and very low recall rate indicate that the sparse features negatively affected the detection performance.

9.2.2.2 Maximum Entropy-based Modeling with Artificial Data

In order to solve the data sparsity issue discussed in the previous section, we try to build artificial data. From the annotated data, we learn the possible candidates of replacement for each word and create the artificial data based on these observations.

For example, as discussed in Section 9.2.2.1, the word *was* can be replaced with *etwas* or *das*, depending on the context. In order to generate the artificially spontaneous corpus, we observe the well-written corpus. Whenever we spot a word *etwas* and *das*, we replace it with *was*.

The modeling technique introduced in Section 9.2.2.1 is then applied to this artificially built spontaneous data. For the well-formed, written-style data, we take 1.5M sentences of the EPPS corpus introduced in Section 5.3.1.

The results of this method are shown in Table 9.7. It is shown that the artificial data harms the performance, by letting the model observe examples much more frequently

BLEU	74.25
Correct decisions	5,723/7,745
Correct keeping decisions	5,579/7,088
Correct changing decisions	144/657
No. replacements	1485
Precision	9.70
Recall	23.41
F-score	13.72

Table 9.7: Results of replacement step using the artificial data

than it is in the annotated data. By replacing words too often, the precision is largely dropped.

In order to avoid this issue, we apply the same frequency of the replacement operation in the annotated data when building the artificial data. For example, we already observed the frequency of replacement for the word *was* in the annotated data. For 7.48% of chance it is replaced with *etwas*, and for 4.21% of chance it is replaced with *das*. Therefore, instead of changing the word *was* into *etwas* all the time when building the artificial data, only 7.48% of occurance of the word *was* is replaced with *etwas* in the EPPS data. The artificial data is then merged with the annotated data. The results of using this data are presented in Table 9.8.

BLEU	81.38
Correct decisions	7,111/7,745
Correct keeping decisions	7,040/7,088
Correct changing decisions	71/657
No. replacements	134
Precision	53.80
Recall	10.81
F-score	17.96

Table 9.8: Results of replacement step using the artificial data sampled according to the statistics

The results show that the artificial data sampled according to the frequency of the annotation not only increases the overall F-score, but also the BLEU score. By using

this technique, we can improve the BLEU score by 0.12 points, compared to the test data after the deletion detection step.

9.3 Summary

In this chapter, we show the initial efforts on sentence reconstruction, where casual styled German lectures are reconstructed into a formal speech. We defined the problem scope into two challenges: deletion and replacement of words. The performance is measured in BLEU between the automatically reconstructed sentences and the manually reconstructed ones in order to see their similarity.

The deletion is performed on the ten most frequently deleted source words in the annotated data. For each word we build a maximum entropy model. This method alone brought the improvement of 0.3 BLEU.

On this deletion-performed data we applied the replacement process. An artificial data is utilized for this task, where we inserted artifical noise on the source side according to the patterns we learned from the annotated data.

From these two steps, we achieved 81.38 BLEU points, which is around 0.5 BLEU points better than the sentences before the reconstruction.

10

Evaluation in End-to-end Systems

In previous chapters we introduced different techniques to transform speech transcripts into a format which approaches that of written text. By doing so we aim to achieve improved translation performance for spoken language.

In this chapter, we will give a detailed overview of each technique and its effect on the overall translation performance. We will first briefly review the spontaneous data sets from two genres, to which the techniques developed in this thesis are applied.

Not only did the punctuation and segmentation insertion technique using a monolingual translation system show a decent performance on those two data sets, it also performed outstanding in international evaluation campaigns. After the presenting its performance on the in-house spontaneous data briefly, we show the impact of the suggested model by applying it to several language pairs in the evaluation campaigns.

Afterwards we recap the performance improvement in machine translation of spontaneous speech by applying disfluency removal techniques additionally. The performance of the CRF-based disfluency detection model is summarized. This model later is integrated into the SMT system.

Finally, we review the joint modeling of punctuation and disfluency. Two different modeling techniques, NNs and CRFs, are combined log-linearly for this task.

10.1 Genres

As described in Section 5.1, we use an in-house spontaneous speech corpora in order to evaluate the techniques developed in this thesis. In order to address the problem of modeling spontaneousness more thoroughly, we chose two corpora with different characteristics and degrees of spontnaoues speech.

The first one is the university lecture corpus in German. Compared to other manuscript talks such as TED or political speech, lecture data contains much more disfluency. Not only does the spontaneousness affect the grammaticality and readability of utterances, it also poses a difficulty in defining proper sentence-like units. As our second spontaneous speech data source, we used the multi-party meeting corpus in English. While each lecture in the university lecture data is held by a single speaker, each meeting session in this data set has 5 to 12 participants. The interactions between the participants cause more partial sentences and interrupted phrases, which negatively affects the MT performance.

Detailed categories of disfluency and punctuation marks are annotated in the two corpora. All lecture data is translated into English, while only selected portion of the multi-party meeting data has a reference translation in French.

10.2 Monolingual Translation System for Segmentation and Punctuation Insertion

This thesis introduced a monolingual translation system as an effective means of inserting punctuation marks and segmentation prior to the translation of speech transcripts. The monolingual translation system translates from a non-punctuated source language into a punctuated source language, and has shown a great performance to improve the subsequent translation quality.

In this section, we summarize its impact in diferent tasks, for language pairs, and in different scenarios.

10.2.1 Results on German Lecture Test Data

In Chapter 6.2.2, the monolingual translation system is applied to the German lecture data and improved the following MT performance greatly. The resummarized numbers

are given in Table 10.1.

System	BLEU
ASR output	20.70
+ Punctuations from monolingual translation system	22.23
Oracle	22.48

Table 10.1: Performance of monolingual translation system as a punctuation model on the lecture data. The monolingual translation system is used to punctuate the German lecture data. The performance is measured by translating the punctuated test data into English.

An ASR output of a subset of German lecture data described in Section 5.1 is translated into English, and we achieve 20.70 BLEU points. Keeping the potential word errors in this test data, we insert punctuation marks into it using the monolingual translation system. Using this system, we achieved around 1.63 BLEU points of improvement. This score is impressively only 0.25 points worse than the result of the oracle experiment, where the manually created segmentation and punctuation marks are inserted, based on the word edit distance between the ASR output and its manual transcript.

10.2.2 Results on English Multi-party Meeting Data

The monolingual translation system is also applied to English multi-party meeting data, for different disfluency conditions. In Table 10.2, we are comparing the three punctuation methods to punctuate English multi-party meeting data. All numbers are compared in two conditions, either the test data contains all disfluencies or no disfluencies based on the human annotation. Each test set is then translated into French, and its quality is measured in BLEU.

We can see that inserting punctuation marks using the monolingual translation system outperforms the conventional LM-based method by around three to four BLEU points, depending on whether we keep all disfluencies or not.

System	With disfluency	No disfluency
LM based segmentation	9.53	12.21
Punctuation from monolingual translation system	12.44	16.34
Oracle punctuation	15.64	20.93

Table 10.2: Performance of monolingual translation system as a punctuation model on the meeting data. The monolingual translation system is used to punctuate the English multi-party meeting data, under two different disfluency conditions. The performance is measured by translating the punctuated test data into French.

10.2.3 Streaming Input System for Latency

Since the original input format of the monolingual translation system is based on sliding window of 10 words, it is likely that this will cause a latency issue in a real-time spoken language translation scenario. While maintaining this format for our off-line systems, we modified the input format for the on-line system.

For this system, we design a punctuation module which takes the output from the resending ASR, that constantly outputs its current best hypothesis. In this punctuation module, we use the streaming input rather than the overlapping window. Utilizing a history stack of the ASR system output, we aim to remove the structural delay that the overlapping window induces.

Punctuation	ASR Output	Manual Transcript
LM, Prosody	9.74	-
Punctuation using overlapping input	11.18	19.57
Punctuation using streamingInput	11.55	19.41

Table 10.3: Performance when using the streaming input. Streaming input for monolingual translation system can generate a comparable performance, decreasing the structural latency.

Table 10.3 demonstrates that the streaming input method can maintain a comparable performance to the overlapping window approach. By removing the required future contexts in the overlapping window, the streaming input scheme effectively decreases structural latency.

10.2.4 Results from IWSLT Evaluation Campaign

As shown in Section 6.2.2, the monolingual translation system demonstrated a good performance on inserting punctuation marks and segmentation. Encouraged by this, we also applied this technique to test data provided for SLT track of the official IWSLT Evaluation Campaigns.

For the 2013 and 2014 IWSLT Evaluation Campaign, the test sets for SLT track were distributed in a way where sentence boundaries, according to the manual transcript, are given but without punctuation marks. It is necessary to modify the system introduced in Section 6.2, so that it does not insert sentence boundaries but only punctuation marks instead. For the 2015 IWSLT Evaluation Campaign, on the other hand, no sentence boundaries are given. Instead, only a very simple language model based segmentation was provided. Taking this as our baseline, we applied the monolingual translation system in the original design to augment the source text with proper sentence boundaries as well as other punctuation marks. A detailed description is given in the following section.

10.2.4.1 IWSLT Evaluation Campaign 2013

As previously mentioned, in this evaluation campaign the gold-standard sentence boundaries based on manual transcript are present in the test sets. Therefore, the monolingual translation system is used only for predicting commas, instead of all punctuation marks. In addition to predicting commas, we also predict the casing of words using the monolingual translation system.

Instead of randomly cutting the training data in order to detect sentence boundaries, we use the training data as is. For the source side of the training data of the monolingual translation system, we take the punctuation-removed preprocessed data. At the end of each sentence on the source side, we inserted a period. For the target side, we keep all commas and all sentence-final punctuation marks such as “!”, “?”, “.” are replaced by a period. The only difference between the source and the target side corpus is the inserted commas on the target side.

In this evaluation campaign, we use the monolingual translation system for two source languages, English and German. For the English system we use the true-cased

corpus for the source and the target side. As the test set often contains only lower-cased letters, we take this already lower-cased, preprocessed automatic transcript for translation. In order to match this input during decoding, the source side of a phrase table is lower-cased.

As the case information contains more information for German, the German monolingual translation system is built using lower-cased German source and true-cased target side. All words in the preprocessed German automatic transcript are lowercased, but are translated into true-cased text using the monolingual translation system. In this system, therefore, we are translating a lower-cased source language into a true-cased text with commas when needed.

Once the punctuation marks are inserted, the English test set is translated into German, French and Chinese. The German test set is translated into English and the performance is measured in BLEU. A detailed system description of the monolingual translation system and SMT systems for different language pairs used in the IWSLT 2013 is given in Ha et al. (2013). A description and comparison to other participants' systems can be found in the official report (Cettolo et al., 2013).

Results of German-English Table 10.4 demonstrates how much the monolingual translation system can improve translation quality of speech data. When using the MT system without adapting it to the SLT task for translating the test data, we get 18.33 BLEU points. As the test data does not have any reliable case or punctuation information, we remove these from the phrase table. Using this system, we improve the translation quality by 0.8 BLEU points. The performance is further improved when using the test set punctuated by the monolingual translation system, reaching 20.10 BLEU points.

System	Test
Baseline	18.33
Phrase Table	19.09
MonoTrans Input	**20.10**

Table 10.4: Experiments using monolingual translation system for German→English (SLT)

This system also greatly outperforms the other participants in the evaluation campaign as shown in Table 10.5. The numbers are reported a case-sensitive BLEU and TER. The difference between the two systems is around 4.4 BLEU points. As a comparison, the performance of the MT systems of the participants can be found in Appendix B.1.

System	BLEU	TER
KIT	19.34	62.27
UEDIN	14.92	68.12

Table 10.5: IWSLT 13' official translation results for SLT German-English (SLT_{DeEn})

Results of English-German Table 10.6 shows an overview of the speech translation system for English to German. The baseline is a strong phrase-based system whose performance is boosted using a POS-based language model, a cluster-based language model using MKCLS and a DWL with source context. The baseline system achieves 17.60 BLEU points on the test data. When we replace the input with the test set which went through the monolingual translation system, we achieve around 1.3 BLEU points of improvement. This system is used to generate the translation of the official test set.

System	Test
Baseline	17.60
MonoTrans Input	**18.92**

Table 10.6: Experiments using monolingual translation system for English→German (SLT)

The results on the official test set against other participants is shown in Table 10.7. As can be seen in the table, the system from KIT, which uses the monolingual translation system for inserting punctuation marks into the test set outperforms the system from RWTH by 0.8 BLEU points. Again the MT systems of all participants, which are often used as a baseline for the SLT tasks, are compared in Appendix B.2.

System	BLEU	TER
KIT	18.05	64.46
RWTH	17.27	66.33

Table 10.7: IWSLT 13' official translation results for SLT English-German (SLT_{EnDe})

Results of English-French The punctuation-added English test set is also translated into French. In this translation direction, we tried two distinct methods. As well as the monolingual translation system, we also build a system using translation models from modified GIZA alignments dedicated for ASR data. In order to match the ASR data, we removed the case and punctuation marks for modifying the alignments. This system is referred to as ASR-Dedicated. Table 10.8 shows the results.

System	Test
Baseline	20.75
MonoTrans Input	**23.69**
ASR-Dedicated	22.90

Table 10.8: Experiments using monolingual translation system for English→French (SLT)

A big improvement of around 3 BLEU points is reached when using the monolingual translation system. The ASR-Dedicated system showed its effectiveness as well by improving the baseline system by over 2 BLEU points. For the official test set, we use the monolingual translation system. Its result is displayed in Table 10.9.

System	BLEU	TER
KIT	26.81	55.08
RWTH	25.62	57.21
UEDIN	22.45	61.34
MSR-FBK	22.42	63.69

Table 10.9: IWSLT 13' official translation results for SLT English-French (SLT_{EnFr})

Here KIT's system exhibits a performance difference of 1.2 BLEU points compared to RWTH. As shown in Appendix B.3, when both systems are used to the official MT

task the KIT is only 1 BLEU point better than RWTH. Details of the systems can be found in Cettolo et al. (2013). This underlines the importance of proper punctuation marks prior to translation of speech transcripts.

Results of English-Chinese Inspired by the success of the monolingual translation system on other language pairs, we also use the punctuated test set for the English to Chinese SLT task as a sole participant. It achieved 17.28 BLEU points for our test data, and 16.91 points for the official test set. The details can be found in Cettolo et al. (2013); Ha et al. (2013).

10.2.4.2 IWSLT Evaluation Campaign 2014

The monolingual translation system we used in the IWSLT Evaluation Campaign 2014 is similar to the one described in Section 10.2.4.1. The difference comes from casing. Whereas the casing was handled only in the German system, now the casing is also handled in the English system. There is no additional effort necessary to prepare the lower-cased phrase table. A detailed description of the systems can be found in Slawik et al. (2014). A performance comparison of our system to the other participants' systems in the official evaluation campaign can be found in Cettolo et al. (2014).

Results of English-German The results of using the monolingual translation for the English to German SLT task is shown in Table 10.10. When we replace the input with the one punctuated using the monolingual translation system, the performance is improved by 1.3 BLEU points. The system is further improved by using different language models and rescoring techniques on the punctuated test data and used to submit the final system for the official task. Table 10.11 shows how the KIT system outperforms the other participants' submission in the SLT task for English→German. The results of the MT task of the other participants' are given in Appendix B.4, for comparison.

Results of English-French For English→French SLT task, we take the monolingual translation system as baseline system of our MT system and punctuate the official test set using it. The results of the official evaluation campaign is shown in Table 10.12. Among seven participants, KIT achieved the best performance in BLEU. Compared

System	Dev	Test
Baseline	27.3	17.57
MonoTrans Input	-	18.83
Rescoring	-	18.91
RBMLM	-	19.02
RBMTM	-	18.96
RBMLM+TM	-	**19.01**

Table 10.10: Experiments using monolingual translation system for English→German (SLT)

System	BLEU	TER
KIT	17.05	68.01
RWTH	17.00	68.36
USFD	14.75	70.15
KLE	13.00	71.70

Table 10.11: IWSLT 14' official translation results for SLT English-German (SLT_{EnDe})

to the performance in the MT task, shown in Appendix B.5, the difference between KIT and the other participants is increased by a large amount. This suggests that the adaptation that we are using for the speech transcripts is effective.

System	BLEU	TER
KIT	27.45	57.80
RWTH	26.94	57.29
LIUM	26.82	59.03
UEDIN	25.50	57.23
FBK	25.39	59.53
LIMSI	25.18	60.79
USFD	23.45	59.94

Table 10.12: IWSLT 14' official translation results for SLT English-French (SLT_{EnFr})

Results of German-English Table 10.13 shows different approaches to handle the ASR transcript for MT and their impact. The baseline system uses the best MT system

we have for the evaluation campaign. Using the ASR transcript as it is gives us 16.86 BLEU points as baseline. Nearly 2 BLEU points are gained when we simply insert a final period at the end of each segment. This proves again that punctuation marks greatly influence the translation quality.

When we apply the monolingual translation system to the test data in order to have more detailed punctuation marks in it, we achieve 3.7 BLEU points over the baseline. Another 0.2 BLEU points are gained by reoptimizing the system using the development data also punctuated by the monolingual translation system for the consistency.

System	Dev	Test
Baseline	39.03	16.86
+ Final period	-	18.79
MonoTrans Input	-	20.59
+ MonoTrans Dev	35.79	**20.79**

Table 10.13: Experiments using monolingual translation system for German→English (SLT)

The resulting final submission and its performance is compared to the other participants in Table 10.14. The joint submission of KIT, UEDIN and RWTH, named EU-Bridge as described in Freitag et al. (2014), achieved the best performance. As an individual submission KIT was again the best system, with a difference of 0.7 BLEU points from the other participants. The MT task performance, compared to the other participants, are given in Appendix B.6.

System	BLEU	TER
EU-Bridge	19.09	63.80
KIT	18.34	63.91
UEDIN	17.67	66.04
RWTH	17.24	65.04
KLE	9.95	74.05

Table 10.14: IWSLT 14' official translation results for SLT German-English (SLT_{DeEn})

10.2.4.3 IWSLT Evaluation Campaign 2015

Unlike the previous years' evaluation campaigns, no correct sentence boundaries are given for the SLT track in IWSLT 2015. This means that not only punctuation marks within a segment, but also the segment boundaries themselves need to be augmented. As shown in Section 10.2.4.2, modeling casing information in addition to punctuation marks boosts the subsequent translation performance even further. The systems built in IWSLT 2015 also jointly model punctuation marks and casing information.

For the punctuation insertion scheme, we built three different models for English and two models for German and compared their performance. The monolingual translation system showed the best performance among them and is used for generating our submission to IWSLT 2015 (Ha et al., 2015).

Monolingual Translation System We build a monolingual translation system for two source languages, English and German. The monolingual translation system for punctuating English and German data are trained on the European Parliament data, News Commentary, TED, and the common crawl corpus of each language. Altogether the training data consists of 106.9 million words for English and 85.1 million words for German.

As a preprocessing step, the noisy part of the common crawl data is filtered out using the SVM model described in Mediani et al. (2011). After preprocessing is applied, the normalized training data is resegmented randomly so that punctuation marks can be observed in all possible locations in each line.

For the source side of the training, we removed final periods, commas, question marks, and exclamation marks. Double quotation marks are also removed as they are relatively frequent in TED talks. In addition to processing the punctuation marks, we also lowercased every single word on the source side. Following the previous evaluations' successes, we aim to restore the case information together with the punctuation marks using this single system.

The translation models and language models for the two languages are designed in the same way. The Moses package (Koehn et al., 2007) is used to build the phrase table. We build a 4-gram language model on the entire punctuated target side using the SRILM Toolkit (Stolcke). Word alignment is learned automatically using the GIZA++ Toolkit (Och and Ney, 2003). A bilingual language model (Niehues et al., 2011) is

used along with a 9-gram part-of-speech-based language model. The POS is learned from TreeTagger (Schmid, 1994). We train a 1,000 class cluster (Och, 1999) for the target language and use the cluster codes for the additional 9-gram language model. The models were optimized on the official test set of the IWSLT evaluation campaign in 2012.

CRF based Model As described in Section 3.3, a conditional random field is a sequence labeling framework, which has been used extensively for various natural language processing tasks. We build a CRF-based model for predicting punctuation marks given observed sequence in the English data. In this work, we use the linear chain CRF modeling technique implemented in GRMM package Sutton (2006).

Trained with the default parameters, the model is trained on two output labels for each token: one for punctuation marks and the other for casing information. For punctuation marks, we use four classes: final period, comma, question mark, and exclamation mark. For casing, we use binary training. When a word token is labeled to be cased, we take the most frequently casing form of the word in the training data.

We use the 3.4 million tokens of TED data, which is a genre- and style-matching data to the test data, as our training data. Lexical features, such as word, POS, and their pattern within a window of seven tokens, are used for the modeling. The pattern feature aims to capture repetitions and therefore the surrounding punctuation marks.

NN based Model Punctuation has been modeled as a tagging task in previous research (Huang and Zweig, 2002). Inspired by this, we designed a punctuation prediction model for German and English using neural networks, which have proven themselves to be very useful at classification tasks.

As training data, we used all available parallel data, the EPPS, News, and TED corpora, which sum up to 54.3 million tokens. We choose a five layer feed-forward NN for our punctuation insertion and casing system. The description of the scheme with two output layers and topology are given in detail in Section 8.2.2.2. For pretraining and fine-tuning we used *Theano* (Bergstra et al., 2010).

The input layer of the English system has 1,745 dimensional feature vector, which consists of the same lexical features as the CRF model and uses the same window length of 7. For German we used 1,759 dimensional feature vector. Each word is represented

by a 100 dimensional vector as described in Mikolov et al. (2013a). For POS, we used 1-of-n encoding.

Same as in the CRF model, there are four punctuation classes, final period, comma, question mark, and exclamation mark, as well as a binary class for the casing information. We use the pre-generated map for the casing, so that the most frequently used casing form of the word can be used when the word is to be cased.

Performance Comparison Table 10.15 displays a comparison of the performance of the three different systems for English. They are applied to the official test set of the MT track of IWSLT 2013. The test data is a manual transcript of the TED talks. For this experiment, we removed all punctuation marks in the test data and lower-cased it. We can compare the systems' performance without any influence of word errors encountered from an ASR system.

System	BLEU	F_{punc}	P_{punc}	R_{punc}	F_{case}	P_{case}	R_{case}
MonoTrans	83.20	60.79	59.46	62.19	94.25	97.49	91.23
CRF	79.00	55.31	56.33	54.32	79.54	90.86	70.74
NN	78.57	43.43	54.43	36.13	86.51	87.82	85.23

Table 10.15: Punctuation prediction for English using different techniques. Comparison on punctuation and case detection performance for monolingual translation system, CRF, and NN is given. Tested on manual transcript of the English TED official test data of IWSLT 13'.

The BLEU score is generated by evaluating the test data with system-generated punctuation and case-information against the test data with human-generated punctuation and casing. This number represents how similar the newly punctuated, case test set is to the human-generated one. We also measured the precision, recall, and F-score of both punctuation marks and casing information. The precision is denoted as P, recall as R and F-score as F. The two labels are represented as *punc* and $disf$ for punctuation and disfluency respectively in the table. The score suggests that when we use the monolingual translation system we can achieve the best score in all measures. Especially the NN-based model seems more conservative than the other techniques when predicting punctuation marks, lowering its F-score.

We use the three systems also for punctuating the official test set of the SLT track of IWSLT 2014. The scores are shown in Table 10.16.

System	BLEU
MonoTrans	60.38
CRF	59.30
NN	57.19

Table 10.16: Performance of punctuation and case information systems for the English ASR test data

Due to word errors, we did not measure the F-scores but only the BLEU score against the manual transcript. The result shows the similar tendency. We achieved the best score by using the monolingual translation system.

The comparison between monolingual translation system and neural networks was made for the German test data and is shown in Table 10.17. In order to measure the performance difference between the two techniques without any word errors, we take the official test data of MT track from the IWSLT 2014.

System	BLEU	F_{punc}	P_{punc}	R_{punc}	F_{case}	P_{case}	R_{case}
MonoTrans	79.84	61.98	63.47	60.56	93.84	92.52	95.19
NN	75.79	54.41	67.11	45.76	91.42	90.71	92.15

Table 10.17: Punctuation prediction for German using different techniques. Comparison on punctuation and case detection performance for monolingual translation system and NN is given. Tested on manual transcript of the German TEDx official test data of IWSLT 13'.

Similar to the results for English, the monolingual translation system shows the better performance. The same result can be found for the ASR test set in Table 10.18.

System	BLEU
MonoTrans	53.47
NN	50.90

Table 10.18: Performance of punctuation and case information systems for the German ASR test data

Results of German-English After the successful use of the monolingual translation system in the previous years' evaluation campaigns, we used the output of the monolingual translation system as a baseline of our SLT systems directly.

Table 10.19 demonstrates the official SLT results of IWSLT 15'. Among the two participants, KIT showed a better performance by 0.85 case sensitive BLEU points. As can be found in Appendix B.7, KIT's MT system is 0.42 BLEU worse than the one of RWTH. This emphasizes once again that the performance of monolingual translation system as a punctuation prediction module is outstanding. We can overcome the performance difference of the MT systems by inserting better punctuation marks.

System	BLEU	TER
KIT	19.64	62.22
RWTH	18.79	65.18

Table 10.19: IWSLT 15' official translation results for SLT German-English (SLT_{DeEn})

Results of English-German We also participated the English to German SLT track as a sole participant. Using the input punctuated using our monolingual translation system, we achieved 16.18 case sensitive BLEU points. Details of the evaluation campaign along with a description of each task and system can be found in Cettolo et al. (2015).

10.3 Integration of the Disfluency Detection Model into SMT

The CRF-based disfluency detection model, inspired by previous works (Fitzgerald et al., 2009a; Liu et al., 2006), is extended in this thesis using a semantically inspired features. We utilized RNNs in order to represent each word into vectors, which are then grouped into different clusters using a k-means algorithm. The word clusters as well as their patterns are used as features in our CRF model. Additionally, we use the phrase table information. For each word or phrase, we check its potential translation in the phrase table.

System	BLEU
Baseline	19.98
+ no *uh*	21.28
CRF	21.94
Oracle	23.14

Table 10.20: Performance of the CRF-based disfluency detection model

By detecting disfluencies using this model, we could improve the translation performance for the German lecture data by around 2 BLEU points, as shown in Table 10.20. In the baseline system, we translated the test data with all disfluencies kept. We provided another baseline system, where the simple disfluencies are removed manually. Compared to this system, we could achieve 0.7 BLEU points of improvement by removing disfluencies detected using our CRF model.

Applying the disfluency detection in a preprocessing step, as shown above, is a conventional approach to translate speech transcripts. However, this has the drawback that choosing which string to translate and determining which string is a disfluency is not dependent on whether or not it will imporve the translation quality. In order to be able to choose a string that is helpful for translation, we integrate the disfluency detection module into the SMT system.

From the CRF model, we extract the disfluency probability for each word, and embed this information on the word reordering lattices. A new path is introduced when the disfluency probability exceeds an empirically obtained threshold. The importance of the newly introduced paths is encoded on the optimized weights.

System	**Dev**	**Text**	**ASR**
Baseline	23.45	22.70	14.50
No *uh*	25.09	24.04	15.10
CRF-hard	25.32	24.50	15.15
Pruning + Tight integration	25.38	**24.84**	**15.51**
Oracle	25.57	24.87	-

Table 10.21: Impact of the disfluency removal integrated into the SMT in BLEU

Table 10.21 shows the results. In the baseline system, all disfluencies are kept and translated. We also provided another baseline system, where all obvious, trivial

disfluencies such as *uh* or *uhm* are removed. Here we also offer the performance of the CRF model, using its output label directly. Compared to the trivial baseline system, we achieved an improvement of 0.5 BLEU points on the manual transcript. When we integrate the disfluency detection into an SMT system using word lattices, however, the translation performance is improved by 0.8 BLEU points for the manual transcript and 0.3 for the ASR output. The best performance was achieved when we use the integration scheme with pruning, where the words with a very high disfluency probability are pruned out before generating the lattices.

10.4 Joint Detection of Punctuation and Disfluency

In Chapter 8 we studied the performance of machine translation for multi-party meeting data where disfluency and punctuation are modeled together. First we show the cascaded model, where disfluencies are detected and then the punctuation is predicted. This scheme is motivated by the available training data for the two models. While disfluencies are trained on the human-annotated, a relatively limited amount of data, punctuation can be trained on all monolingual data containing punctuation marks. Since the monolingual data does not contain any disfluencies, we choose the scheme where the punctuation prediction step is applied once the disfluencies are removed from the data.

In order to give insights into different scenarios, we conduct all experiments under two conditions, depending on the turn information availability. Also, the genre-portability of the CRF-based disfluency detection model is investigated, by building the model using only in-domain or out-of-domain data. The results are summarized in Table 10.22.

System	No turn	Turn
Baseline	9.53	12.93
Cascaded model	14.41	15.01
Oracle	20.93	

Table 10.22: Cascaded approach for punctuation and disfluency in multi-party meeting data

The disfluencies in the multi-party meeting data are detected by the CRF-based

model described in section 7.1. The monolingual translation system, as described in section 6.2, is applied to the test data. When using the cascaded approach, we could improve by 2 BLEU points on the translation, when the availability of the turn information is assumed. When it is not assumed, we could achieve an even bigger improvement of 4.9 BLEU points.

In the following part of the thesis, disfluency and punctuation are modeled jointly in one process. For this task, we use two machine learning techniques with different advantages, CRFs and NNs. Each technique then models punctuation and disfluency jointly. However, instead of taking the output labels, we extracted the disfluency and punctuation probability for each token. The probabilities from two models are features that we combine in the log-linear model. For the combination, we used a language model built on the labels.

System	BLEU
Baseline	14.42
+ No *uh*	14.94
CRF	16.32
NN	16.18
Log-linear combination	16.93
Oracle	22.76

Table 10.23: Combined model for punctuation and disfluency in multi-party meeting data. Translation scores after disfluency removal and punctuation insertion using various systems are measured in BLEU.

The results can be found in Table 10.23. It is shown that when we use the combined model of the two techniques, we achieve the BLEU score of 16.93 points for the multi-party meeting data, outperforming the individual models. This score is also around 2 BLEU points better than translating the test data without simple disfluencies.

10.5 Sentence Reconstruction

In this thesis, we conducted initial experiments on reconstructing spoken-style sentences into a formal, written-style text. The problem scope is defined into two major

issues: deletion and replacement of words. The results of sentence reconstruction are summarized in Table 10.24.

System	BLEU
Baseline	80.92
+ Deletion	81.26
+ Replacement	81.38

Table 10.24: The impact of sentence reconstruction on translation performance. The performance is measured in BLEU.

All scores are evaluated by measuring the text similarity by BLEU, against the manually reconstructed text. The baseline shows the score of the test data where no sentence reconstruction is applied. For the deletion process, we consider the ten most frequently deleted source words in the annotated German lecture data and an ME model is built for each of them. The replacement task is applied on this deletion-performed data. In order to recover from the data sparsity issue, we build an artificial data where we inserted artificial noise learned from the limited size of annotated data. From the two steps of sentence reconstruction, we achieved around 0.5 BLEU points of improvement over the baseline.

11

Conclusion

The processing of spontaneous language poses a great number of challenges for natural language processing tasks, due to its distinctive characteristics compared to written language. While written language generally consists of well-formed, grammatically correct sentences, spontaneous speech very often contains disfluencies. Also, unlike text written by humans, conventional automatic speech recognition systems do not provide reliable sentence boundaries and proper punctuation marks in their outputs.

These differences can negatively impact the performance of subsequent applications, such as machine translation systems, which are in most cases trained using written texts. When we deploy machine translation systems for spoken language, there is a mismatch between the training data and the output of the automatic speech recognition system which recognize the spontaneous speech. As well as degrading the translation quality both, speech disfluencies and the lack of proper punctuation marks, greatly reduce the readability when presenting the recognition of spontaneous speech to users.

From this thesis we learned that a dedicated model for punctuation prediction prior to machine translation can greatly improve its performance. Based on our investigation on specific characteristics of spontaneous speech, we observed that it is possible to model them using different machine learning techniques. By tightly integrating the disfluency detection model into an SMT model, we saw its promising potential to be successfully used in MT. We gained a deeper comprehension of the two challenging issues of spontaneous speech and addressed them by jointly modeling both punctuation prediction and disfluency detection as well as exploiting the synergistic effects of two different ML techniques.

11.1 Summary

In this thesis, we motivated the importance of a proper segmentation and disfluency removal process for the machine translation of spoken language. Describing the challenges in natural language processing of spoken language, we illustrated different types of speech disfluencies. In order to represent different degrees of spontaneousness, we annotated disfluencies in two speech corpora; university lectures and multi-party meeting. We have discussed the creation of the corpora and gave an in-depth analysis on each corpus. The two corpora have been used for training, evaluation, and analysis of automatic segmentation insertion and speech disfluency removal models suggested in this thesis.

After beginning with a discussion on the importance of segmentation and punctuation marks in speech transcripts, this thesis introduced a monolingual translation system that improved the quality of a subsequent machine translation. The monolingual translation system, an MT-driven system that translates non-punctuated speech transcripts into punctuated ones, has been applied to test data in different genres and languages and yielded very good results. This technique is further extended and adapted to a real-time spoken language translation scenario. While decreasing the required context length, and thereby the system latency, we showed that this adaptation can nevertheless maintain the translation performance for spoken language.

In the next part of the work, a conditional random field-based disfluency detection model was presented. Using several features from recurrent neural networks and translation models based on semantics, the model successfully removes speech disfluencies. Later this model is integrated into a statistical machine translation model using word lattices. By optimizing the disfluency path in the translation model, the translation of spontaneous lecture speech is further improved.

This thesis also surveyed the issue of punctuation insertion and disfluency detection as a joint task on the multi-party meeting data. In the first approach we built a conditional random fields-based cascade system which first detects the disfluencies and then augments sentence boundaries and punctuation marks. We demonstrated the advantage of modeling punctuation separately, using commonly available monolingual data. In second approach, punctuation marks and disfluencies are modeled jointly by

combining two techniques with different strengths. In this approach, conditional random fields and neural networks are combined log-linearly and the combination showed synergistic effects improving both the detection accuracy and translation quality of the multi-party meeting data.

Finally, we conducted an analysis on sentence reconstruction, which aimed to give first insights on how sentences that have had their disfluencies removed but are still in a spoken language-style can be reconstructed. Given this analysis, we classified the problem of sentence construction into three parts. In the first part, we aim to delete unnecessary words. Colloquial expressions are then replaced into a form of formal speech. In the last step, words are reordered and the words required to formulate written-style texts are inserted. On our annotated German lecture data, we demonstrated that the suggested sequential model was promising for building well-formed sentences out of spoken-style ones.

From all these experiments, we demonstrated that it is crucial to transform the output from an automatic speech recognition system into a text with well-defined sentence boundaries without speech disfluencies, in order to achieve substantially better translation performance. Our techniques were applied to two source languages, German and English, and two spontaneous speech genres, lectures and multi-party meeting, showing the effectiveness in all conditions.

11.2 Future Work

In the future we hope researchers will apply the insights and techniques presented in this thesis to more languages, diverse genres and scenarios. Since the annotated data for modeling spontaneous speech is expensive, an interesting and practical research direction will be unsupervised training of speech phenomena. Even though prosodic features except for pause duration did not yield significant improvement over the conventional methods for punctuation insertion (Rao et al., 2007b), it would be interesting to confirm the potential gain of using them in future work. We also believe that further topologies of different neural networks on the speech disfluency detection task can be investigated in order to bring more improvements. Another promising research direction would be speech disfluency detection and punctuation insertion for spontaneous speech with code-switching. Combined with language identification, such a system will

be essential to support machine translation systems for multilingual meetings, for example. In addition, although a problem analysis and initial experiments on sentence reconstruction conducted in this thesis provide first insights, there are remains a lot to be done to address the issue extensively. The capability of neural network-based models to represent semantics in a limited data condition is a promising tool to model sentence reformulation allowing versatile reordering, insertion, or replacement of source words or phrases successfully.

Appendices

Appendix A

The Impact of Context Length for Punctuation Insertion

In a related project, we investigate into the relationship between the length of the context and the performance of punctuation prediction models. For a punctuaion prediction model, we explore two methods: the monolingual translation system introduced in Chapter 6 and CRFs. The punctuation prediction models are built on 3 million of English words from TED corpus. For test data, we used 27.1k words of TED talk.

In each model, we fix the length of the past context to four. In order to measure the impact of future context, which has a negative impact on the latency, we control the length of the future context from 0 to 4. The results of these experiments are summarized in Table A.1.

Future context length	0	1	2	3	4
CRF	27.9	50.8	56.2	58.1	57.9
Monolingual translation system	25.7	45.1	48.9	48.9	49.2

Table A.1: Impact of future context lenght on punctuation prediction performance. Performance is given in F-score.

Note that the monolingual translation system is developed with mininal models, unlike the systems used throughout this thesis. They are extended using further translation models in order to enhance the performance. The purpose of this experiment is to observe the punctuation prediction performance for different future context length.

A. THE IMPACT OF CONTEXT LENGTH FOR PUNCTUATION INSERTION

For both modeling techniques, the longer future context the better punctuation prediction performance is achieved. The improvement given by a longer future context, however, tends to saturate after a certain point. This suggests that an optimum future context length can be defined where a similar performance can be yielded while maintaining a shorter latency.

Appendix B

Evaluation Campaigns

B.1 IWSLT 2013

System	BLEU	TER
KIT	26.48	57.52
EU-Bridge	26.33	56.70
NTT-NAIST	25.69	60.96
UEDIN	25.54	59.99
RWTH	25.32	59.67
HDU	22.91	59.65
POSTECH	21.26	67.61
BASELINE	19.25	65.03

Table B.1: IWSLT 13' official translation results for MT German-English, case-sensitive (MT_{DeEn})

System	BLEU	TER
KIT	25.71	54.46
RWTH	24.74	55.52
NTT-NAIST	24.60	54.86
UEDIN	24.00	55.94
POSTECH	22.43	57.57
BASELINE	19.58	59.81

Table B.2: IWSLT 13' official translation results for MT English-German, case-sensitive (MT_{EnDe})

System	BLEU	TER
EU-Bridge	38.86	42.96
KIT	38.63	43.20
UEDIN	38.45	43.96
FBK	37.69	44.13
RWTH	37.67	44.00
PRKE-IOIT	37.59	45.07
MITLL-AFRL	37.05	45.36
BASELINE	31.94	48.59

Table B.3: IWSLT 13' official translation results for MT English-French, case-sensitive (MT_{EnFr})

B.2 IWSLT 2014

System	BLEU	TER
Eu-Bridge	23.25	57.27
KIT	22.66	57.70
UEDIN	22.61	58.95
NTT-NAIST	22.09	57.60
KLE	19.26	61.36
BASELINE	18.44	61.89

Table B.4: IWSLT 14' official translation results for MT English-German, case-sensitive (MT_{EnDe})

System	BLEU	TER
EU-Bridge	36.99	45.20
KIT	36.22	45.18
UEDIN	35.91	45.78
RWTH	35.72	44.54
MITLL-AFRL	35.48	45.69
FBK	34.24	46.75
BASELINE	30.55	49.66
MIRACL	25.86	54.16
SFAX	16.09	62.89

Table B.5: IWSLT 14' official translation results for MT English-French, case-sensitive (MT_{EnFr})

System	BLEU	TER
EU-Bridge	25.77	54.61
RWTH	25.04	55.49
KIT	24.62	55.62
NTT-NAIST	23.77	56.43
UEDIN	23.32	57.50
FBK	20.52	63.37
KLE	19.31	63.88
BASELINE	17.50	65.56

Table B.6: IWSLT 14' official translation results for MT German-English, case-sensitive (MT_{DeEn})

B.3 IWSLT 2015

System	BLEU	TER
RWTH	31.50	47.11
KIT	31.08	47.24
PJAIT	26.08	52.34
BASELINE	21.78	55.45

Table B.7: IWSLT 15' official translation results for MT German-English, case-sensitive (MT_{DeEn})

References

G.H. Al-Gaphari and M. Al-Yadoumi. A method to convert Sana'ani accent to Modern Standard Arabic. *International Journal of Information Science and Management (IJISM)*, 8(1):39–49, 2012. 130

Don Baron, Elizabeth Shriberg, and Andreas Stolcke. Automatic Punctuation and Disfluency Detection in Multi-party Meetings using Prosodic and Lexical Cues. In *ICSLP*, Denver, CO, USA, 2002. 29, 101

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural Probabilistic Language Models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2006. 20

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: A CPU and GPU Math Expression Compiler. In *Proceedings of the Python for scientific computing conference (SciPy 2010)*, volume 4. Austin, Texas, USA, 2010. 20, 46, 115, 151

Jana Besser and Jan Alexandersson. A Comprehensive Disfluency Model for Multi-party Interaction. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue (SigDial 2007)*, volume 8, Antwerp, Belgium, 2007. 4

Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995. 20

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A Statistical

Approach to Machine Translation. *Computational Linguistics*, 16(2):79–85, June 1990. 16

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311, 1993. 16, 29

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 10th IWSLT Evaluation Vampaign. In *Proceedings of the tenth International Workshop on Spoken Language Translation (IWSLT 2013)*, Heidelberg, Germany, 2013. 46, 144, 147

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th IWSLT Evaluation Campaign. In *Proceedings of the eleventh International Workshop on Spoken Language Translation (IWSLT 2014)*, Lake Tahoe, California, USA, 2014. 147

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 12th IWSLT Evaluation Campaign. In *Proceedings of the Eleventh International Workshop on Spoken Language Translation (IWSLT 2015)*, Da Nang, Vietnam, 2015. 154

Stanley F. Chen and Joshua Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics (ACL 1996)*, Santa Cruz, California, USA, 1996. Association for Computational Linguistics. 17

Eunah Cho, Jan Niehues, and Alex Waibel. Segmentation and Punctuation Prediction in Speech Language Translation using a Monolingual Translation System. In *Proceedings of the International Workshop for Spoken Language Translation (IWSLT 2012)*, Hong Kong, China, 2012. 51

Eunah Cho, Christian Fügen, Teresa Herrmann, Kevin Kilgour, Mohammed Mediani, Christian Mohr, Jan Niehues, Kay Rottmann, Christian Saam, Sebastian Stüker, and Alex 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, 2013a. 10, 68

Eunah Cho, Thanh-Le Ha, and Alex Waibel. CRF-based Disfluency Detection using Semantic Features for German to English Spoken Language Translation. In *Proceedings of the International Workshop for Spoken Language Translation (IWSLT 2013)*, Heidelberg, Germany, 2013b. 75, 91

Eunah Cho, Sarah Fünfer, Sebastian Stüker, and Alex Waibel. A Corpus of Spontaneous Speech in Lectures: The KIT Lecture Corpus for Spoken Language Processing and Translation. In *Proceedings of the 9th edition of the Language Resources and Evaluation Conference (LREC 2014)*, Reykjavik, Iceland, 2014a. 33

Eunah Cho, Jan Niehues, and Alex Waibel. Tight Integration of Speech Disfluency Removal into SMT. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, Gothenburg, Sweden, 2014b. 75

Eunah Cho, Jan Niehues, and Alex Waibel. Machine Translation of Multi-party Meetings: Segmentation and Disfluency Removal Strategies. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2014)*, Lake Tahoe, California, USA, 2014c. 1, 101

Eunah Cho, Kevin Kilgour, Jan Niehues, and Alex Waibel. Combination of NN and CRF Models for Joint Detection of Punctuation and Disfluencies. In *Proceedings of the Sixteenth Annual Conference of the International Speech Communication Association (INTERSPEECH 2015)*, Dresden, Germany, 2015a. 101

Eunah Cho, Jan Niehues, Kevin Kilgour, and Alex Waibel. Punctuation Insertion for Real-time Spoken Language Translation. In *Proceedings of the Eleventh International Workshop on Spoken Language Translation (IWSLT 2015)*, Da Nang, Vietnam, 2015b. 52

Christopher Cieri, David Miller, and Kevin Walker. The Fisher Corpus: a Resource for the Next Generations of Speech-to-Text. In *Proceedings of the fourth edition of the Language Resources and Evaluation Conference (LREC 2004)*, volume 4, Lisbon, Portugal, 2004. 29, 33, 37

George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42, 2012. 20

Takao Doi and Eiichiro Sumita. Splitting Input Sentence for Machine Translation Using Language Model with Sentence Similarity. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, Geneva, Switzerland, 2004. 26

Chris Dyer. Using a Maximum Entropy Model to Build Segmentation Lattices for MT. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2009)*, Boulder, Colorado, USA, 2009. 92

Erin Fitzgerald and Frederick Jelinek. Linguistic Resources for Reconstructing Spontaneous Speech Text. In *Proceedings of the 6th edition of the Language Resources and Evaluation Conference (LREC 2008)*, Marrakech, Morocco, 2008. 29, 37, 129

Erin Fitzgerald, Kieth Hall, and Frederick Jelinek. Reconstructing False Start Errors in Spontaneous Speech Text. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2009)*, Athens, Greece, 2009a. 27, 31, 42, 75, 76, 78, 79, 80, 91, 154

Erin Fitzgerald, Frederick Jelinek, and Robert Frank. What Lies Beneath: Semantic and Syntactic Analysis of Manually Reconstructed Spontaneous Speech. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL 2009)*, Singapore, 2009b. 27

Erin Colleen Fitzgerald. *Reconstructing spontaneous speech.* PhD thesis, Johns Hopkins University, Baltimore, Maryland, USA, 2009. 2, 34

George Foster, Roland Kuhn, and John Howard Johnson. Phrasetable Smoothing for Statistical Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, Sydney, Australia, 2006. 17

Markus Freitag, Joern Wuebker, Stephan Peitz, Hermann Ney, Matthias Huck, Alexandra Birch, Nadir Durrani, Philipp Koehn, Mohammed Mediani, Isabel Slawik, Jan Niehues, Eunah Cho, Alex Waibel, Nicola Bertoldi, Mauro Cettolo, and Marcello Federico. Combined Spoken Language Translation. In *Proceedings of the eleventh international Workshop on Spoken Language Translation (IWSLT 2014)*, Lake Tahoe, California, USA, 2014. 149

Christian Fügen and Muntsin Kolss. The Influence of Utterance Chunking on Machine Translation Performance. In *Proceedings of the eighth Annual Conference of the International Speech Communication Association (INTERSPEECH 2007)*, Antwerp, Belgium, 2007. 25, 52

Christian Fügen, Alex Waibel, and Muntsin Kolss. Simultaneous Translation of Lectures and Speeches. *Machine Translation*, 21:209–252, 2007. 4

Qin Gao and Stephan 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. 46

Sebastian Germesin, Tilman Becker, and Peter Poller. Domain-specific Classification Methods for Disfluency Detection. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH 2008)*, Brisbane, Australia, 2008. 34

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. SWITCHBOARD: Telephone Speech Corpus for Research and Development. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1992)*, volume 1, San Francisco, California, USA, 1992. IEEE. xviii, 4, 33

Thanh-Le Ha, Teresa Herrmann, Jan Niehues, Mohammed Mediani, Eunah Cho, Yuqi Zhang, Isabel Slawik, and Alex Waibel. The KIT Translation Systems for IWSLT 2013. In *Proceedings of the tenth International Workshop on Spoken Language Translation (IWSLT 2013)*, Heidelberg, Germany, 2013. 144, 147

Thanh-Le Ha, Jan Neihues, Eunah Cho, Mohammed Mediani, and Alex Waibel. The KIT Translation Systems for IWSLT 2015. In *Proceedings of the Eleventh Interna-*

tional Workshop on Spoken Language Translation (IWSLT 2015), Da Nang, Vietnam, 2015. 150

Hany Hassan, Lee Schwartz, Dilek Hakkani-Tür, and Gokhan Tur. Segmentation and Disfluency Removal for Conversational Speech Translation. In *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH 2014)*, Singapore, 2014. 28, 31

He He, II Alvin Grissom, Jordan Boyd-Graber, Jordan Boyd Graber, and Hal Daumé III. Syntax-based Rewriting for Simultaneous Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, Lisbon, Portugal, 2015. 30

Peter A. Heeman and James F. Allen. Speech Repairs, Intonational Phrases, and Discourse Markers: Modeling Speakers' Utterances in Spoken Dialogue. *Computational Linguistics*, 25(4):527–571, 1999. 26

Teresa Herrmann, Jan Niehues, and Alex Waibel. Combining Word Reordering Methods on different Linguistic Abstraction Levels for Statistical Machine Translation. In *Proceedings of Seventh Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST 2013)*, Atlanta, Georgia, USA, 2013. 50, 92

Donald Hindle. Deterministic Parsing of Syntactic Non-fluencies. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics (ACL 1983)*. Association for Computational Linguistics, 1983. 3

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. 19

Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006. 19

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7):1527–1554, 2006. 20

Matthias Honal and Tanja Schultz. Correction of Disfluencies in Spontaneous Speech using a Noisy-Channel Approach. In *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH 2003)*, Geneva, Switzerlands, 2003. 26

Julian Hough and Matthew Purver. Strongly Incremental Repair Detection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, Doha, Qatar, 2014. 29

Jing Huang and Geoffrey Zweig. Maximum Entropy Model for Punctuation Annotation from Speech. In *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002)*, Denver, Colorado, USA, 2002. 24, 133, 151

Mark Johnson and Eugene Charniak. A TAG-based Noisy Channel Model of Speech Repairs. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL 2004)*, Barcelona, Spain, 2004. 12, 27, 42, 75, 78, 80, 91

Douglas Jones, Edward Gibson, Wade Shen, Neil Granoien, Martha Herzog, Douglas Reynolds, and Clifford Weinstein. Measuring Human Readability of Machine Generated Text: Three Case Studies in Speech Recognition and Machine Translation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005)*, volume 5, Philadelphia, Pennsylvania, USA, 2005. IEEE. 24

Michaeel Kazi, Brian Thompson, Elizabeth Salesky, Timothy Anderson, Grant Erdmann, Eric Hansen, Brian Ore, Katherine Young, Jeremy Gwinnup, Michael Hutt, and Christina May. The MITLL-AFRL IWSLT 2015 Systems. In *Proceedings of the Eleventh International Workshop on Spoken Language Translation (IWSLT 2015)*, Da Nang, Vietnam, 2015. 25

Kevin Kilgour. *Modularity and Neural Integration in Large-Vocabulary Continuous Speech Recognition.* PhD thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, 2015. 19, 20

Reinhard Kneser and Hermann Ney. Improved Backing-off for m-gram Language Modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and*

Signal Processing (ICASSP 1995), volume 1, Detroit, Michigan, USA, 1995. IEEE. 17

Philipp Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the Tenth Machine Translation Summit*, MT Summit X, 2005. 47

Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009. 17

Philipp Koehn and Kevin Knight. Empirical Methods for Compound Splitting. In *Proceedings of the tenth Conference on European Chapter of the Association for Computational Linguistics (EACL 2003)*, Budapest, Hungary, 2003. Association for Computational Linguistics. 48

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech Republic, 2007. 46, 150

Taku Kudoh. CRF++: Yet Another CRF Toolkit. 2007. URL `http://crfpp.sourceforge.net`. 19, 94

Gaurav Kumar, Graeme Blackwood, Jan Trmal, Daniel Povey, and Sanjeev Khudanpur. A Coarse-Grained Model for Optimal Coupling of ASR and SMT Systems for Speech Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, 2015. 30

Girish Kumar, Mike Post, Daniel Povey, and Sanjeev Khudanpur. Some Insights from Translating Conversational Telephone Speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2014)*, Florence, Italy, 2014. IEEE. 30

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional Random Fields: Probabilitic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, Williamstown, Massachusetts, USA, 2001. 18, 19, 93

Quoc V Le. Building high-level features using large scale unsupervised learning. In *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013. 20

John Lee and Stephanie Seneff. Automatic grammar correction for second-language learners. In *Proceedings of the Ninth International Conference on Spoken Language Processing (INTERSPEECH 2006-ICSLP)*, Pittsburgh, Pennsylvania, USA, 2006. 130

Willem J.M. Levelt. Monitoring and self-repair in speech. *Cognition*, 14(1):41–104, 1983. 3

Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8):707–710, 1966. 58

Lori S Levin, Donna Gates, Alon Lavie, and Alex Waibel. An interlingua based on domain actions for machine translation of task-oriented dialogues. In *ICSLP*, volume 8, 1998. 26

Robin J. Lickley. *Detecting Disfluency in Spontaneous Speech*. PhD thesis, University of Edinburgh, Edinburgh, Scotland, 1994. 11

Yang Liu, Andreas Stolcke, Elizabeth Shriberg, and Mary Harper. Using Conditional Random Fields for Sentence Boundary Detection in Speech. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, Ann Arbor, Michigan, USA, 2005. 19

Yang Liu, Elizabeth Shriberg, Andreas Stolcke, Dustin Hillard, Mari Ostendorf, and Mary Harper. Enriching Speech Recognition with Automatic Detection of Sentence Boundaries and Disfluencies. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1526–1540, 2006. 27, 31, 76, 154

Wei Lu and Hwee Tou Ng. Better Punctuation Prediction with Dynamic Conditional Random Fields. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, Cambridge, Massachusetts, USA, 2010. Association for Computational Linguistics. 24

Andrew L Maas, Awni Y Hannun, Christopher T Lengerich, Peng Qi, Daniel Jurafsky, and Andrew Y Ng. Increasing deep neural network acoustic model size for large vocabulary continuous speech recognition. *arXiv preprint*, 2014. 19

Sameer Maskey, Bowen Zhou, and Yuqing Gao. A Phrase-Level Machine Translation Approach for Disfluency Detection using Weighted Finite State Tranducers. In *Proceedings of the Ninth International Conference on Spoken Language Processing (INTERSPEECH 2006-ICSLP)*, Pittsburgh, Pennsylvania, USA, 2006. 27

Evgeny Matusov, Gregor Leusch, Oliver Bender, and Hermann Ney. Evaluating Machine Translation Output with Automatic Sentence Segmentation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2005)*, Pittsburgh, PA, USA, 2005. 50, 58, 59, 98

Iain McCowan, Jean Carletta, W Kraaij, S Ashby, S Bourban, M Flynn, M Guillemot, T Hain, J Kadlec, V Karaiskos, et al. The AMI Meeting Corpus. In *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, 2005. xvii, 4, 34

Mohammed Mediani, Eunah Cho, Jan Niehues, Teresa Herrmann, and Alex Waibel. The KIT English-French Translation Systems for IWSLT 2011. In *Proceedings of the Eighth International Workshop on Spoken Language Translation (IWSLT 2011)*, San Francisco, California, USA, 2011. 150

Takashi Mieno, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Speed or Accuracy? A Study in Evaluation of Simultaneous Speech Translation. In *Proceedings of the Sixteenth Annual Conference of the International Speech Communication Association (INTERSPEECH 2015)*, Dresden, Germany, 2015. 67

Margot Mieskes and Michael Strube. A Three-stage Disfluency Classifier for Multi Party Dialogues. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, 2008. 101

Tomas Mikolov, Martin Karafiat, Jan Cernocky, and Sanjeev Khudanpur. Recurrent Neural Network based Language Model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010)*, Makuhari, Japan, 2010. 20, 82

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781, 2013a. 104, 152

Tomas Mikolov, Wen-Tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*, Atlanta, Georgia, USA, 2013b. 82

Frederic Morin and Yoshua Bengio. Hierarchical Probabilistic Neural Network Language Model. In *Proceedings of the International Sorkshop on Artificial Intelligence and Statistics (AISTATS 2005)*, Barbados, 2005. 20

Graham Neubig, Yuya Akita, Shinsuke Mori, and Tatsuya Kawahara. A Monotonic Statistical Machine Translation Approach to Speaking Style Transformation. *Computer Speech & Language*, 26(5):349–370, 2012. 29, 129

Jan Niehues and Muntsin Kolss. A POS-Based Model for Long-Range Reorderings in SMT. In *Proceedings of the Workshop on Statistical Machine Translation*, WMT 2009, Athens, Greece, 2009. 48, 92, 94

Jan Niehues and Alex Waibel. Using Wikipedia to Translate Domain-specific Terms in SMT. In *Proceedings of the eighth International Workshop on Spoken Language Translation (IWSLT 2011)*, San Francisco, California, USA, 2011. 49

Jan Niehues, Teresa Herrmann, Stephan Vogel, and Alex Waibel. Wider Context by Using Bilingual Language Models in Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation (WMT 2011)*, Edinburgh, Scotland, 2011. 48, 49, 50, 86, 150

Franz Josef Och. An Efficient Method for Determining Bilingual Word Classes. In *Proceedings of the Ninth Conference on European chapter of the Association for Computational Linguistics (EACL 1999)*, Bergen, Norway, 1999. 50, 151

Franz Josef Och. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, Sapporo, Japan, 2003. 46

Franz Josef Och and Hermann Ney. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, Philadelphia, Pennsylvania, USA, 2002. 17, 116

Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003. 46, 132, 150

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Optimizing Segmentation Strategies for Simultaneous Speech Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, Baltimore, Maryland, USA, 2014. 26, 52

Mari Ostendorf, Benoît Favre, Ralph Grishman, Dilek Hakkani-Tur, Mary Harper, Dustin Hillard, Julia Hirschberg, Heng Ji, Jeremy G Kahn, Yang Liu, et al. Speech Segmentation and Spoken Document Processing. *Signal Processing Magazine, IEEE*, 25(3):59–69, 2008. 2, 23, 56

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, Philadelphia, Pennsylvania, USA, 2002. Association for Computational Linguistics. xvii, 15, 21

Matthias Paulik, Sharath Rao, Ian Lane, Stephan Vogel, and Tanja Schultz. Sentence Segmentation and Punctuation Recovery for Spoken Language Translation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2008)*, Las Vegas, Nevada, USA, April 2008. 24, 51

Stephan Peitz, Markus Freitag, Arne Mauser, and Hermann Ney. Modeling Punctuation Prediction as Machine Translation. In *Proceedings of the eight International Workshop on Spoken Language Translation (IWSLT 2011)*, San Francisco, California, USA, 2011. 10, 25, 31, 51, 60, 61, 105

Stephan Peitz, Markus Freitag, and Hermann Ney. Better Punctuation Prediction with Hierarchical Phrase-Based Translation. In *Proceedings of the eleventh International Workshop on Spoken Language Translation (IWSLT 2014)*, Lake Tahoe, California, USA, 2014. 25

Matt Post, Gaurav Kumar, Adam Lopez, Damianos Karakos, Chris Callison-Burch, and Sanjeev Khudanpur. Improved Speech-to-Text Translation with the Fisher and Callhome Spanish–English Speech Translation Corpus. In *Proceedings of International Workshop on Spoken Language Translation (IWSLT 2013)*, Heidelberg, Germany, 2013. 30

Chris Quirk, Chris Brockett, and William B Dolan. Monolingual Machine Translation for Paraphrase Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, Barcelona, Spain, 2004. 29, 129

Anna N. Rafferty and Christopher D. Manning. Parsing Three German Treebanks: Lexicalized and Unlexicalized Baselines. In *Proceedings of the Workshop on Parsing German*, Columbus, Ohio, USA, 2008. 135

Sharath Rao, Ian Lane, and Tanja Schultz. Improving Spoken Language Translation by Automatic Disfluecy Removal: Evidence from Conversational Speech Transcripts. Copenhagen, Denmark, 2007a. 27, 75

Sharath Rao, Ian Lane, and Tanja Schultz. Optimizing Sentence Segmentation for Spoken Language Translation. In *Proceedings of the eighth Annual Conference of the International Speech Communication Association (INTERSPEECH 2007)*, Antwerp, Belgium, 2007b. 10, 23, 54, 59, 161

Frank Rosenblatt. The perceptron, a perceiving and recognizing automaton Project Para. 1957. 19

Kay Rottmann and Stephan Vogel. Word Reordering in Statistical Machine Translation with a POS-Based Distortion Model. In *Proceedings of the International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2007)*, Skövde, Sweden, 2007. 17, 48, 50, 92, 94

Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran. Deep convolutional neural networks for LVCSR. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013. 19

Helmut Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, England, 1994. 48, 80, 133, 151

Helmut Schmid and Florian Laws. Estimation of Conditional Probabilities with Decision Trees and an Application to Fine-Grained POS Tagging. In *Proceedings of the 22nd International Conference on Computational Linguistics, Proceedings of the Conference (COLING 2008)*, Manchester, UK, 2008. 133

Holger Schwenk. Continuous space language models. *Computer Speech & Language*, 21 (3):492–518, 2007. 19

Frank Seide, Gang Li, and Dong Yu. Conversational Speech Transcription Using Context-Dependent Deep Neural Networks. In *Proceedings of the eighth Annual Conference of the International Speech Communication Association (INTERSPEECH 2011)*, 2011. 20

Alan Senior, Georg Heigold, Marc'Aurelio Ranzato, and Ke Yang. An empirical study of learning rates in deep neural networks for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013. 46

Fei Sha and Fernando Pereira. Shallow Parsing with Conditional Random Fields. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL-HLT 2003)*, Edmonton, Canada, 2003. 19

Hassan S. Shavarani, Maryam Siahbani, Rantim M. Seraj, and Anoop Sarkar. Learning Segmentations that Balance Latency versus Quality in Spoken Language Translation. In *Proceedings of the Eleventh International Workshop on Spoken Language Translation (IWSLT 2015)*, Da Nang, Vietnam, 2015. 26, 52

Hiroaki Shimizu, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Constructing a Speech Translation System using Simultaneous Interpretation Data. In *Proceedings of International Workshop on Spoken Language Translation (IWSLT 2013)*, Heidelberg, Germany, 2013. 30

Elizabeth Shriberg, Andreas Stolcke, and Don Baron. Can Prosody Aid the Automatic Processing of Multi-Party Meetings? Evidence from Predicting Punctuation, Dis uencies, and Overlapping Speech. In *ISCA Tutorial and Research Workshop (ITRW) on Prosody in Speech Recognition and Understanding*, 2001. 28, 101

Elizabeth E. Shriberg and Robin J. Lickley. Intonation of clause-internal filled pauses. *Phonetica*, 50(3):172–179, 1993. 3

Elizabeth Ellen Shriberg. *Preliminaries to a theory of speech disfluencies*. PhD thesis, University of California at Berkeley, Berkeley, California, USA, 1994. 2, 12, 13

Isabel Slawik, Mohammed Mediani, Jan Niehues, Yuqi Zhang, Eunah Cho, Teresa Herrmann, Thanh-Le Ha, and Alex Waibel. The KIT Translation Systems for IWSLT 2014. In *Proceedings of the eleventh International Workshop for Spoken Language Translation (IWSLT 2014)*, Lake Tahoe, California, USA, 2014. 49, 147

Hagen Soltau, Florian Metze, Christian Fügen, and Alex Waibel. A One-pass Decoder based on Polymorphic Linguistic Context Assignment. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 200)*, Madonna di Campiglio, Italy, 2001. IEEE. 45

Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. Segmentation Strategies for Streaming Speech Translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*, Atlanta, Georgia, USA, 2013. 25, 52

Andreas Stolcke. SRILM – An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP 2002)*. 46, 48, 117, 150

Andreas Stolcke and Elizabeth Shriberg. Statistical language modeling for speech disfluencies. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1996)*, volume 1. IEEE, 1996. 57

Andreas Stolcke, Elizabeth Shriberg, Rebecca Bates, Mari Ostendorf, Dilek Hakkani, Madelaine Plauche, Gökhan. Tür, and Yu Lu. Automatic Detection of Sentence Boundaries and Disfluencies Based on Recognized Words. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP 1998)*, Sidney, Australia, 1998. 28

Sebastian Stüker, Kevin Kilgour, and Florian Kraft. Quaero 2010 Speech-to-Text Evaluation Systems. In *High Performance Computing in Science and Engineering'11*, pages 607–618. Springer, 2012a. 46

Sebastian Stüker, Florian Kraft, Christian Mohr, Teresa Herrmann, Eunah Cho, and Alex Waibel. The KIT Lecture Corpus for Speech Translation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*, Istanbul, Turkey, 2012b. 4, 39

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. LSTM Neural Networks for Language Modeling. In *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH 2012)*, Portland, Oregon, USA, 2012. 20

Charles Sutton. GRMM: A Graphical Models Toolkit. 2006. URL `http://mallet.cs.umass.edu`. 19, 78, 104, 115, 151

Yulia Tsvetkov, Florian Metze, and Chris Dyer. Augmenting Translation Models with Simulated Acoustic Confusions for Improved Spoken Language Translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, Gothenburg, Sweden, 2014. 30

Laurens Van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008. 82

Anand Venkataraman and Wen Wang. Techniques for effective vocabulary selection. *arXiv preprint cs/0306022*, 2003. 46

Ashish Venugopal, Andreas Zollman, and Alex Waibel. Training and Evaluation Error Minimization Rules for Statistical Machine Translation. In *Proceedings of the Workshop on Data-drive Machine Translation and Beyond (WPT 2005)*, Ann Arbor, Michigan, USA, 2005. 46

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Metwork with a Local Denoising Criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010. 115

Stephan Vogel. SMT Decoder Dissected: Word Reordering. In *Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering*, Beijing, China, 2003. 47, 116

Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme Recognition using Time-Delay Neural Networks. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 1989)*, 37(3):328–339, 1989. 19

Wen Wang, Gokhan Tur, Jing Zheng, and Necip Fazil Ayan. Automatic Disfluency Removal for Improving Spoken Language Translation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2010)*, Dallas, Texas, USA, 2010. 27, 76, 91

Xuancong Wang, Khe Chai Sim, and Hwee Tou Ng. Combining Punctuation and Disfluency Prediction: An Empirical Study. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP 2014)*, Doha, Qatar, 2014. 28

Ye-Yi Wang and Alex Waibel. Modeling with structures in statistical machine translation. In *Proceedings of the 17th international conference on Computational linguistics*, volume 2. Association for Computational Linguistics, 1998. 24

Jia Xu, Richard Zens, and Hermann Ney. Sentence Segmentation using IBM Word Alignment Model. In *Proceedings of the 10th Annual Conference of the European Association for Machine Translation (EAMT 2005)*, Budapest, Hungary, 2005. 26

Wei Xu, Alan Ritter, Bill Dolan, Ralph Grishman, and Colin Cherry. Paraphrasing for Style. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, 2012. 29, 129

Sandrine Zufferey and Andrei Popescu-Belis. Towards Automatic Identification of Discourse Markers in Dialogs: The Case of 'Like'. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue (SigDial 2004)*, Boston, Massachusetts, USA, 2004. 12