Karlsruhe Institute of Technology

Display strategies for dynamic transcription in speech translation

Master's Thesis of

Jieqi Zheng

at the Department of Informatics
Interactive Systems Lab (ISL)
Institute for Anthropomatics and Robotics

Reviewer:	Prof. Dr. Alexander Waibel
Second reviewer:	Prof. Dr. Tamim Asfour
Advisor:	Dr. Jan Niehues

08. November 2016 – 07. May 2017

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

Karlsruhe, 07. May 2017

. . . .

(Jieqi Zheng)

Abstract

Translation has been a bridge to connect people and organizations from different linguistic background. With the grossing trend of globalization in the 21st century, translation by human effort has shown its limitation in time and space. The cost has raised dramatically for multilinguistic large organization such as European Commission. The development of low-cost high-efficiency machine translation is being called for enthusiastically by individuals as well as international corporations. The frontier of the research has reached spontaneous speech machine translation, which allows for a real-time communication between people speaking different languages anytime and anywhere they want. One of the major challenges in spontaneous speech machine translation is the balance between accuracy and latency. This work tests novel display strategies aiming to reduce the latency during the dynamic display of the translated transcription. Different strategies has been designed and implemented to the translation of TED talks, which covers almost all topics and is given in over 110 languages. The results in this work provides first-hand knowledge of the influence of display strategies on the balance of accuracy and latency. It shows promising future for the dynamic translation scheme. Once fully developed, it will allow for a much lower latency without the cost of accuracy, which is key to spontaneous speech translation.

Zusammenfassung

Übersetzung wurde eine Brücke, um Menschen und Organisationen aus verschiedenen sprachlichen Hintergrund zu verbinden. Mit der zunehmenden Tendenz der Globalisierung hat die Übersetzung durch menschliche Anstrengung ihre Einschränkung in Zeit und Raum gezeigt. Die Kosten haben sich für eine mehrsprachige Großorganisation wie die Europäische Kommission drastisch erhöht. Die Entwicklung einer kostengünstigen hocheffizienten maschinellen Übersetzung wird von Einzelpersonen und internationalen Konzernen begeistert gefordert. Die Grenze der Forschung ist die Maschine Übersetzung für spontane Rede. Die spontane Sprachübersetzung ermöglicht eine Echtzeit-Kommunikation zwischen Menschen irgendwann und irgendwo, die verschiedene Sprachen sprechen. Eine der großen Herausforderungen bei der spontanen Sprachübersetzung ist das Gleichgewicht zwischen Genauigkeit und Latenz. Diese Arbeit prüft neuartige Display-Strategien, um die Latenz während der dynamischen Darstellung der übersetzten Transkription zu reduzieren. Für die Übersetzung von TED-Gesprächen wurden verschiedene Strategien entworfen und implementiert. Die Ergebnisse dieser Arbeit liefern die Kenntnisse über den Einfluss von Display-Strategien auf die Balance von Genauigkeit und Latenz. Es zeigt vielversprechende Zukunft für das dynamische Übersetzungsschema. Wenn es voll entwickelt, wird es eine viel geringere Latenz ohne die Kosten der Genauigkeit ermöglichen, es ist einer der wichtigen Schlüssel zur spontanen Sprachübersetzung.

Contents

List of Figures

List of Tables

1. Introduction

1.1. A brief history

Communication between people with different cultural, especially different linguistic backgrounds has long been challenging. Thousands of years ago, our ancestors have imagined how fast and strong can our civilization can grow if there was only one language and all people could communicate without language barriers. The tower of babel was merely a mysterious legend, yet, the real problem behind that story hinders until today. Translation is of great importance in many aspects in our life, for example, from official diplomacy between governments, to international business cooperation, to personal communication between individual persons. Especially with the fast development of technologies in transportation and communication in the past century, the isolation between different countries and individuals is being eliminated culturally by the omnipresent internet and geometrically by the intercontinential airlines. However, the real challenge for people to make a friend in another cultural background or to make a trading deal with a company in another continent comes mainly from the language barriers. The effort which has been invested to get through these barriers has been increasing in the last decades. For example, facing the challenge of 24 official and working language in European Union, 1750 permenant linguistic staff and 6000 supporing staff is working for European Commission [6]. In European Union, around 1 billion euros is spent on translation on 506 language directions [7]. The globalization is boosting these demands every single day. A proportionally increasing cost of translation with the increasing demands is definitely not our best long-term solution. Therefore, translations by human effort, even though probably with a higher quality today, clearly appear to be impotent in the near future.

Machine translation (MT), here, has the most promising potential to be the best solution to these problems. The MT now can already bring a fast and efficient solution translation at relative lower cost compared to hiring fully qualified professional interpreters. A long distance is still to be covered to make it perfect, and dedicated scientists have been working hard on it. Once it is well developed, the MT systems will eventually allow everyone in our planet to be able to communicate with another no matter where and when and what. This is a long dream coming true and it will significantly benefit our economy and advance our civilization.

The spontaneous speech translation is one of the major direction of machine translation development in the 21st century. With the increasing number of international confer-

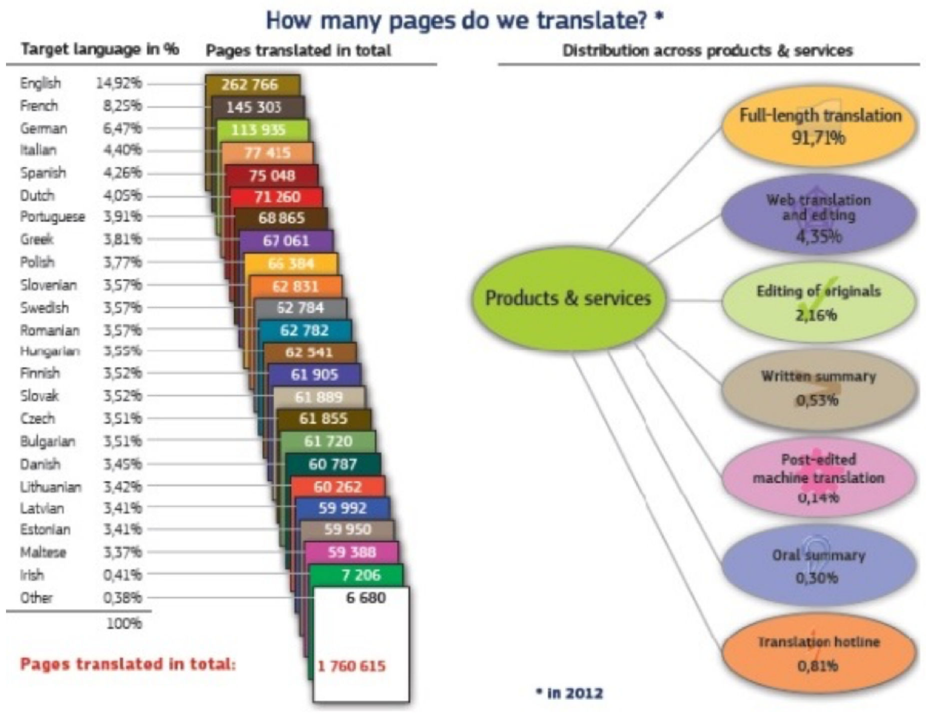


Figure 1.1: An official figure from Translation Directorate General of the European Commission showing the amount of translation work the Commission has carried out [1].

ences and meetings, the spontaneous translation of speech is desperately desired right now. Just imagine an attendee is listening to a conference talk given in a foreign language. It is of great importance for him/her to be able to follow the action of the speaker and the features on the slides spontaneously as the speaker talks. If the translation is offered after the end of the sentence, the speaker probably has already been continuing the talk and presenting the next slide while the listener is merely just receiving the information from the last slide. Besides the conferences and meetings, where professional interpreters might actually be present, at a probably high cost, spontaneous speech translation is needed in many other situations in our life, for example when watching foreign news in a TV, browsing information on the internet, attending lectures in universities while studying abroad.

1.2. Motivation

The importance of display strategy originated from the requirement of real-time translation. In the real-time communication, our ultimate goal is to eliminate the delay due to the MT system, so that the listener in a conversation / talk could real-time follow up the message and vice versa, exactly like they are speaking the same language.

This is especially significant in a real-time dialogue, for example, in a scientific conference talk or during a business negotiation. When the speaker talks, the number of words is increasing with the proceeding of the sentences. The input to our translation system is then on-going sentences. Before the speaker finish a long sentence, the listener could and should already start to interpret his / her meaning. This requires a extremely low latency during the translation process. In order to reduce the latency of speech translation system, a new scheme was presented in [8]. Within this scheme, the current best translation is displayed in an early stage. The sub-optimal outputs will be presented to the user with low latency, and the transcribed text and its translation will be updated later. As the updates are applied, the translation will be better and more accurate during the proceeding of the sentence. When updates are applied, the output of the translation will be rewritten until final version of output is displayed.

The rewriting during the display means that inaccurate information is presented to users. This inaccuracy increases when the number of rewritten words increases. In order to present more accurate information, the most direct strategy is to hide some information, and display later when more context for better translation is available. This simple strategy, however, will cause a information delay. New display strategies are needed, which give a good balance between accuracy and information delay.

The ideal display strategy should be able to perfectly predict the number of rewritten words in the current output at any time. This number of words will then be hidden from the current display. By hiding these words, of course, a information delay will be generated. However, this delay is the minimum delay without any rewrite. The aim of this thesis is to develop new strategies for predicting the number of rewritten words, which can find the best balance between information delay and display rewrites.

1.3. Overview of this thesis

In Chapter 1, after a brief introduction over the machine translation and spontaneous speech translation, the origin for the work in this thesis is elucidated and details about the task will be introduced. In Chapter2, some fundamental knowledge about speech translation is described. A literature review is given in Chapter 3. The important works in the development of spontaneous speech translation and and the main challenges are reviewed. Since this thesis focuses on reducing latency during MT, the literature in this direction is slightly more than other direction. In Chapter 4, some general phenomena concerning translation and language nature is described. Chapter 5 describes the test data used in this thesis with an analysis in detail. The detailed ideas of the developed strategies based on the data analysis are also introduced in this chapter. Chapter 6 presents our results on novel strategies for display of spontaneous speech translation. The effectiveness as well as the mechanisms behind are discussed in depth. Chapter 7 draws the thesis' conclusions and outlook for future work is proposed.

2. Fundamentals: Speech translation

Speech translation technology enables speakers of different languages to communicate. A speech translation system typically consists of the following three software technologies: automatic speech recognition (ASR), machine translation (MT) and voice synthesis. The human speaks into a microphone and the ASR module recognizes the utterance. The input is then converted into a string of words. The MT module then translates this string. As output, the translated text can be directly shown to users or converted into speech using a text-to-speech (TTS) system. Figure 2.1 shows the workflow of an end-to-end speech translation system.

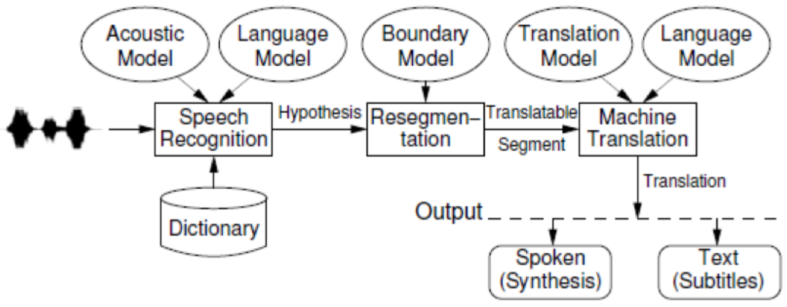


Figure 2.1: A diagram showing the workflow an end-to-end speech machine translation system [2].

The content of spontaneous speech appears more likely to be in a natural, casual and daily style. Therefore, speech translation faces a much more complex situation comparing to a written or well prepared speech. The typical characteristics of spontaneous speech are: frequent use of filler words such as “well” or “um”; repetition of words such as “you know” and “I mean”; change of ideas or the way of expression, etc. These characteristics are usually not expected in well-structured sentences and normally do not help to clarify the meaning of the speaker.

Many works have been carried out in the past decades, and great progress has been made (TC-STAR for parliamentary speeches, GALE for broadcast news, interACT, STAR-DUST, TC-STAR for lectures and seminars). Figure 2.2 presents the progress in both ASR and MT system [2]. A real-time translation system specifically applied in the lectures has been developed specifically for German-English lecture translation in universities [9], [10]. The components of the system are modified for the subject of the lecture domain in order to reduce the latency of the translation. It has been already online and tested in several lectures and achieved a considerable performance in high accuracy and low latency [11], [12].

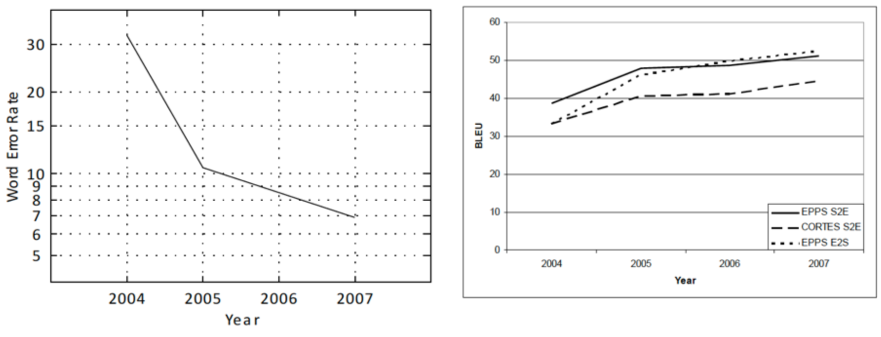


Figure 2.2: Progress of the TC-STAR system from 2004 to 2007. The progress in word error rate and BLEU score are ploted over the time [2].

2.1. Automatic speech recognition

Automatic speech recognition (ASR) is the method used for converting human speech into text by computers. The components of an ASR system are shown in Figure 2.3.

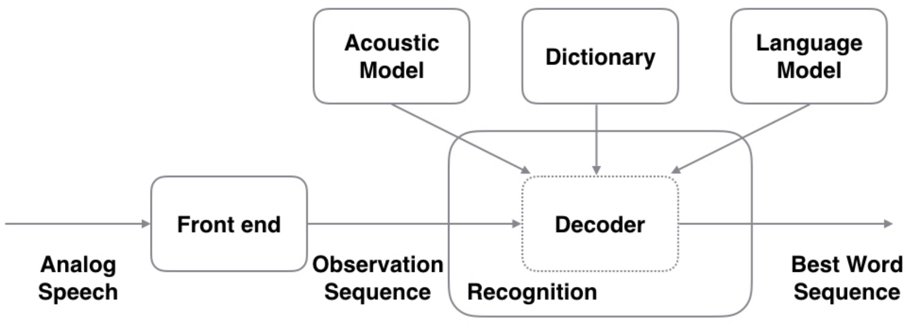


Figure 2.3: Components of a speech recoginition system.

Front End

Front End is the first part of an ASR system. It includes signal processing and spectral features extraction. The *Front End* reduces the influence of undesired components in order to improve the accuracy of the system, for example environmental noise. The *Front End* also reduces the amount of data to improve the efficiency of the system. The typical steps in the processing of *Front End* are: Anti-Aliasing Filter, Analog-to-Digital(AD) conversion, Windowing, fast Fourier transform (FFT), Power-Spectrum computing, feature extraction. The spectrum contains the most important information, from which the spectral features can be extracted. For example, Mel-Frequency Cepstral Coefficients (MFCCs) and Linear Predictive Coefficient (LPC) are two commonly used spectral features. More details can be found in [13].

As shown in Figure 2.3, after the processing of *Front End*, an observation sequence, which is the spectral feature representation sequence of the acoustic observation, is obtained. Denote $O = o_1, o_2, ..., o_k$ as the observation sequence, and $W = w_1, w_2, ..., w_n$ as the word sequence. The *Decoder* find the word sequence W that is the best match of the input observation sequence O. Applying Bayes' rule,

$$P(W|O) = \frac{P(O|W) \cdot P(W)}{P(O)}, \tag{2.1}$$

where $P(O)$ is a constant for a complete sentence.

The *Decoder* chooses the word sequence $\widehat{W}$ that maximize the probability $P(W|O)$ given the observation sequence O. Thus,

$$\begin{aligned} \widehat{W} &= \arg\max_{w} P(W|O) \\ &= \arg\max_{w} P(O|W) \cdot P(W). \end{aligned} \tag{2.2}$$

In Equation 2.2, the probability $P(O|W)$ represent the *Acoustic Model*, and $P(W)$ represent the *Language Model*.

The process of ASR using acoustic model, pronunciation dictionary and language model is shown in Figure 2.4.

Acoustic Model

Acoustic model represents the relationship between an audio signal and the phonemes or other linguistic units that make up a speech. The audio signal is represented in term of feature vectors. Many factors can affect the building of an acoustic model: speakers in different genders and at different ages, the environment in which the audio is recorded, the microphone used. The acoustic model is often based on hidden Markov models (HMMs). Each phoneme has a HMM. The model of a word is the concatenation of the HMMs of its phonemes sequence. More details about HMM and its application in speech recognition can be found in [13].

Pronunciation Dictionary

Pronunciation Dictionary defines the mapping from words to sub-word units, usually phonemes. It means that the pronunciation dictionary contains a list of words with associated pronunciation represented as a combination of phonemes. Generally, most words have a single pronunciation. The variants of pronunciation can cause errors in speech

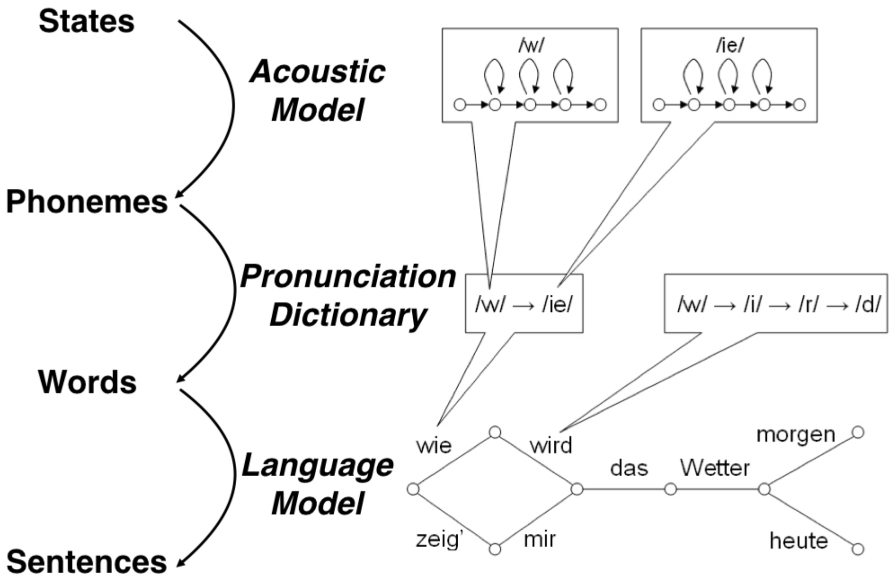


Figure 2.4: Process of speech recognition. The original image is in German from [3].

recognition. The size of the dictionary varies from few to millions of words and depends on the application and the language.

Language Model

Language model is used to model the word sequences in the language. The A-priori-Probability $P(W)$ for a word sequence W is given by language model. In speech recognition, the computer tries to match sounds with word sequences. Generally, the language model presents the linguistic properties of the language and provides context to distinguish between words and phrases that sound similar. Language modeling is used in many language processing applications, such as speech recognition, machine translation, part-of-speech tagging, parsing, handwriting recognition, information retrieval. More details will be presented later in this chapter.

2.2. Statistical machine translation

The task of machine translation is to translate text from one language to an other language. A foreign sentence f is denoted as $f = f_1 f_2 ... f_i ... f_I$, where f_i is a word in source language, I is the length of the sentence. The sentence in target language is denoted as $e = e_1 e_2 ... e_j ... e_J$. The translation tasks turn into finding the most probable translation $\hat{e}$

for a given sentence in the source language f. Applying Bayes' rule,

$$\begin{aligned} \hat{e} &= \arg\max_{e} p(e|f) = \arg\max_{e} \frac{p(f|e)p(e)}{p(f)} \\ &= \arg\max_{e} p(f|e)p(e) \end{aligned} \tag{2.3}$$

Equation 2.3 is proposed by Brown et al. [14] for a word-based translation model. It is the fundamental equation of machine translation. As shown in Figure 2.5 , the translation process can be taken into three parts: the *Translation Model* provides $p(f|e)$, the *Language Model* provides $p(e)$ and the *Decoder* searches for the best translation $\hat{e}$.

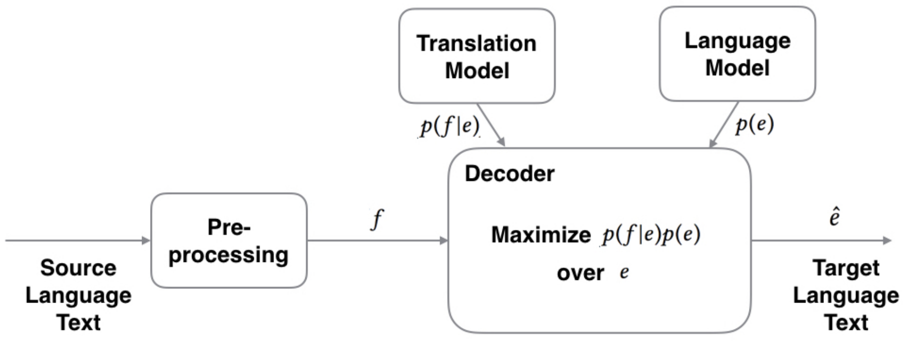


Figure 2.5: A word-based translation system.

2.2.1. Language model

As mentioned above, the language model is used to model the word sequences in the language. It represents the relationship between words. The language model provides the probability $P(W)$ over a given word sequence $W = w_1, w_2, ..., w_m$. The language model indicates whether a sentence is fluent or reasonable. Since some words are more likely to co-occur with others, the language model also indicates whether a word translation fits in the sentence. The occurrence sequence of the words also plays a role. The language model can indicate whether the word order is good in the sentence.

The most common language models are n-gram models. In general, a n-gram model provides the probability of a word given the $n-1$ previous words. The probability $P(W)$ of a word sequence W can be decomposed as:

$$\begin{aligned} P(W) &= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)...P(w_m|w_1, ..., w_{m-1}) \\ &= \prod_{i=1}^{m} P(w_i|w_1, ..., w_{i-1}) \end{aligned} \tag{2.4}$$

Based on the n^{th} order Markov property, it is assumed that the probability of observing the word w_i in the context history of the preceding $i - 1$ words can be approximated by the probability of observing it in the shortened context history of the preceding $n - 1$ words. With this assumption, $P(W)$ can be written as:

$$P(W) \approx \prod_{i=1}^{m} P(w_i|w_{i-(n-1)}, ..., w_{i-1}) \tag{2.5}$$

The n-gram probabilities can be calculated by simply counting the occurrences of n-gram in a text corpus. This is called the maximum likelihood estimation.

$$P(w_i|w_{i-(n-1)}, ..., w_{i-1}) = \frac{count(w_{i-(n-1)}, ..., w_{i-1}, w_i)}{\sum_w count(w_{i-(n-1)}, ..., w_{i-1}, w)} \tag{2.6}$$

The n-gram models with $n = 2$ and $n = 3$ are commonly referred to as bigram and trigram language models respectively. The unigram model is the most simply n-gram model, it provides the probability without the history of previous word.

In practice, problems occur when a n-gram is not seen in training. Therefore, smoothing is necessary to solve this problem. There are many different methods of smoothing. The simplest appoach is called "Add-One" smoothing. It adds a fixed count of 1 to every count. One of the most common used smoothing is called Kneser-Ney smoothing proposed by Kneser and Ney in [15]. A study of smoothing techniques for language modeling [16] has shown that the Kneser-Ney smoothing and modified Kneser-Ney smoothing work well and outperform the other methods.

2.2.2. Translation model

The translation model provides the probability of the target sentence being a translation of the source sentence. The translation models using Equation 2.3 are proposed by Brown et al. [14]. These models translate directly word to word, and model the word-by-word translation probabilities. A word-based translation is not popular nowadays, because there is no strict word to word correspondence for many language pairs. These models (also known as IBM models) are currently used to generate word alignment for phrase extraction in phrase-based translation models.

Phrase-based MT system is the state-of-the-art MT system using phrase-based translation approach [17]. In this approach, the sentence is translated in the unit of phrase instead of word. Here, the phrases typically are not linguistic phrases, they could be any continuous sequence of words. They are automatically extracted from corpora using statistical methods.

The phrase-based translation allows for alignments with different number of words. For example, one Chinese word is often aligned to a multi-word English phrase. This is difficult to generate from word-based translation. The phrase-based model also encapsulates context, which is only modelled by the language model in word-based model. An other advantage of phrase-based model comparing to word-based model is that it allows for local reordering within phrase boundaries.

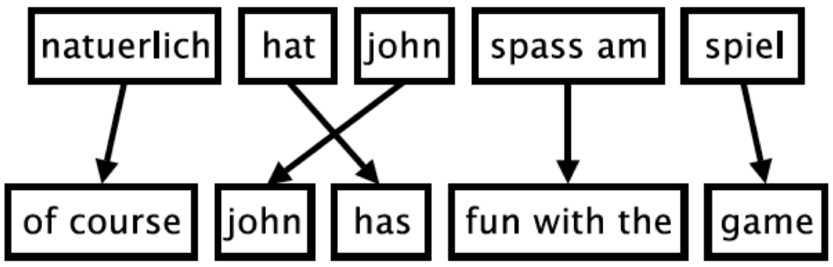


Figure 2.6: Phrase-based translation with reordering [4].

An example of phrase-based translation is shown in Figure 2.6. The sentence is split into non-overlapping phrases, and each phrase is then translated into the target language.

2.2.3. Decoder

The *Decoder* searches for the best translation. It finds the word sequence that maximize $p(f|e)p(e)$ from all possible word sequences in the target language to get the best translation. Here, only two models, language model (LM) and translation model (TM), are taken into account. The fundamental equation Equation 2.3 can be represented as

$$\begin{aligned}\hat{e} &= \arg\max_e p(f|e)^{\lambda_{TM}} p(e)^{\lambda_{LM}} \\ &= \arg\max_e exp(\lambda_{TM} log(p(f|e)) + \lambda_{LM} log(p(e)))\end{aligned} \tag{2.7}$$

where λ_{LM} is a weight for language model and λ_{TM} is a weight for translation model. It can be generalized unsing *log-linear model*:

$$p(x) = exp(\sum_{i=1}^{n} \lambda_i h_i(x)), \tag{2.8}$$

where $h_i(x)$ is a feature function and λ_i is the weight for that feature. For example, for translation model, $h_{TM}(e) = log\, p(f|e)$, λ_{TM} leads to more accurate translations.

Thus, the equation is now represented as

$$\begin{aligned}\hat{e} &= \arg\max_e (exp(\sum_{i=1}^{n} \lambda_i h_i(x))) = \arg\max_e (\sum_{i=1}^{n} \lambda_i h_i(x)) \\ &= \arg\min_e (\sum_{i=1}^{n} -\lambda_i h_i(x))\end{aligned} \tag{2.9}$$

With the log-linear model, the translation system is no longer restricted to translation model and language model. More models that indicate different features can be added to the translation system. As shown in Figure 2.7, a modern phrase-based translation system includes models such as distortion model, word-count model and phrase-count model. Each of these models has a feature function that returns a score. The score of a sentence *e* is the sum of all weighted feature scores. All these feature weights will be tuned in the training step to produce the best translation.

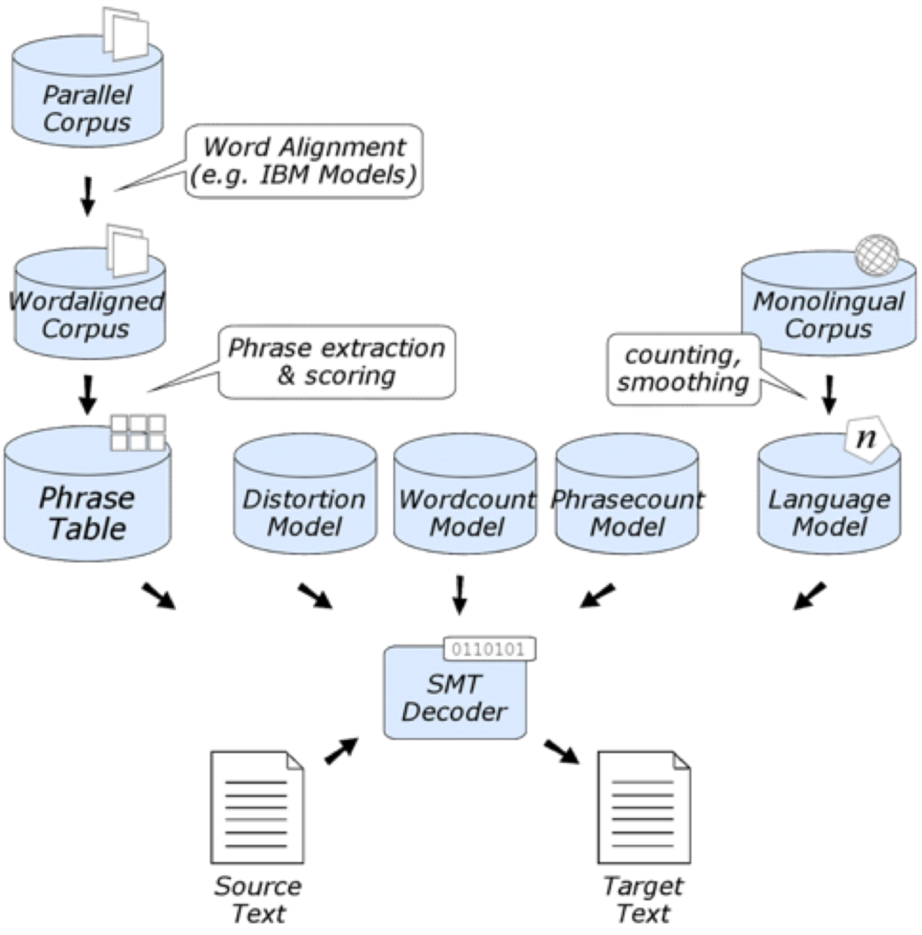


Figure 2.7: A Phrase-based SMT system [5].

3. Related Works

The subject of this work originates from the field of spontaneous speech translation. Challenges and their detailed solutions in this field would be too long stories for this thesis. The general goal of my work is based on the question: how to reduce the latency during the spontaneous speech translation without sacrificing the translation quality. Some of the related work in the field will be reviewed here.

The spontaneous speech translation normally contains two components: ASR and MT. The hypotheses from ASR are sent to MT for translation. The very simplest strategy would be to start the MT process when the ASR recognized that the sentences are finished. Influence of various strategies of the automatic sentence segmentation and punctuation prediction are discussed in [18]. It has been shown that these strategies have improved the interface between ASR and MT. Yet, the latency here with these strategy is too large to support a spontaneous speech translation. The chunking of segments of data from ASR into larger meaningful groups will greatly help to reduce the latency here, and have become the next focus of the research.

Utterance chunking has been considered as a common method to Fügen and Kolss in 2007 [19] investigated the influence of utterance chunking on the latency of the MT with an empirical study. Influence on the MT latency from different chunking strategies such as using sentence boundaries as criterion and using punctuation as criterion (only fits for ASR of Spanish) were compared. Strategies in this work have shown their limitation, leaving open questions for the future.

The chunking process here between ASR and MT does improved the spontaneity of the whole translation process, yet the time lag of translation output is still unsatisfying for spontaneous speech translation. A decoder was then designed to process to streaming input from the speaker [20]. This structure and the algorithm of the decoder have shown better performance than the other sentence segmentation strategies in ASR by then.

A segmentation and punctuation recovery scheme was designed by Paulik et al. in 2008 [21], which significantly improved the spontaneous speech MT performance for three pairs of languages (English to Spanish, Chinese to English and Arabic to English). Additionally, two features were introduced which allow for introducing short intra-sentence segments without sacrificing the translation quality.

An incremental speech translation approach was introduced in 2012 [22] for speech-to-speech translation. The partial hypotheses generated by the ASR is used as the input for MT. The method has shown its advancement in lowering latency. However, when the output translation was synthetic voice, this was at a cost of translation accuracy since the output voice cannot be rewritten. This tradeoff between latency and accuracy remained as a question.

In 2013, Sridhar et al. [23] on one hand improved the accuracy of ASR by applying the vocal tract length normalization (VTLN) and constrain model adaptation (CMA); and on the other hand pointed out that the stalls in pipelines is the major reason for the latency in MT, and buffers would be helpful to improve the synchronization between different components in order to further reduce the latency.

Phrase alignment structure of the language pair [24] was applied as a input segmentation method. Monotonic phrase alignments from were extracted from a word alignment matrix [25], which was used to train the translation program. Here the best result from the tradeoff between latency and accuracy was found using incremental translation of monotone-based segments.

An algorithm for sentence segmentation was proposed by Oda et al. in 2014 [26]. Here, the mean number of words in a segmentation is decided in the first place and then the segmentation boundaries were inserted using greedy search and dynamic programming followed by feature controlled regularization. With this algorithm, the sentences were to be segmented into smaller units for translation and therefore a lower latency is acquired. An expansion of the features applied was foreseen as the future work. Challenging the results in [26], Shavarani et al. in 2015 [27] use Pareto-optimal segmentation approach to achieve a 12% improvement in latency without lowering the quality. The average segment length and the segment number tuned the tradeoff between accuracy and latency.

Facing the fact that a long-range word reordering in some language pairs appears to be inevitable, for example, German-English and Japanese-English, algorithm concerning prediction of the unseen part of the sentence (verbs in this case) has been developed [28]. In this work, the case of German (a typical verb-final language) translated into English (a verb-medial language) is studied specifically. The reinforcement learning was applied to organize prediction and translation into a novel strategy. The method still needed work to be put into action. This very challenging and pioneering idea appears to be very promising to revolutionarily reduce the latency in spontaneous speech translation. Oda et al. in 2015 [29] followed up this idea and developed syntax-based simultaneous translation strategy. Here, both prediction of syntactic constituents and waiting until a meaningful unit is available is implemented. Through experiments in English-Japanese translation for TED talks, an improvement in reducing latency has been seen.

Looking for different strategies for MT to acquire a better accuracy and latency balance, researchers investigated in detail the strategies chosen by human interpreters during the real-time translation and interpretation. He et al. in 2016 [30] studied the differences in simultaneously interpreted text and batch translated text done by human interpreters, in order to model these strategies for MT. For verb prediction, human's and machine's performance was put into test [31] and compared. The results showed that for both human and machine, predicting the verb is possible. The more the sentence was revealed, the better prediction can be made. The authors also suggested a benchmark which can be used in many fields concerning spontaneous translation.

Inspired by human interpreters, Niehues et al. [8] developed a dynamic transcription scheme in order to drastically reduce the latency. Instead of waiting for enough information for accurate translation with large latency, a hypothesis could already be outputted with the proceeding within one utterance in a dynamic fashion. Any improvement / correction to the already outputted transcription is allowed to update the former output in order to guarantee a high final accuracy. Here, the accuracy of the translation is high at the end of a sentence, and inaccuracy in the scheme appears as the inaccurate information displayed during the output updates.

In the existing works, the latency in the spontaneous speech translation is mostly interpreted as a result from where ASR output for the subsequent MT. There is a clear trend that the latency caused by the ASR accuracy and segmentation strategy will soon come to an acceptable range for spontaneous translation. Here another downstream step – spontaneous output of MT results to display – starts to show more and more of its effect on the latency in the whole translation process. This is where this work comes in and embarks on improving the MT display strategy in order to further improve the spontaneous speech translation system performance.

4. Rewrite during display

In order to reduce the latency of speech translation system, a dynamic transcription can be implement. The current best translation is displayed in an early stage. The translation of the on-going sentence will be displayed and updated. The display wil be rewritten until the whole sentence is spoken.

To explain the reasons which cause rewrite during display, an example of English to German translation using MT system is described here. In this thesis, the "input" and "output" means the input to MT system and output from MT system, "display" is the informations presented to users.

Complete sentence: I do not like coffee.		
Input:	Output:	Display without rewrites:
I	Ich	Ich
I do	Ich mache	Ich
I do not	Ich tue nicht	Ich
I do not like	Ich mag nicht	Ich mag
I do not like coffee	Ich mag keinen Kaffee	Ich mag keinen Kaffee

Table 4.1: Example showing rewriting in translation between English and German

To mimic the process of speaking a sentence word-by-word, a complete sentence is broken into segments (Input in Table 4.1). Each segment is the integration of the previous segment and a new added word.

The output of MT system is the translation result of the input segment (Output in Table 4.1). When a new word is added to the input (input is updated), the exisiting translation (output) will be updated (replaced by the translation of the updated segment). This updated output may simply add some new words to the previous output, or rewrite the previous output partially or even entirely.

One can, of course, "honestly" display all the output to users at a cost of a considerable number of rewrites during the display. More rewrites in this process means more inaccurate information is presented to users. As shown in Table 4.1, some words of output can be hidden to reduce rewrites during display in order to present accurate information.

For example, if one "honestly" displays the output to users, there will be 4 rewritten words when the whole sentence is spoken. If one hides some words of each output, there will be no rewritten word while the information presented to users will be delayed. The display will be frozen as "Ich" in the period of 3 updates (see Table 4.1 Display without rewrites).

4.1. Why rewrite

The rewriting during the display cannot be avoided. Here are some reasons cause display rewriting.

1. The machine translation system is normally optimized to work on whole sentences. This means that the translation system can only generate a high-quality translation after the whole sentence is inputted. The translation of uncomplete sentences is normally inaccurate. With more words is added, the previous output of the MT may change.

2. Nature of language.

Lexical ambiguities. Some words have more than one meanings, and the translation of the word depends on context. Therefore, a new added word may change the meaning of present input, and of course the previous output will be rewritten.

Different word order in different languages. Most nominative-accusative languages, which have a major word class of nouns and clauses that include subject and object, define constituent word order in terms of the finite verb (V) and its arguments, the subject (S), and object (O). Theoretically, there are six possible basic word orders for the transitive sentence: subject-verb-object (SVO), subject-object-verb (SOV), verb-subject-object (VSO), verb-object-subject (VOS), object-subject-verb (OSV) and object-verb-subject (OVS). Therefore, rewriting is necessary in translation between languages in different word orders.

4.2. Long-Range Reordering

As mentioned above, different languages has different word orders in the sentence to represent the same meaning. MT systems need to reorder words in the source sentence to produce fluent output in the target language. Comparing to word-based translation, phrase-based machine translation systems can capture short range reorderings within the phrase boundaries.

However, for some language pairs, long-range reorderings have to be performed. This appears very often in translation between languages with different word order, especially in German-to-English and English-to-German translation. These long-range reordering means that the whole sentence may have to be rewritten when the last word of the sentence is known, even if the translation of the sentence before is perfectly correct. This is usually only due to the differences in languages and cannot yet be avoided by improving the MT system. Here are some cases showing the differences which can cause severe rewriting in translation between German and English:

1. Position of the negation. In German, the negation “nicht” likes to travel all the way to the end of a sentence at times. This happens most often with declarative sentences. For example:

*Er hilft mir **nicht**.*

He **doesn’t** help me.

2. Position of the verb. With a compound verb (consisting of a main verb and a helping verb), English usually keeps the two parts together. In German, however, the conjugated verb must be in the second position, while the other verb almost always goes at the end of the phrase. For example:

*Ich **werde** das Buch bald **lesen**.*

I **will read** the book soon.

3. Prefix verb. Verb prefixes in German can be separable or inseparable. A separable prefix usually is moved to the end of a sentence when the verb is conjugated. For example:

*Sie **hörte** mir **zu**.*

She **listened** to me.

4. Subordinate clause. German has verb-medial order in main clauses, but verb-final order in subordinate clauses. The verbs all go at the end of the phrase in subordinate clauses. For example:

*Ich wusste nicht, dass du so klug **bist**.*

I didn’t know that you **were** so smart.

These long-range reorderings introduce great difficulties during the display of the translation of an on-going sentence by presenting inaccurate information to users.

5. Display Strategies

5.1. Analysis

Before the display strategies are developed, the data analysis is necessary.

The data used in this thesis is taken from TED talks. TED talks cover almost all topics in more than 100 languages. The speakers should present their ideas in an innovative way in form of short talks of maximum 18 minutes. TED talks are very good parallel data available free for speech translation.

Here two talks are taken as the test data. The data is a set of bilingual translations: from English to German and from German to English.

TED talk 1 contains 1700 complete sentences, a total of 23192 words in German and 24778 words in English. TED talk 2 contains 1565 complete sentences, a total of 23927 words in German and 25263 words in English.

5.1.1. Data pre-process

To mimic the word-by-word input, preprocessing of the data is necessary.

First, these two talks are separated into sentences, and each sentence is broken into a series of segments. As shown in Table 4.1, sentence "I do not like coffee." will be broken into 5 segments ("I", "I do", "I do not", "I do not like", "I do not like coffee") as mentioned above. These segments are the inputs to MT system. Thus, 4 data are obtained: TED talk 1 in English with 23192 segments, TED talk 1 in German with 24778 segments, TED talk 2 in English with 23927 segments, TED talk 2 in German with 25263 segments.

Second, each data is translated into target language with two different reordering type (long-range reordering and short-range reordering). For example, TED talk 1 in English will be translated into German with short-range reordering (named as TED1.EN-DE.Short), and with long-range reordering (named as TED1.EN-DE.Long). The same procedure and same naming method is applied to the other three data from the first step. For example, TED2.DE-EN.Short refers to the data of TED talk 2 from German to English translation using short-range reordering. The 8 translated text here is the test data in this thesis.

The experiments based on these two talks get similar result, so in this work only the results of TED talk 2 data are shown. All figures and tables shown in this work are based on the results of TED talk 2.

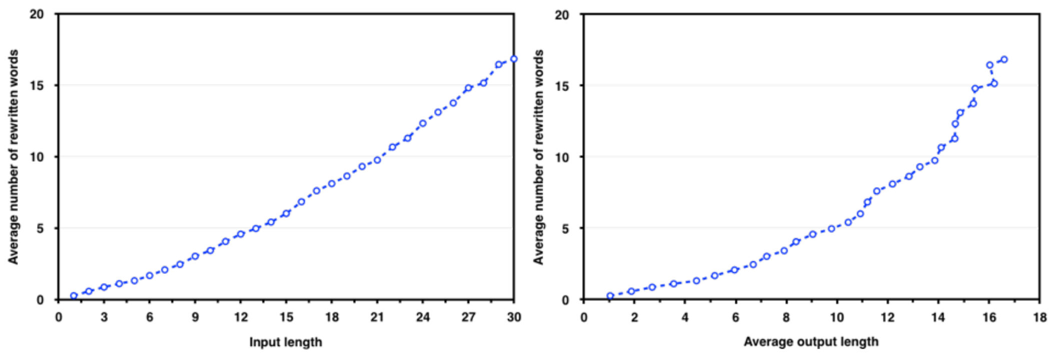
5.1.2. Data analysis

In Chapter 4, the possible reasons causing rewriting during the spontaneous speech translation were described. It is reasonable to speculate that the number of rewritten words will increase with the increasing length of input to the MT system. Before starting to develop new display strategies, it was necessary to analyze our data first to acquire a clear knowledge of the input and the output. The data TED2.EN-DE.Long is analyzed as an example, and part of the results are presented in Table 5.1.

Input length	Average output length	Average number of rewritten words
1	1.04	0.26
2	1.88	0.57
3	2.7	0.86
4	3.55	1.1
5	4.45	1.31
6	5.17	1.67
7	5.95	2.0
8	6.69	2.45
9	7.22	3.02
10	7.91	3.42
11	8.38	4.05
12	9.04	4.57
13	9.78	4.96
14	10.44	5.41
15	10.92	6.01
16	11.2	6.83
17	11.57	7.6
18	12.19	8.1
19	12.84	8.63
20	13.27	9.3
21	13.86	9.75
22	14.11	10.66
23	14.65	11.28
24	14.67	12.32
25	14.86	13.1
26	15.38	13.74
27	15.45	14.8
28	16.21	15.14
29	16.03	16.44
30	16.6	16.83

Table 5.1: English to German translation using long-range reordering based on the TED talk 2 (first 30 rows)

In Table 5.1, the first column is the length of the input to MT system. The second column represents the average length of output from MT system (translation of input). The third column is the average number of rewritten words in the current output (words will be rewritten in next output update).



(a) (b)

Figure 5.1: (a) Change of average number of rewritten words with the increasing of input length; (b) Change of average number of rewritten words with the increasing of average output length. This figure is plotted using the data of English to German translation using long-range reordering based on the TED talk 2.

Figure 5.1 plots the average number of rewritten words against input length and average output length. It can be clearly observed in Figure 5.1a that the average number of rewritten words increases with an increasing length of input, in an almost linear fashion. Figure 5.1b shows that the the average number of rewritten words increases with the increasing of average output length, while the slope of the curve is increasing showing an exponential rise.

Taking an overview of the data, it is worth noticing that the majority of the outputs has less than three rewritten words. For example, for data TED2.DE-EN.Short: There are in total 23927 outputs, and the number of rewritten words for all output is 49678. Among all these outputs, there are 15275 of them that have 0 rewritten words, 4053 have 1 rewritten word, and 2062 have 2 rewritten words. The outputs with less than three rewritten words are 89.4% of the total output number.

The distribution of the output with different number of rewritten words is shown in Figure 5.2. It is interesting that over 60% of outputs have 0 rewritten word. For nearly 90% of outputs, the number of rewritten words is not larger than 2, but these 90% output only rewritten 8177 words, which is about 15% of all rewritten words. This means that only a very small part of outputs causes a huge number of rewritten words. For example, from the data in Figure 5.2, some outputs even need to have more than 20 words rewritten. This is a major contribution to the number of rewritten words, yet it is like “accidents”, which are practically impossible to be foreseen.

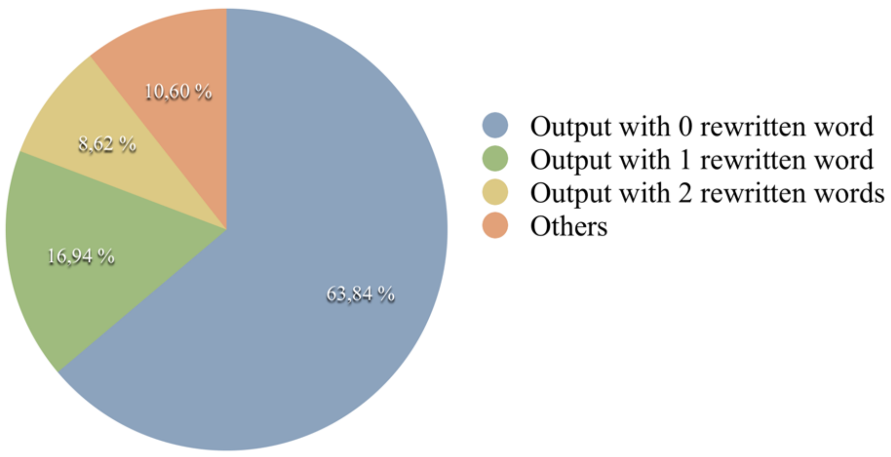


Figure 5.2: Distribution of the number of outputs with different number of rewritten words. This figure is based on the data TED2.DE-EN.Short.

5.2. Strategies

5.2.1. Definitions

As introduced in Chapter 4, the translation process for a word-by-word speaking sentence is that, when a new word is added to the MT system input (input is updated), the exisiting translation (MT system output) will be updated (replaced by the translation of the updated input), and the display to users will also be updated.

For a mathematical representation, assuming that there are X sentences in the data, and there are Y segments for each sentence. An input to MT system can then be denoted as S_IN_{ij}, where i is an index from 0 to $X-1$, j is an index from 0 to $Y-1$. The output from the MT system is denoted as S_OUT_{ij}, and the display to users is denoted as S_DIS_{ij}. For two segments, Seg_1 and Seg_2, function $LM(Seg_1, Seg_2)$ represents the longest match of Seg_1 and Seg_2. The length (number of words) of a segment Seg_1 is denoted as $N(Seg_1)$.

The rewritten words for an output are the words that will be rewritten in next output update. The number of rewritten words for an output is denoted as $N\_RW\_OUT_{ij}$,

$$N\_RW\_OUT_{ij} = N(S\_OUT_{ij}) - N(LM(S\_OUT_{ij}, S\_OUT_{i(j+1)})) \tag{5.1}$$

For each output the number of hidden words is denoted as n_{ij}, which is also known as output delay. The display to users then can be written as the follow equation:

$$S\_DIS_{ij} = S\_OUT_{ij}(0 : (N(S\_OUT_{ij}) - n_{ij} - 1)) \tag{5.2}$$

where $Seg(a : b)$ means part of the segment Seg from word index a to b.

The number of rewritten words for a display is denoted as N_RW_{ij}.

$$N\_RW_{ij} = N(S\_DIS_{ij}) - N(LM(S\_DIS_{ij}, S\_DIS_{i(j+1)})) \tag{5.3}$$

The number of rewritten words, N_RW_i, during display until the whole sentence i is spoken can be expressed as:

$$N\_RW_i = \sum_j N(S\_DIS_{ij}) - N(LM(S\_DIS_{ij}, S\_DIS_{i(j+1)})) \tag{5.4}$$

The average number of rewritten words will then be

$$\overline{N\_RW} = \frac{\sum_i \sum_j N(S\_DIS_{ij}) - N(LM(S\_DIS_{ij}, S\_DIS_{i(j+1)}))}{X} \tag{5.5}$$

The average output delay is expressed as

$$\bar{n} = \frac{\sum_i \sum_j n_{ij}}{X \times Y} \tag{5.6}$$

For example, the results in Table 5.1 are obtained as following:

The average output length can be expressed as $\frac{\sum_i N(S\_OUT_{i0})}{X}$ when input length is 1. Because the output is "honestly" displayed to users, $S\_DIS_{ij} = S\_OUT_{ij}$. The average number of rewritten words can be expressed as $\frac{\sum_i N\_RW_{i0}}{X}$ when input length is 1.

5.2.2. Display strategies

Based on the definitions above, my task, in another wrods, is to develop a strategy to determine the output delay n_{ij}, which can minimize the average number of rewritten words $\overline{N\_RW}$ without increasing $\bar{n}$. Five strategies (including a baseline strategy) are developed to achieve this goal.

Here, an example from data TED2.EN-DE.Long is given in order to better introduce the strategies.

Complete sentence: And we got talking about music, and I got an email from Steve a few days later saying that Nathaniel was interested in a violin lesson with me.				
	Input	**Output**	**Input length**	**Number of rewritten words**
1	And we got talking about music, and I	Und wir reden über Musik , und ich	8	0
2	And we got talking about music, and I got	Und wir reden über Musik , und ich habe	9	1
3	And we got talking about music, and I got an	Und wir reden über Musik , und ich bekam eine	10	3
4	And we got talking about music, and I got an email	Und wir reden über Musik , und bekam ich eine E-Mail	11	0

Table 5.2: An example of segment translation.

Baseline strategy S0:

Hypothesizing that a certain number of words will always be rewritten in the next output update, the most direct strategy is put into test as a starting point: the last n words in the output will be hidden, and the rest of them will be displayed. In this strategy, $n_{ij} = n$. Here n is changed from 0 to 20. Because the average sentence length in English is 15-20 words [32], it is reasonable to set the number of hidden words not bigger than 20. The average number of all the rewritten words during display are recorded, and plotted against the output delay.

Applying strategy S0 to the example in Table 5.2, when $n = 2$, the result is shown in Table 5.3.

Strategy S0: n = 2			
	Display	Delay	Number of rewritten words
1	Und wir reden über Musik ,	2	0
2	Und wir reden über Musik , und	2	0
3	Und wir reden über Musik , und ich	2	1
4	Und wir reden über Musik , und bekam ich	2	0

Table 5.3: An example of strategy S0.

Strategy S1:

As shown in Figure 5.1a, the average number of rewritten words increases with an increasing length of input, the slope of the curve is nearly unchanged. The value of average output delay for certain input length is calculated as a mean value of all 8 data. When $N(S\_IN_{ij}) = 10$, the average output delay is 2.65, when $N(S\_IN_{ij}) = 20$, the average output delay is 6.98, when $N(S\_IN_{ij}) = 25$, the average output delay is 9.87, when $N(S\_IN_{ij}) = 30$, the average output delay is 12.61.

Therefore, for each input, the input length $N(S\_IN_{ij})$ is checked and r is set as a factor to present the ratio of number of hidden words to input length. If $0 < N(S\_IN_{ij}) \leqslant 10$, $n_{ij} = 0 \times r$; if $10 < N(S\_IN_{ij}) \leqslant 20$, $n_{ij} = 1 \times r$; if $20 < N(S\_IN_{ij}) \leqslant 25$, $n_{ij} = 2 \times r$; if $25 < N(S\_IN_{ij}) \leqslant 30$, $n_{ij} = 3 \times r$; if $30 < N(S\_IN_{ij})$, $n_{ij} = 4 \times r$. For r = 1, 2, 3 and 4, four groups of test are performed in this strategy. The value of r is set to approach the value of average output delay mentioned above.

Apply strategy S1 to the example in Table 5.2. The input length $N(S\_IN_{ij})$ of segment 4 is larger than 10, therefore, $n_{ij} = 1 \times r$. For the other three segments, $n_{ij} = 0$. When $r = 2$, the result is shown in Table 5.4

Strategy S2:

Going through the data in greater detail, it is found that when the previous output has rewritten words, there is a strong possibility that the current output also has rewritten words. Therefore, a new strategy (referred to as strategy S2.0) is developed checking the

Strategy S1: r = 2			
	Display	Delay	Number of rewritten words
1	Und wir reden über Musik , und ich	0	0
2	Und wir reden über Musik , und ich habe	0	1
3	Und wir reden über Musik , und ich bekam eine	0	3
4	Und wir reden über Musik , und bekam ich	2	0

Table 5.4: An example of strategy S1.

number of rewritten words in previous output $S\_OUT_{i(j-1)}$, which is $N\_RW\_OUT_{i(j-1)}$, and hidding the same number of words in the current output S_OUT_{ij} and displaying the rest of them.Thus, $n_{ij} = N\_RW\_OUT_{i(j-1)}$. A threshold T for n_{ij} is then set. The tests are done with different thresholds T ($T_1 = 5$, $T_2 = 10$, $T_3 = 15$, $T_4 = 20$).

Further on, it turns out that the number of rewritten words in the current output is sometimes bigger than the previous one. In other words, $N\_RW\_OUT_{ij} \geqslant N\_RW\_OUT_{i(j-1)}$. Therefore, as alternatives of strategy S2.0 in the same frame, another two strategies, S2.1 and S2.2, is set up asigning $n_{ij} = N\_RW\_OUT_{i(j-1)} + 1$ and $n_{ij} = N\_RW\_OUT_{i(j-1)} + 2$, respectively. For each of them, a threshold T ($T_1 = 5$, $T_2 = 10$, $T_3 = 15$, $T_4 = 20$) is set.

Applying strategy S2.1 to the example in Table 5.2, the output delay changes with the rewritten number of previous output, $n_{ij} = N\_RW\_OUT_{i(j-1)} + 1$. For example, for segment 2, the number of rewritten words is 1, the output delay is then set as 2 for segment 3. The result is shown in Table 5.5.

Strategy S2.1: T = 5			
	Display	Delay	Number of rewritten words
1	Und wir reden über Musik , und ich	0	0
2	Und wir reden über Musik , und ich habe	0	1
3	Und wir reden über Musik , und ich	2	1
4	Und wir reden über Musik , und	4	0

Table 5.5: An example of strategy S2.1.

Strategy S3:

Strategy S1 and S2 are designed with the aim of improving the basic strategy S0. Strategy S3.x in this section is sketched up aiming for the ideal situation. My goal in this thesis is to develop a strategy to determine the number of hidden words for each display n_{ij}, which can minimize the average number of rewritten words $\overline{N\_RW}$. The ideal strategy should then allow for perfect prediction of the number of rewritten words $N\_RW\_OUT_{ij}$ for each output, and these words are hidden for a more accurate display to users. In this case, $n_{ij} = N\_RW\_OUT_{ij}$, and there will then be no rewrites during display at a cost of the theoretically lowest output delay.

To approach this strategy, a method of classification is used to predict n_{ij}. The prediction of number of rewritten words for each output is denoted as n_p_{ij}, and then $n_{ij} = n\_p_{ij}$.

First of all, it is important to select the relevant features for classifiers. In real-world situations, we often have little knowledge about relevant features. Therefore, to better represent the domain, many candidate features are introduced.

The following 6 features are choosen:

Feature 1: input length $N(S\_IN_{ij})$

Feature 2: output length $N(S\_OUT_{ij})$

Feature 3: length difference between current output and previous output $N(S\_OUT_{ij}) - N(S\_OUT_{i(j-1)})$

Feature 4: number of rewritten words of previous output $N\_RW\_OUT_{i(j-1)}$

Feature 5: POS tag of last word of the translation of input

Feature 6: POS tag of second last word of the translation of input

Based on the data analysis in Subsection 5.1.2, it is clear that the number of rewritten words increases with the increasing length of input, and it is also increased with the increasing length of output. Therefore, Feature 1 and 2 are choosen as candidate features. For Feature 3, an hypothesis is made based on the data characteristics. In this hypothesis, it is assumed that when the length difference between current output and previous output is too large or negative, it is more likely to have rewrites. Feature 4 is chosen based on the same thoughts as in strategy S2 - when the previous output has rewritten words, the following output has a high possibility also having rewritten words. Feature 5 and 6 is chosen as candidate features based on the speculation, in which different POS-tag of the last two words in the output may influence the chance of the words to be rewritten. For example, an article in German may have a better chance to be rewritten than a noun.

In order to find out which combination of features can get the best classification results, classifiers with different features are tested (shown in Table 5.6).

As the first strategy of S3.x (referred to as S3.0), the possible prediction results (target classes) are the possible number of rewritten words, which are numbers from 0 to 80. Based on the fact that for all the output data, only 0.06% of the outputs have a length of more than 80 words. As discussed in Subsection 5.1.2, over 60% of the outputs have no rewrites, and for nearly 90% of outputs, the number of rewritten words is not larger than 2. To perform a prediction in higher accuracy, a further step reducing the target classes can be implemented. In strategy S3.1, the target classes are reduced to two (0 and 1). If an output has a rewrite, set the number of rewritten words as 1. In strategy S3.2, the target

Classifier	used features	Classifier	used features
1	1	9	1, 4
2	2	10	3, 4
3	3	11	1, 5
4	4	12	3, 5
5	5	13	4, 5
6	5, 6	14	1, 3, 5
7	1, 2	15	1, 4, 5
8	1, 3	16	1, 2, 3, 4, 5, 6

Table 5.6: Classifiers using different features.

classes are reduced to three (0, 1 and 2). If an output has a rewrite of more than two words, set the number of rewritten words as 2.

Applying strategy S3.2 to the example in Table 5.2, the output delay is predicted by the classifier. The result is shown in Table 5.7.

Strategy S3.2: Binary classification, Classifier 16			
	Display	Delay	Number of rewritten words
1	Und wir reden über Musik , und ich	0	0
2	Und wir reden über Musik , und ich habe	0	1
3	Und wir reden über Musik , und ich bekam eine	0	3
4	Und wir reden über Musik , und bekam ich eine E-Mail	0	0

Table 5.7: An example of strategy S3.2.

Strategy S4:

Strategy S1 and S2 are based on the data analysis in Subsection 5.1.2, and strategy S3 used classification with different features to predict the number of hidden words. A combination of strategy S3 and S1/S2 may get some improvement.

Here, strategy S3.2 is choosen. For each output S_OUT_{ij}, denote that p_1 is the prediction of binary Classifier 16 in strategy S3.2, and p_2 is the prediction of strategy S1/S2.x. If $p_1 = 0$ and $p_2 = 0$, then $n_{ij} = 0$; if $p_1 \neq 0$ and $p_2 \neq 0$, then $n_{ij} = p_2$; else $n_{ij} = a$, where a is constant. In strategy S4.1, a is 1, and in strategy S4.2, a is 2.

In other words, this strategy works in a way that when both S3.2 and S1/S2.x predict that the output has no rewrite, then no word is hidden from display; when both of them predict that the output has some rewrites, then the hidden words are the same with S1/S2.x; when only one of them predict that the output has reswrites, then *a* words will be hidden from display.

Applying strategy S4.1 to the example in Table 5.2. For example, the output delay for segment 3 is set as 2 in strategy S2.1 and set as 0 in strategy S3.2, then one word will be hidden from display. The result is shown in Table 5.8.

Strategy S4.1: Combination of S3.2 and S2.1			
	Display	**Delay**	**Number of rewritten words**
1	Und wir reden über Musik , und ich	0	0
2	Und wir reden über Musik , und ich habe	0	1
3	Und wir reden über Musik , und ich bekam	1	2
4	Und wir reden über Musik , und bekam ich eine	1	0

Table 5.8: An example of strategy S4.1.

The experiments and results of all these strategies are shown in Chapter 6.

5.3. Tools

Tools and methods applied in this thesis are presented in this section.

5.3.1. Translation system

The translation system applied in this project is an in-house phrase-based machine translation system. The core of this system is a phrase-based decoder described in [33], which uses a local reordering window of 2 words. The GIZA++ Toolkit is implemented to realize the word alignments over the data. Language models are built using the SRILM Toolkit. The different word order between the languages is modelled using POS-based reordering [34]. For the work in this thesis, two different reordering types are employed: long-range reordering and short-range reordering. To model a long-range word reordering for the translation between German and English, the approach described in [35] is used. The POS-tags for the reordering models are generated with the TreeTagger [36].

5.3.2. Classification software

A classification softeare, MegaM, is used to predict the number of rewritten words. In natural language processing, the maximum entropy models are very popular. The MegaM is an implementation of maximum likelihood and maximum a posterior optimization of the parameters of these models. Three types of problems can be solved using this software: binary classification (classes are 0 or 1), binomial regression ("classes" are real values between 0 and 1; the model will try to match its predictions to those values), and multiclass classification (classes are 0, 1, 2, and so on). In this project, the binomial classification and multiclass classification are used.

6. Experiments and Results

As introduced in Chapter 5, five display strategies (baseline strategy included) are developed. The experiments and results are shown in this chapter.

6.1. Baseline strategy (S0)

In the baseline strategy, the last n words in the output will be hidden, and the rest of them will be displayed. Here n is changed from 0 to 20. The average number of all the rewritten words during display are recorded, and plotted against the output delay.

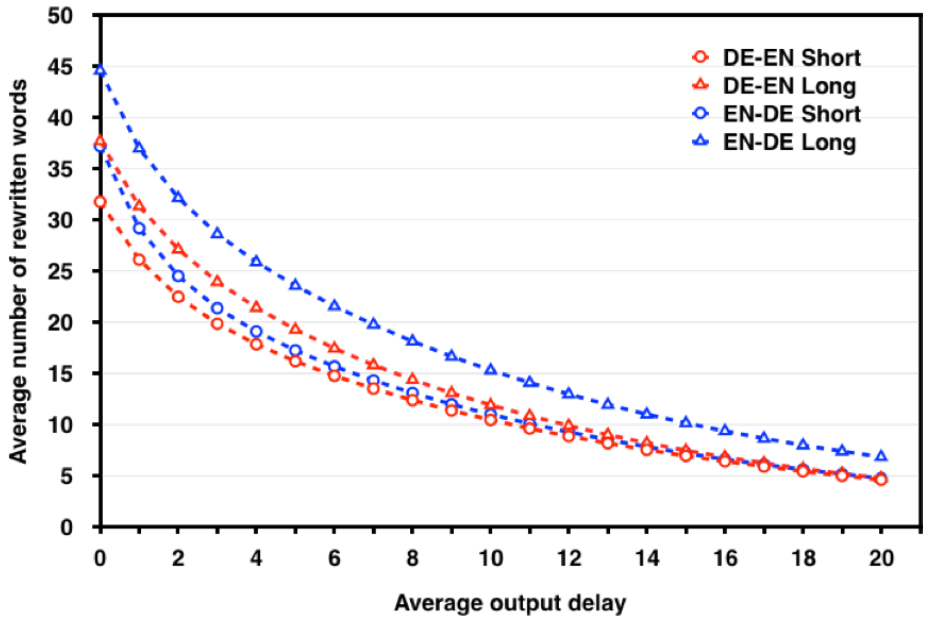


Figure 6.1: Average number of rewritten words plotted against average output delay for baseline strategy S0. This figure is plotted using the four data from the TED talk 2.

In Figure 6.1, x-axis presents the average output delay, also known as $\bar{n}$ in 5.2.1. In this case, n words are hidden from each output, which means $n_{ij} = n$. The y-axis presents the average number of rewritten words, also known as $\overline{N\_RW}$.

In general, the average number of rewritten words decreases with an increasing average output delay, which is expected from our analysis. When the number of average output delay is small, for example from 0 to 4, the change in the absolute value of the slope is larger compared to the case when the average output delay is large. This indicates that in this range (small average output delay) the number of rewritten words drops significantly with the increasing output delay. With the increase of the average output delay (for example after 10), the change in the slope of the curve become smaller and smaller, starting to show a linear feature. The curves are appears to be more and more horizontal. This is due to the fact that most outputs have less than 4 rewritten words (shown in Figure 5.2).

What can also be observed in these curces is that the long-range reordering causes more rewrites than the short-range reordering. The long-range reordering is not ideal for lowering the average number of rewritten words, but it is designed to yield a more accurate translation.

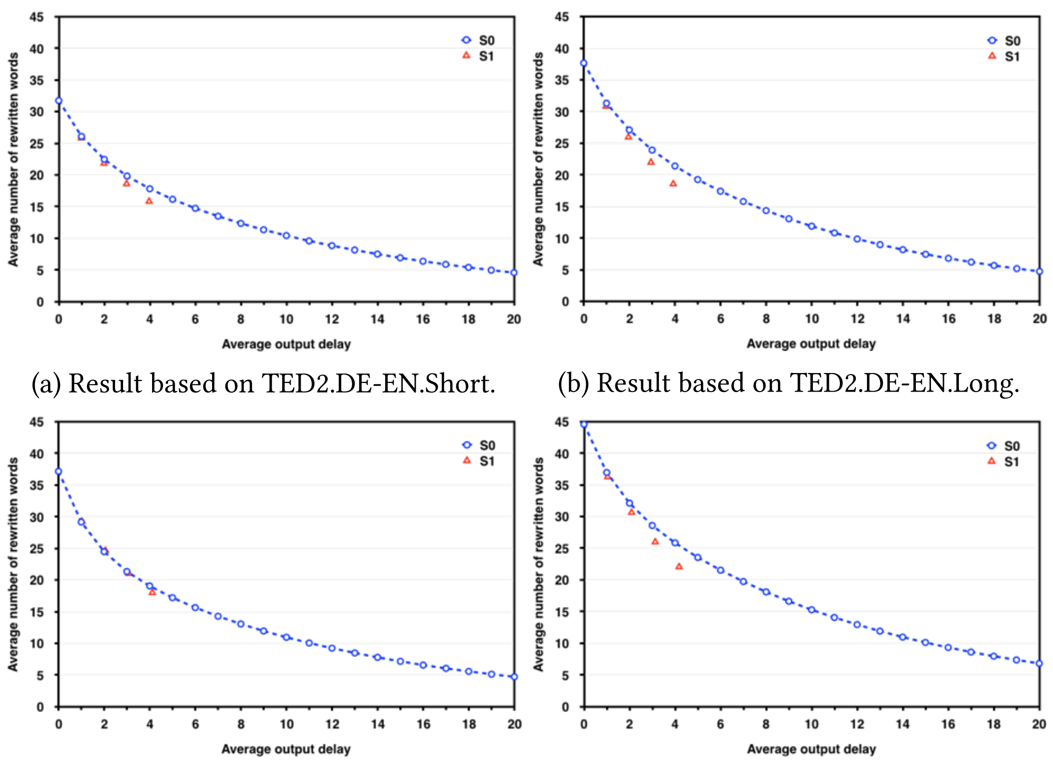
Another interesting point is that English-to-German output has more rewrites than German to English output. This is probably due to the special grammar of German. Besides the reasons summarized in Section **??** Differences between English and German, some additional situation specifically concerning English-to-German translation needs to be discussed here. Unlike English, German nouns can be in three genders: masculine, feminine or neuter. There are four "cases" in German, which correspond to four different roles a noun can play in a sentence (Nominative, Accusative, Dative and Genitive). Adjectives function in German just like that in English, except that they take on case endings when they come right before a noun. The endings of articles and adjectives are determined by the gender and case of the following nouns. For examples, if one wants to translate English phrase "my little", for a dative masculine noun it will be translated as "meinem kleinen", for a nominative feminine noun the result will be "meine kleine". After the noun is inputted to MT system, the words in output before this noun will highly likely be rewritten. This will probably cause some rewrites in short length (below 4).

It is also worth noticing that, with the increasing of output delay from 0 to 4, the change in the slope for EN-DE.Short is slightly more than the other three data. This means that comparing to the others, EN-DE.Short may have more outputs in which the number of rewritten words is less than 4. This may because, on one hand, the short-range reordering causes rewrite with small number of words; on the other hand, as mentioned above, special grammar characteristic of German nouns causes rewrite in a small range before the nouns.

Using this direct strategy as a starting point, the next step is trying to find new strategies which allow for a better balance between the number of rewritten words and the average output delay. With the new strategies, the number of rewritten words should decrease while the ouput delay does not increase. In other words, new results should be able to reach the area closer to the (0,0) point.

6.2. Strategy 1 (S1)

In strategy S1, the number of hidden words increases withe the increasing length of input. In order to appraoch the actually average number of rewritten words, *r* is setted as a factor to present the ratio of number of rewritten words to input length, for *r* = 1, 2, 3 and 4, four groups of test are performed with their results presented in Figure 6.2. The details of results can be found in Table A.1 in Appendix.



(a) Result based on TED2.DE-EN.Short.

(b) Result based on TED2.DE-EN.Long.

(c) Result based on TED2.EN-DE.Short.

(d) Result based on TED2.EN-DE.Long.

Figure 6.2: Average number of rewritten words plotted against average output delay for strategy S1. This figure is plotted using the four data from the TED talk 2.

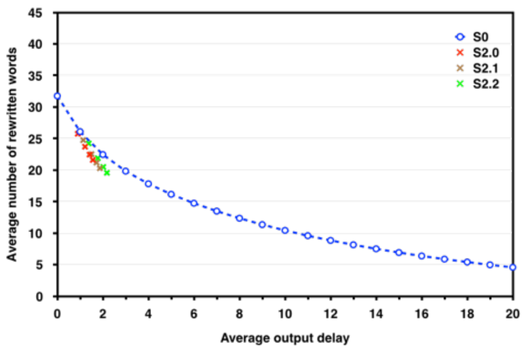
As shown in Figure 6.2, when *r* is increased from 1 to 4, the average output delay increases and the average number of rewritten words decreases. The strategy S1 with *r* = 4 is the most effective. Though the average output delay is more than others, the decrease of average number of rewritten words relative to baseline is the most significant.

What can be observed in this plot is that this strategy is more effective for data using long-range reordering than short-range reordering. And it is notable that, in Figure 6.2c, the results fall nearly into the fit curve from results using baseline strategy S0. And the result of *r* = 2 is even slightliy above the baseline. The reason for this phenomenon can be explained as following: with the increasing of output delay from 0 to 4, strategy S0 is slightly more effective for EN-DE.Short than for the other data (see section 5.1.2), while the effect of strategy S1 is relatively equivalent to all the data. This means that for a data the more outputs there are with less than 4 rewritten words, the more effective is the strategy S0. It is reasonable to believe that in certain extreme situations, the results from strategy S1 will be worse than those from strategy S0. For example, when all the outputs has less than 4 rewritten words.

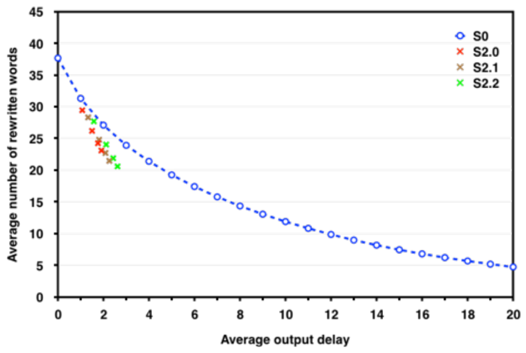
To sum up, comparing to strategy S0, the strategy S1 can not constantly reduce the number of rewritten words with the same output delay. It depends on data.

6.3. Strategy 2 (S2.0, S2.1, S2.2)

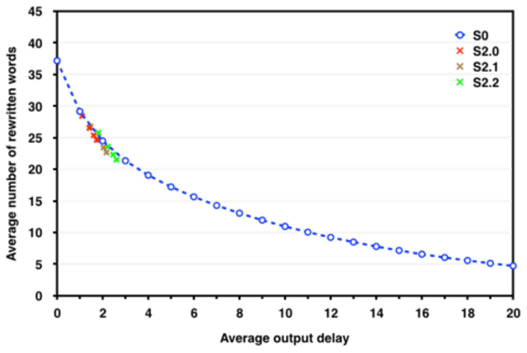
In strategy S2, the number of hidden words for each display is determined by the number of rewritten words in the previous output. Three substrategies are tested (for S2.0, $n_{ij} = N\_RW\_OUT_{i(j-1)}$; for S2.1, $n_{ij} = N\_RW\_OUT_{i(j-1)} + 1$; for S2.2, $n_{ij} = N\_RW\_OUT_{i(j-1)} + 2$.), and the results are shown below. The details of results can be found in Table A.1 in Appendix.



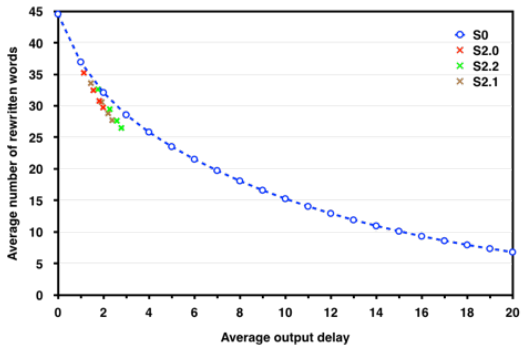
(a) Result based on TED2.DE-EN.Short.



(b) Result based on TED2.DE-EN.Long.



(c) Result based on TED2.EN-DE.Short.



(d) Result based on TED2.EN-DE.Long.

Figure 6.3: Average number of rewritten words plotted against average output delay for strategy S2. This figure is plotted using the four data from the TED talk 2.

As shown in Figure 6.3, for different threshold T, the results of strategy S2 improves with the increasing value of T. The best result in each strategy of strategy S2 comes from $T_4 = 20$. For each strategy, with the increasing T, the improvement appears to be less and less pronounced. This indicates that there may be a limitation of improvement by simply increasing the value of T.

It is an interesting point that most of results have an average output delay around 2. Its reason somehow remains elusive. Compared to short-range reordering data, strategy S2.x shows a more significant improvement for long-range reordering data. This indicates that, during the output update for long-range reordering data, the outputs contains rewrites tends to appear one after another.

In general, strategy S2.x gets an improvement over the baseline strategy S0 and offers more stable improvement compared to strategy S1.

6.4. Strategy 3 (S3.0, S3.1, S3.2)

In strategy S3, the classification method is used to predict the number of hidden words. The MegaM toolkit is used with two different classification types: binomial classification and multiclass classification. The classifiers are trained on the data from TED talk 1 and test on the data from TED talk 2. Four groups of test results are obtained from TED2 (EN-DE.Short, EN-DE.Long, DE-EN.Short, DE-EN.Long), in this section, only the results from TED2.DE-EN.Long are shown as an example, the results of others can be found in Table A.2, Table A.3 and Table A.4 in Appendix.

The results using strategy S3.1 are shown in Table 6.1.

In Table 6.1, the column “accuracy” is the percentage of correct prediction. “average rewritten” lists the average number of rewritten words during display until a whole sentence is spoken, also known as . “average delay” represents the average output delay for each display,. The row “do nothing” represents the results of “honestly” displaying the output to users, and “perfect prediction” represents the result when all the rewritten words in output are perfectly predicted and hidden from display. The cell highlighted with red color is the highest accuracy among all 16 classifiers, and the cell highlighted with green color is the lowerst average number of rewritten.

As shown in Table 6.1, for both binary and multiclass classification, Classifier 3 has no effect (the same result as “do nothing”); Classifier 5 has the highest prediction accuracy; and Classifier 13 can get the lowest average number of rewritten words. It has be mentioned that for binary classification, the results using Classifier 9, 10, 15, 16 is somehow unreasonable. This is probably due to the unexpected effects of combining different features.

As shown in Table 6.1, all 16 classifiers do not yet bring satisfactory results: the accuracy of prediction is not high enough, and comparing to “do nothing”, the improvement in reducing average number of rewritten words is still rather large. This may be because the characteristics of the data: as discussed in section 5.1.2, over 60% of the outputs have

	binary			mutilclass		
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**37.66**	**0.00**		**37.66**	**0.00**
perfect prediction		**0.00**	**2.46**		**0.00**	**2.46**
Classifier 1	58.38 %	37.44	0.02	57.31 %	37.00	0.08
Classifier 2	58.37 %	37.47	0.02	57.22 %	37.19	0.05
Classifier 3	58.39 %	37.66	0.00	58.39 %	37.66	0.00
Classifier 4	58.32 %	37.39	0.03	57.88 %	36.95	0.07
Classifier 5	59.13 %	36.81	0.07	59.13 %	36.81	0.07
Classifier 6	59.06 %	36.79	0.07	59.06 %	36.90	0.06
Classifier 7	58.29 %	37.25	0.09	51.18 %	35.35	0.33
Classifier 8	58.37 %	37.41	0.03	56.69 %	37.09	0.06
Classifier 9	0.38 %	7.74	16.00	54.99 %	36.59	0.12
Classifier 10	0.13 %	2.16	34.63	58.00 %	36.08	0.17
Classifier 11	59.10 %	36.54	0.09	58.57 %	36.28	0.12
Classifier 12	59.13 %	36.81	0.07	58.59 %	37.01	0.06
Classifier 13	59.07 %	36.34	0.11	58.91 %	35.27	0.21
Classifier 14	59.11 %	36.56	0.09	56.52 %	36.90	0.08
Classifier 15	1.88 %	19.25	5.00	56.36 %	36.90	0.08
Classifier 16	0.35 %	6.80	16.00	54.38 %	36.14	0.17

Table 6.1: Results of S3.0 prediction using both binomial and multiclass classification for all classifiers. These results are based on TED2.DE-EN.Long.

no rewrites, and for nearly 90% of outputs, the number of rewritten words is not larger than 2. This being said, the rest 10% of the outputs contribute most of the rewritten words. Therefore, those outputs with long rewrites may be ignored as "accidents". The classifiers trained on these data may tend to classifiy the test data to class 0, 1 or 2.

Due to the reasons above, there migh be too many target classes in strategy S3.0. To perform a prediction in higher accuracy, a further step reducing the target classes can be implemented.

In strategy S3.1, the target classes are reduced to two (0 and 1). If an output has a rewrite, set the number of rewritten words as 1. The results using strategy S3.1 are shown in Table 6.2.

In strategy S3.2, the target classes are reduced to three (0, 1 and 2). If an output has a rewrite of more than two words, set the number of rewritten words as 2. The results using strategy S3.2 are shown in Table 6.3.

	binary			mutilclass		
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**37.66**	**0.00**		**37.66**	**0.00**
perfect prediction		**31.30**	**0.42**		**31.30**	**0.42**
Classifier 1	59.99 %	36.48	0.14	59.74 %	34.94	0.34
Classifier 2	60.20 %	36.43	0.14	58.39 %	37.66	0.00
Classifier 3	58.39 %	37.66	0.00	58.39 %	37.66	0.00
Classifier 4	61.60 %	36.70	0.09	61.60 %	36.70	0.09
Classifier 5	63.78 %	36.29	0.13	63.71 %	36.41	0.11
Classifier 6	63.96 %	36.15	0.14	63.99 %	36.24	0.13
Classifier 7	60.07 %	36.43	0.14	58.75 %	35.79	0.24
Classifier 8	60.03 %	36.44	0.14	58.39 %	37.66	0.00
Classifier 9	61.95 %	36.32	0.14	62.03 %	36.51	0.11
Classifier 10	61.62 %	36.69	0.10	61.62 %	36.69	0.10
Classifier 11	64.34 %	35.65	0.20	58.22 %	34.25	0.45
Classifier 12	63.78 %	36.29	0.13	58.39 %	37.66	0.00
Classifier 13	64.79 %	36.03	0.15	64.94 %	35.86	0.17
Classifier 14	64.38 %	35.71	0.20	58.39 %	37.66	0.00
Classifier 15	65.11 %	35.63	0.20	62.43 %	36.35	0.13
Classifier 16	65.13 %	35.43	0.22	61.63 %	36.27	0.15

Table 6.2: Results of S3.1 prediction using both binomial and multiclass classification for all classifiers. These results are based on TED2.DE-EN.Long.

It should be mentioned that in strategy S3.0, when all rewritten words are prefectly predicted and hidden from display, there will be no rewritten word during display. However, in strategy S3.1 and S3.2, the rewrites still exist even when the prediction is perfect.

Similar to the situation in strategy S3.0, Classifier 3 in strategy S3.1 and S3.2 has no effect. Unlike in strategy S3.0, Classifier 9, 10, 15, 16 work fine in strategy S3.1 and S3.2. For binary classification, Classifier 16 has the best performance, which has both the highest accuracy and lowest number of rewritten words. For multiclass classification, Classifier 13 has the highest accuracy.

In Table 6.2 and Table 6.3, only the results for data TED2.DE-EN.Long are shown. The classifiers that perform the best here may not perform as well as for other data. Observing the results from all the data (TED2.DE-EN.short, TED2.DE-EN.long, TED2.EN-DE.short, TED2.EN-DE.long), Classifier 16 of binary classification has a generally better performance.

Strategy S3.1 has a higher accuracy compared to strategy S3.2, while strategy S3.2 has a larger improvement in reducing the average number of rewritten words.

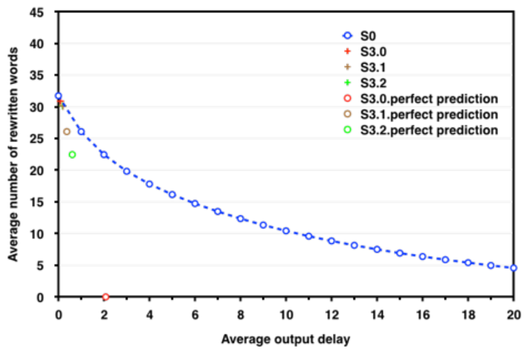
		binary			mutilclass	
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**37.66**	**0.00**		**37.66**	**0.00**
perfect prediction		**27.09**	**0.69**		**27.09**	**0.69**
Classifier 1	59.24 %	36.31	0.16	59.24 %	36.31	0.16
Classifier 2	59.18 %	36.29	0.16	59.18 %	36.29	0.16
Classifier 3	58.39 %	37.66	0.00	58.28 %	37.62	0.00
Classifier 4	60.57 %	36.30	0.14	60.57 %	36.30	0.14
Classifier 5	61.39 %	36.20	0.13	61.39 %	36.20	0.13
Classifier 6	61.45 %	36.05	0.15	61.39 %	36.00	0.16
Classifier 7	59.24 %	36.24	0.17	58.39 %	37.66	0.00
Classifier 8	59.31 %	36.23	0.17	59.27 %	36.31	0.16
Classifier 9	60.50 %	35.88	0.19	60.48 %	35.91	0.19
Classifier 10	60.53 %	36.28	0.14	60.51 %	36.30	0.14
Classifier 11	61.65 %	35.09	0.27	59.25 %	36.41	0.15
Classifier 12	61.39 %	36.20	0.13	61.35 %	36.21	0.13
Classifier 13	62.33 %	35.33	0.23	62.35 %	35.35	0.22
Classifier 14	61.56 %	35.17	0.26	59.17 %	36.20	0.18
Classifier 15	62.26 %	34.89	0.29	58.41 %	37.65	0.00
Classifier 16	62.39 %	34.79	0.30	60.08 %	35.92	0.19

Table 6.3: Results of S3.2 prediction using both binomial and multiclass classification for all classifiers. These results are based on TED2.DE-EN.Long.

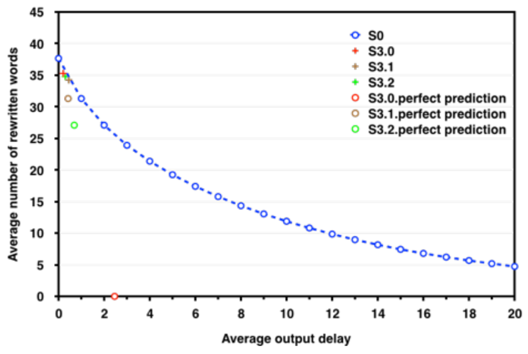
For strategy S3.0, S3.1 and S3.2, each of them has 32 results for all classifiers. Most of the points are concentrated in a very small area. Therefore, only one result from each strategy is shown in Figure 6.4.

It is shown in Figure 6.4, among all the three strategies, strategy S3.0 has the poorest performance for all four data. The average number of rewritten words may even be higher than using baseline strategy S0 (e.g. as shown in Figure 6.4d). Strategy S3.1 performs better for German to English translation, while strategy S3.2 performs better for English to German translation. However, the improvement of both strategy S3.1 and strategy S3.2 referenced to the baseline strategy S0 are limited.

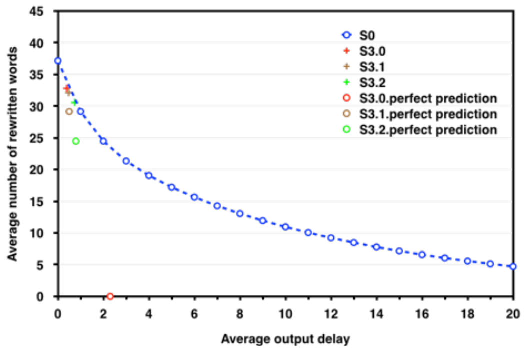
In general, even though using classification to predict the number of rewritten words is expected to bring a more systematic design of strategies, the results do not appear to be promising. Most of the prediction are 0, and a small part of them are 1. There is almost no other prediction. The reason could be, as mentioned in section 5.1.2, that the outputs with long rewrites are only a small part of the outputs, which can hardly be predicted.



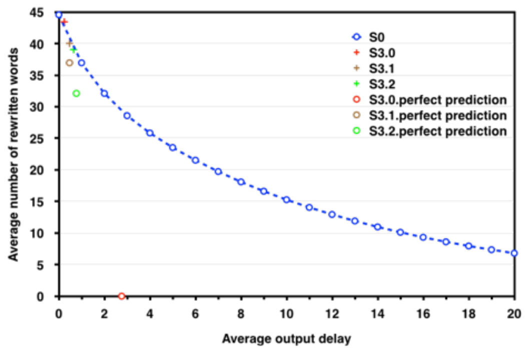
(a) Result based on TED2.DE-EN.Short.



(b) Result based on TED2.DE-EN.Long.



(c) Result based on TED2.EN-DE.Short.



(d) Result based on TED2.EN-DE.Long.

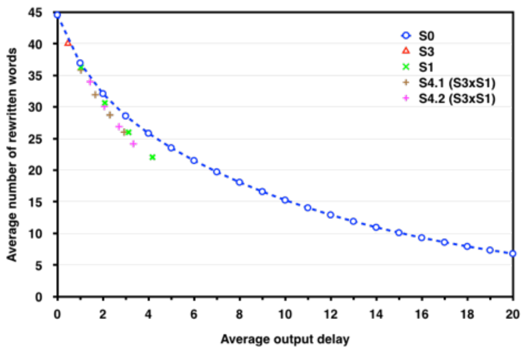
Figure 6.4: Average number of rewritten words plotted against average output delay for strategy S3. This figure is plotted using the four data from the TED talk 2.

6.5. Strategy 4 (S4.1, S4.2)

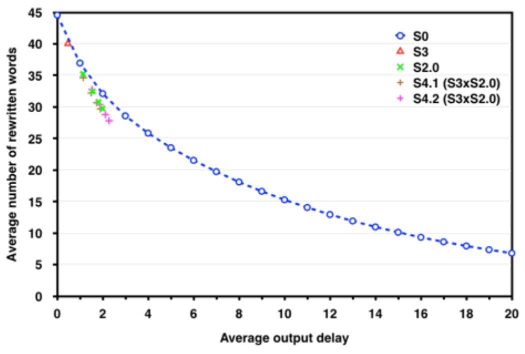
As shown above, for strategy S3.2, although the improvement to baseline strategy S0 is small, but the predictions still have about 65% accuracy. It seems that strategy S1 and strategy S2.x can more effectively reduce rewrites. Therefore, strategy S4, a combination of S3.2 with S1/S2.x, may get more improvement. The following figures only show the results based on data TED2.EN-DE.Long, the details of the results can be found in Table A.5 in Appendix.

A mixture between strategy S3 and S1 does not show applauded improvement compared to strategy S1 (Figure 6.5a). However, a mixture between strategy S3 and S2.x achieved some improvement compared to strategy S2.x (figs. 6.5b to 6.5d).

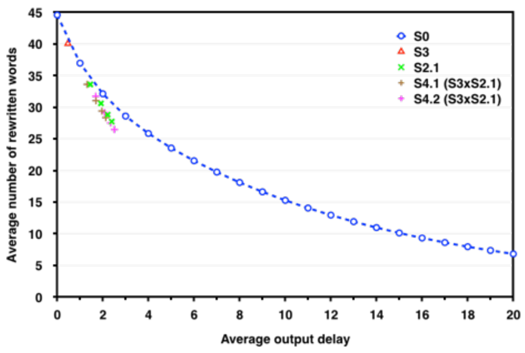
Comparing to strategy S2.x, strategy S4.1 almost do not reduce the average number of rewritten words while the average output delay is slightly decreased (as shown in figs. 6.5c and 6.5d). Comparing to strategy S2.x, the results of strategy S4.2 reduce the average number of rewritten words with very little (see figs. 6.5b and 6.5c) or without (see Figure 6.5d) increasing the output delay.



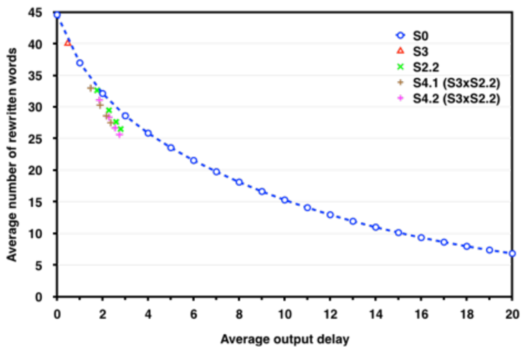
(a) Result of combining S3.2 and S1.



(b) Result of combining S3.2 and S2.0.



(c) Result of combining S3.2 and S2.1.



(d) Result of combining S3.2 and S2.2.

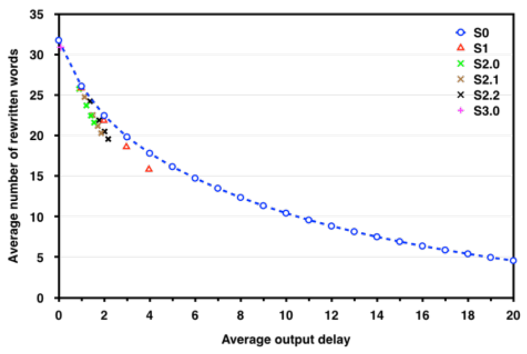
Figure 6.5: Average number of rewritten words plotted against average output delay for strategy S4. This figure is plotted using the EN-DE.Long data from the TED talk 2.

6.6. Comparison of different strategies

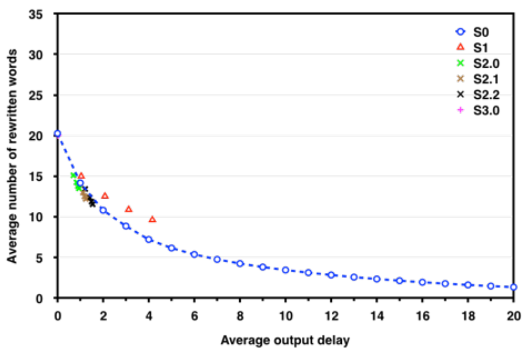
In order to better compare the strategies tested above, the results of strategy S0, S1, S2 and S3 is shown in Figure 6.6a. The results using strategy S4.x slightly improve the results of strategy S2.x. For clearer visualization, results using strategy S4.x is not plotted in Figure 6.6a. Figure 6.6b shows the results using different strategies on another data, which is from a additional TED talk to the two previous TED talks in this thesis. This additional data is introduced to test the stability of the strategies.

Both Figure 6.6a and Figure 6.6b are based on data which is translated from English to German using short-range reordering. As shown in Figure 6.6b, the additional data has smaller average number of rewritten words than the test data described in section 5.1. When the outputs are "honestly" displayed to users, most of the outputs have less than 4 rewritten words.

It is shown in Figure 6.6 that strategy S1 depends on translation results. It shows improvement in Figure 6.6a, but in Figure 6.6b it appears to have a poor performance (compared to baseline strategy S0).



(a) Result based on data TED2.EN-DE.Short.



(b) Result based on an additional data.

Figure 6.6: A comparison of strategy S1, S2 and S3. Average number of rewritten words plotted against average output delay.

The performance of strategy S2.x is more stable comparing to strategy S1. When strategy S1 does not show any improvement with the additional data (Figure 6.6b), strategy S2.x consistently shows a slightly better performance.

The results of strategy S3.x indicate that the classification method to predict the number of rewritten words does not appear to be promising. Most of the predictions are 0, and a small part of them are 1.

Although the strategy S3.x is not effective to reduce the rewrite during display, the strategy S3.2 can be used to predict if an output has rewrite or not with an accuracy of about 65%. Based on this fact, the strategy S4.x which combines S3.2 with S2.x can get some improvement.

In general, all these strategies tested above are more effective to the data with more rewrites. For example, the results based on the data of long-range reordering are better than the results based on the data of short-range reordering.

7. Summary and outlook

Dynamic transcription is a good solution for low-latency speech translation. The current best translation can be displayed in an early stage to lower the latency, and the transcribed text and its translation will be updated later to improve the accuracy [8]. The outputs displayed to users will be updated until the whole sentence is spoken. In order to display more accurate information to users, the most direct display strategy is to hide some inaccurate information from display. This inaccurate information might be later updated.

The task of this thesis is to develop new display strategies to give a good balance between accuracy and information delay. In other words, my goal is to find a strategy to predict the number of words rewritten in output update and then hide these words, so that the number of rewritten words can be minimized at a low output delay during display.

Five different strategies are proposed and implemented. As baseline, a constant number of words is hidden for each output. In strategy S1, the number of hidden words increases with the increasing length of input. In strategy S2.x, the number of hidden words is determined by the number of rewritten words in the previous output. In strategy S3.x, the classification method is used to predict the number of hidden words. Strategy S4.x is a combination of strategy S3.2 and strategy S1/S2.x.

The experiments are based on the data of English-to-German and German-to-English translation. Strategy S1 is not stable. The results may or may not get improved compared to the baseline strategy S0 depending on data. Strategy S2.x gets an improvement over the baseline and offers more stable improvement compared to S1. The improvement using strategy S3.x is not satisfactory due to the fact that the number of rewritten words in each output can hardly be predicted by classification method. Strategy S4.x, the combination of S3.2 and S2.x, shows improvement compared to strategy S2.x.

The display strategies for dynamic transcription in speech translation plays a key role in the trade off between accuracy and latency. Ideally, if the rewrites of outputs can be perfectly predicted and hidden, the information displayed to users during the dynamic speech translation process will be accurate. This means that the accurate information can be displayed to users with the lowest information delay. It is important to users to get unconfusing translation in time. More stable and effective display strategies are still to be found for speech translation between English and German.

Future work may involve designing of strategies using the neural netwok, which may improve the classification method to better predict the number of rewritten words in each output. Display strategies which can be applied to other language pairs other than German and English is an important direction of research to extend the functionality of dynamic transcription speech translation.

Bibliography

[1] "Translation in the European Union – Facts and Figures 2013 - Terminology Coordination Unit [DGTRAD] - European Parliament." [Online]. Available: http://termcoord.eu/2014/01/translation-in-the-european-union-facts-and-figures-2013/

[2] "Speech Translation – KIT lecture Machine Translation." [Online]. Available: https://ilias.studium.kit.edu/goto.php?target=file_536059_download&client_id=produktiv

[3] "Spracherkennung–Wikipedia." [Online]. Available: https://de.wikipedia.org/wiki/Spracherkennung

[4] "Phrase Alignment – KIT lecture Machine Translation." [Online]. Available: https://ilias.studium.kit.edu/goto.php?target=file_536053_download&client_id=produktiv

[5] "Phrase-based SMT – KIT lecture Machine Translation." [Online]. Available: https://ilias.studium.kit.edu/goto.php?target=file_536056_download&client_id=produktiv

[6] "Official languages of the EU - European Commission." [Online]. Available: http://ec.europa.eu/education/official-languages-eu-0{_}en

[7] "EU Parliament makes cuts to translation budget – EURACTIV.com." [Online]. Available: http://www.euractiv.com/section/languages-culture/news/eu-parliament-makes-cuts-to-translation-budget/

[8] J. Niehues, T. Nguyen, E. Cho, T. Ha, and K. Kilgour, "Dynamic Transcription for Low-latency Speech Translation," *Interspeech*, 2016.

[9] E. Cho, C. Fügen, T. Hermann, K. Kilgour, M. Mediani, C. Mohr, J. Niehues, K. Rottmann, C. Saam, S. Stüker, and A. Waibel, "A real-world system for simultaneous translation of German lectures," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2013, pp. 3473–3477.

[10] M. Kolss, M. Wölfel, F. Kraft, J. Niehues, and M. Paulik, "Simultaneous German-English lecture translation." 2008.

[11] M. Müller, T. Nguyen, J. Niehues, E. Cho, and B. Krüger, "Lecture Translator Speech translation framework for simultaneous lecture translation," *NAACL HLT*, 2016.

[12] M. Müller, S. Fünfer, S. Stüker, and A. Waibel, "Evaluation of the KIT Lecture Translation System," *Language Resources and Evaluation*, 2016.

[13] X. Huang, A. Acero, H.-W. Hon, and R. Foreword By-Reddy, *Spoken language processing: A guide to theory, algorithm, and system development.* Prentice hall PTR, 2001.

[14] P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Computational linguistics*, vol. 19, no. 2, pp. 263–311, 1993.

[15] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1. IEEE, 1995, pp. 181–184.

[16] F. Stanley and J. Goodmani, "An empirical study of smoothing techniques for language modeling," *Computer Speech and Language*, vol. 13, pp. 359–394, 1999.

[17] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1.* Association for Computational Linguistics, 2003, pp. 48–54.

[18] E. Matusov, A. Mauser, and H. Ney, "Automatic sentence segmentation and punctuation prediction for spoken language translation." *IWSLT*, 2006.

[19] C. Fügen and M. Kolss, "The influence of utterance chunking on machine translation performance." *INTERSPEECH*, 2007.

[20] M. Kolss, S. Vogel, and A. Waibel, "Stream decoding for simultaneous spoken language translation." *INTERSPEECH*, 2008.

[21] M. Paulik, S. Rao, I. Lane, and S. Vogel, "Sentence segmentation and punctuation recovery for spoken language translation," *Acoustics, Speech and*, 2008.

[22] S. Bangalore, V. R. Sridhar, and P. Kolan, "Real-time incremental speech-to-speech translation of dialogs," *Proceedings of the*, 2012.

[23] V. Sridhar, J. Chen, S. Bangalore, and A. Ljolje, "Segmentation Strategies for Streaming Speech Translation." *HLT-NAACL*, 2013.

[24] M. Yarmohammadi, V. Sridhar, and S. Bangalore, "Incremental Segmentation and Decoding Strategies for Simultaneous Translation." *IJCNLP*, 2013.

[25] M. Siahbani, R. Seraj, and B. Sankaran, "Incremental translation using hierarchichal phrase-based translation system," *Workshop (SLT), 2014 . . .*, 2014.

[26] Y. Oda, G. Neubig, S. Sakti, T. Toda, and S. Nakamura, "Optimizing Segmentation Strategies for Simultaneous Speech Translation." *ACL (2)*, 2014.

[27] H. Shavarani and M. Siahbani, "Learning segmentations that balance latency versus quality in spoken language translation," *Proceedings of the*, 2015.

[28] A. G. II, H. He, J. Boyd-Graber, and J. Morgan, “Don’t Until the Final Verb Wait: Reinforcement Learning for Simultaneous Machine Translation.” *EMNLP*, 2014.

[29] Y. Oda, G. Neubig, S. Sakti, T. Toda, and S. Nakamura, “Syntax-based Simultaneous Translation through Prediction of Unseen Syntactic Constituents.” *ACL (1)*, 2015.

[30] H. He, J. Boyd-Graber, and J. Graber, “Interpretese vs. translationese: The uniqueness of human strategies in simultaneous interpretation,” *Proceedings of NAACL-*, 2016.

[31] A. II, N. Orita, and J. Boyd-Graber, “Incremental Prediction of Sentence-final Verbs: Humans versus Machines,” *CoNLL 2016*, 2016.

[32] M. Cutts, *Oxford guide to plain English*, 2013.

[33] S. Vogel, “SMT decoder dissected: Word reordering,” in *Natural Language Processing and Knowledge Engineering, 2003. Proceedings. 2003 International Conference on.* IEEE, 2003, pp. 561–566.

[34] K. Rottmann and S. Vogel, “Word reordering in statistical machine translation with a POS-based distortion model,” *Proc. of TMI*, pp. 171–180, 2007.

[35] J. Niehues and M. Kolss, “A POS-based model for long-range reorderings in SMT,” in *Proceedings of the Fourth Workshop on Statistical Machine Translation.* Association for Computational Linguistics, 2009, pp. 206–214.

[36] H. Schmid, “Probabilistic part-of speech tagging using decision trees,” in *New methods in language processing.* Routledge, 2013, p. 154.

A. Appendix

	DE-EN. Short		DE-EN. Long		EN-DE. Short		EN-DE. Long	
	average rewritten	average delay	average rewritten	average delay	average rewritten	average delay	average rewritten	average delay
	#	#	#	#	#	#	#	#
do nothing	**31.74**	**0.00**	**37.66**	**0.00**	**37.16**	**0.00**	**44.56**	**0.00**
perfect prediction	**0.00**	**2.08**	**0.00**	**2.46**	**0.00**	**2.30**	**0.00**	**2.76**
S1 (r = 1)	25.84	0.99	30.82	0.98	29.28	1.03	36.27	1.04
S1 (r = 2)	21.84	1.99	25.96	1.96	24.66	2.06	30.63	2.09
S1 (r = 3)	18.59	2.98	21.97	2.95	21.03	3.09	25.99	3.13
S1 (r = 4)	15.82	3.97	18.55	3.93	18.00	4.12	22.05	4.18
S2.0 (T = 5)	25.74	0.90	29.44	1.07	28.43	1.10	35.23	1.14
S2.0 (T = 10)	23.69	1.22	26.19	1.50	26.52	1.42	32.46	1.56
S2.0 (T = 15)	22.44	1.42	24.24	1.76	25.35	1.61	30.77	1.82
S2.0 (T = 20)	21.59	1.56	23.09	1.91	24.64	1.74	29.74	1.99
S2.1 (T = 5)	24.74	1.14	28.31	1.33	26.75	1.46	33.59	1.45
S2.1 (T = 10)	22.51	1.50	24.79	1.81	24.67	1.83	30.59	1.92
S2.1 (T = 15)	21.17	1.72	22.70	2.09	23.44	2.04	28.81	2.21
S2.1 (T = 20)	20.28	1.87	21.46	2.26	22.68	2.17	27.72	2.39
S2.2 (T = 5)	24.21	1.38	27.69	1.58	25.78	1.82	32.62	1.76
S2.2 (T = 10)	21.87	1.78	24.03	2.12	23.58	2.23	29.47	2.28
S2.2 (T = 15)	20.48	2.02	21.87	2.43	22.31	2.46	27.62	2.59
S2.2 (T = 20)	19.56	2.18	20.58	2.62	21.52	2.61	26.49	2.79

Table A.1: Results of S1 and S2. These results are based on data from TED talk 2.

	binary			mutilclass		
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**31.74**	**0.00**		**31.74**	**0.00**
right prediction		**0.00**	**2.08**		**0.00**	**2.08**
Classifier 1	62.97 %	31.71	0.01	62.64 %	31.66	0.01
Classifier 2	62.96 %	31.61	0.02	61.96 %	31.41	0.05
Classifier 3	62.98 %	31.74	0.00	62.98 %	31.74	0.00
Classifier 4	62.90 %	31.25	0.06	62.75 %	31.15	0.07
Classifier 5	63.07 %	31.38	0.03	63.07 %	31.38	0.04
Classifier 6	63.03 %	31.36	0.04	63.03 %	31.56	0.02
Classifier 7	62.93 %	31.61	0.03	62.42 %	31.62	0.02
Classifier 8	0.20 %	4.57	20.00	61.63 %	31.47	0.03
Classifier 9	0.26 %	5.82	19.29	61.01 %	31.35	0.05
Classifier 10	0.13 %	3.50	23.41	62.75 %	30.98	0.12
Classifier 11	63.05 %	31.30	0.04	62.69 %	31.68	0.01
Classifier 12	63.07 %	31.38	0.03	63.07 %	31.37	0.03
Classifier 13	62.96 %	30.96	0.10	62.86 %	30.88	0.09
Classifier 14	9.40 %	2.11	33.17	60.56 %	31.07	0.09
Classifier 15	0.32 %	8.14	13.00	61.70 %	31.10	0.08
Classifier 16	0.23 %	5.87	17.00	62.97 %	31.74	0.00

(a) Result based on TED2.DE-EN.Short.

	binary			mutilclass		
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**37.16**	**0.00**		**37.16**	**0.00**
right prediction		**0.00**	**2.30**		**0.00**	**2.30**
Classifier 1	50.33 %	37.00	0.02	49.80 %	36.84	0.04
Classifier 2	50.32 %	36.95	0.03	48.79 %	36.22	0.11
Classifier 3	50.39 %	37.15	0.00	50.38 %	37.13	0.00
Classifier 4	50.27 %	36.74	0.04	50.05 %	36.70	0.05
Classifier 5	50.54 %	36.84	0.03	50.51 %	36.83	0.03
Classifier 6	51.32 %	35.89	0.10	51.27 %	32.81	0.39
Classifier 7	50.15 %	36.85	0.09	46.32 %	35.93	0.15
Classifier 8	50.29 %	36.96	0.03	50.01 %	36.85	0.03
Classifier 9	0.20 %	5.13	19.00	49.07 %	36.31	0.09
Classifier 10	15.39 %	24.19	5.76	48.83 %	35.53	0.18
Classifier 11	50.55 %	36.13	0.10	48.24 %	36.17	0.10
Classifier 12	50.58 %	36.73	0.04	50.55 %	36.72	0.04
Classifier 13	50.36 %	36.31	0.08	50.08 %	36.31	0.09
Classifier 14	49.44 %	36.30	0.56	48.85 %	33.05	0.39
Classifier 15	0.17 %	2.90	26.00	48.49 %	32.84	0.41
Classifier 16	0.87 %	14.28	7.00	45.67 %	35.66	0.17

(b) Result based on TED2.EN-DE.Short.

	binary			mutilclass		
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**44.56**	**0.00**		**44.56**	**0.00**
right prediction		**0.00**	**2.76**		**0.00**	**2.76**
Classifier 1	52.85 %	44.18	0.04	49.53 %	43.25	0.13
Classifier 2	52.82 %	44.08	0.05	50.63 %	43.64	0.09
Classifier 3	52.87 %	44.56	0.00	52.87 %	44.56	0.00
Classifier 4	52.70 %	43.96	0.06	52.50 %	43.14	0.16
Classifier 5	53.00 %	44.23	0.03	52.97 %	44.21	0.03
Classifier 6	53.03 %	44.24	0.03	52.95 %	44.25	0.03
Classifier 7	52.70 %	44.00	0.12	51.82 %	43.88	0.07
Classifier 8	52.73 %	43.43	0.24	51.29 %	43.71	0.08
Classifier 9	1.82 %	23.54	5.12	49.93 %	43.04	0.15
Classifier 10	0.91 %	19.03	8.47	51.79 %	43.31	0.13
Classifier 11	52.92 %	43.82	0.07	51.16 %	43.78	0.07
Classifier 12	53.00 %	44.23	0.03	52.87 %	44.56	0.00
Classifier 13	52.81 %	43.44	0.12	52.55 %	43.13	0.15
Classifier 14	9.39 %	1.42	45.28	50.09 %	43.49	0.11
Classifier 15	0.58 %	11.91	13.00	51.77 %	43.06	0.14
Classifier 16	0.59 %	13.38	11.35	52.75 %	44.53	0.01

(c) Result based on TED2.EN-DE.Long.

Table A.2: Results of S3.0 prediction using both binomial and multiclass classification for all classifiers.

		binary			mutilclass	
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**31.74**	**0.00**		**31.74**	**0.00**
right prediction		**26.08**	**0.37**		**26.08**	**0.37**
Classifier 1	63.27 %	31.21	0.07	63.06 %	31.56	0.02
Classifier 2	63.16 %	31.18	0.07	62.98 %	31.74	0.00
Classifier 3	62.98 %	31.74	0.00	62.98 %	31.74	0.00
Classifier 4	63.80 %	31.31	0.05	63.84 %	31.28	0.05
Classifier 5	64.62 %	31.09	0.07	64.37 %	31.38	0.03
Classifier 6	64.78 %	30.85	0.10	64.71 %	31.02	0.08
Classifier 7	63.19 %	31.20	0.07	62.98 %	31.74	0.00
Classifier 8	63.31 %	31.21	0.07	62.98 %	31.74	0.00
Classifier 9	63.93 %	31.09	0.08	63.83 %	30.65	0.13
Classifier 10	63.80 %	31.31	0.05	63.80 %	31.31	0.05
Classifier 11	65.04 %	30.59	0.13	62.98 %	31.74	0.00
Classifier 12	64.62 %	31.08	0.07	62.97 %	31.74	0.00
Classifier 13	65.13 %	30.84	0.10	63.77 %	31.37	0.04
Classifier 14	65.06 %	30.58	0.13	62.98 %	31.74	0.00
Classifier 15	65.26 %	30.51	0.14	62.99 %	31.74	0.00
Classifier 16	64.49 %	30.09	0.20	62.99 %	31.74	0.00

(a) Result based on TED2.DE-EN.Short.

		binary			mutilclass	
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**37.16**	**0.00**		**37.16**	**0.00**
right prediction		**29.15**	**0.50**		**29.15**	**0.50**
Classifier 1	56.36 %	33.98	0.33	56.36 %	33.98	0.33
Classifier 2	57.05 %	33.76	0.35	57.23 %	33.53	0.38
Classifier 3	57.14 %	34.44	0.27	57.14 %	34.44	0.27
Classifier 4	55.26 %	35.39	0.17	54.20 %	35.87	0.12
Classifier 5	64.50 %	32.62	0.42	64.15 %	33.07	0.37
Classifier 6	64.27 %	32.36	0.46	64.35 %	32.42	0.45
Classifier 7	57.38 %	33.49	0.38	49.61 %	29.15	1.00
Classifier 8	58.12 %	33.54	0.37	57.87 %	33.72	0.35
Classifier 9	57.14 %	34.09	0.31	57.14 %	34.09	0.31
Classifier 10	59.36 %	33.66	0.34	59.36 %	33.66	0.34
Classifier 11	64.93 %	32.23	0.47	52.95 %	29.33	0.94
Classifier 12	64.42 %	32.67	0.42	64.03 %	32.89	0.39
Classifier 13	65.43 %	32.49	0.43	63.58 %	33.27	0.35
Classifier 14	65.02 %	32.13	0.48	52.95 %	29.33	0.94
Classifier 15	65.13 %	32.47	0.43	62.22 %	31.27	0.61
Classifier 16	65.24 %	32.12	0.48	52.33 %	29.29	0.96

(b) Result based on TED2.EN-DE.Short.

		binary			mutilclass	
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**44.56**	**0.00**		**44.56**	**0.00**
right prediction		**36.95**	**0.47**		**36.95**	**0.47**
Classifier 1	58.13 %	41.66	0.31	50.80 %	37.16	0.94
Classifier 2	58.53 %	41.44	0.33	49.89 %	37.11	0.95
Classifier 3	55.37 %	42.31	0.25	55.37 %	42.31	0.25
Classifier 4	56.62 %	43.10	0.14	56.62 %	43.10	0.14
Classifier 5	60.30 %	41.18	0.34	60.22 %	41.02	0.37
Classifier 6	60.92 %	40.57	0.41	60.27 %	40.54	0.42
Classifier 7	58.64 %	41.29	0.35	47.13 %	36.95	1.00
Classifier 8	58.92 %	41.35	0.34	58.11 %	41.24	0.36
Classifier 9	58.87 %	41.78	0.29	58.77 %	41.84	0.28
Classifier 10	57.37 %	42.71	0.18	57.37 %	42.71	0.18
Classifier 11	61.95 %	40.22	0.45	60.30 %	41.34	0.33
Classifier 12	60.52 %	40.89	0.38	59.53 %	41.51	0.31
Classifier 13	61.72 %	40.64	0.40	60.06 %	41.30	0.33
Classifier 14	61.85 %	40.03	0.47	60.45 %	40.59	0.42
Classifier 15	62.21 %	40.33	0.43	58.87 %	41.68	0.30
Classifier 16	62.30 %	40.04	0.47	52.54 %	37.39	0.89

(c) Result based on TED2.EN-DE.Long.

Table A.3: Results of S3.1 prediction using both binomial and multiclass classification for all classifiers.

		binary			mutilclass	
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**31.74**	**0.00**		**31.74**	**0.00**
right prediction		**22.45**	**0.61**		**22.45**	**0.61**
Classifier 1	62.81 %	31.20	0.07	62.98 %	31.74	0.00
Classifier 2	62.75 %	31.14	0.08	62.78 %	31.08	0.09
Classifier 3	62.98 %	31.74	0.00	62.88 %	31.72	0.00
Classifier 4	63.52 %	31.10	0.07	63.48 %	31.22	0.06
Classifier 5	63.28 %	31.15	0.07	63.27 %	31.18	0.07
Classifier 6	63.30 %	31.15	0.07	63.30 %	31.18	0.06
Classifier 7	62.79 %	31.17	0.08	62.98 %	31.74	0.00
Classifier 8	62.81 %	31.16	0.08	62.77 %	31.14	0.08
Classifier 9	63.41 %	30.89	0.10	62.69 %	31.10	0.09
Classifier 10	63.47 %	31.09	0.08	63.48 %	31.10	0.07
Classifier 11	63.65 %	30.41	0.16	62.98 %	31.74	0.00
Classifier 12	63.19 %	31.18	0.07	63.04 %	31.70	0.00
Classifier 13	64.14 %	30.59	0.12	63.49 %	31.21	0.06
Classifier 14	63.44 %	30.40	0.17	63.01 %	31.51	0.03
Classifier 15	64.07 %	30.16	0.18	62.71 %	30.85	0.12
Classifier 16	63.97 %	30.26	0.17	62.98 %	31.74	0.00

(a) Result based on TED2.DE-EN.Short.

		binary			mutilclass	
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**37.16**	**0.00**		**37.16**	**0.00**
right prediction		**24.48**	**0.79**		**24.48**	**0.79**
Classifier 1	51.05 %	35.12	0.24	50.43 %	37.12	0.00
Classifier 2	51.15 %	34.76	0.28	50.39 %	37.16	0.00
Classifier 3	50.73 %	36.91	0.02	50.73 %	36.91	0.02
Classifier 4	51.65 %	35.94	0.12	51.55 %	36.06	0.11
Classifier 5	54.05 %	31.07	0.68	53.93 %	31.14	0.67
Classifier 6	55.62 %	32.29	0.49	55.43 %	32.65	0.46
Classifier 7	51.23 %	34.51	0.31	50.54 %	36.59	0.07
Classifier 8	51.90 %	34.66	0.28	50.61 %	35.22	0.24
Classifier 9	52.00 %	35.04	0.23	52.03 %	35.02	0.23
Classifier 10	52.18 %	35.52	0.17	52.20 %	35.54	0.16
Classifier 11	55.52 %	31.01	0.66	54.51 %	31.00	0.68
Classifier 12	53.96 %	31.13	0.67	53.83 %	31.55	0.63
Classifier 13	54.55 %	30.73	0.71	54.28 %	31.04	0.68
Classifier 14	55.53 %	31.02	0.65	52.84 %	34.47	0.28
Classifier 15	54.33 %	30.57	0.74	52.70 %	34.85	0.24
Classifier 16	56.97 %	31.39	0.57	51.89 %	34.08	0.35

(b) Result based on TED2.EN-DE.Short.

	binary			mutilclass		
	accuracy	average rewritten	average delay	accuracy	average rewritten	average delay
	%	#	#	%	#	#
do nothing		**44.56**	**0.00**		**44.56**	**0.00**
right prediction		**32.11**	**0.77**		**32.11**	**0.77**
Classifier 1	54.25 %	42.41	0.24	54.25 %	42.41	0.24
Classifier 2	54.46 %	41.99	0.29	54.46 %	41.99	0.29
Classifier 3	53.10 %	44.33	0.02	53.10 %	44.33	0.02
Classifier 4	54.65 %	43.07	0.15	54.65 %	43.07	0.15
Classifier 5	53.39 %	39.72	0.59	53.32 %	39.74	0.59
Classifier 6	57.27 %	40.99	0.35	56.56 %	42.11	0.23
Classifier 7	54.47 %	41.80	0.31	52.87 %	44.56	0.00
Classifier 8	54.70 %	42.03	0.28	53.97 %	42.20	0.27
Classifier 9	55.37 %	42.10	0.26	55.31 %	42.19	0.25
Classifier 10	54.79 %	42.79	0.18	54.68 %	42.86	0.17
Classifier 11	56.62 %	39.92	0.50	54.41 %	42.39	0.24
Classifier 12	53.39 %	39.74	0.59	53.33 %	39.75	0.59
Classifier 13	54.50 %	39.04	0.65	54.78 %	43.11	0.14
Classifier 14	56.74 %	39.84	0.50	54.10 %	42.40	0.24
Classifier 15	57.37 %	39.59	0.52	55.33 %	42.01	0.27
Classifier 16	58.45 %	39.57	0.50	55.20 %	41.58	0.32

(c) Result based on TED2.EN-DE.Long.

Table A.4: Results of S3.2 prediction using both binomial and multiclass classification for all classifiers.

	S4.1		S4.2	
	average rewritten	average delay	average rewritten	average delay
	#	#	#	#
do nothing	**44.56**	**0.00**	**44.56**	**0.00**
S1 (r = 1)	35.83	1.04	33.98	1.44
S1 (r = 2)	31.93	1.67	30.08	2.07
S1 (r = 3)	28.73	2.31	26.88	2.71
S1 (r = 4)	26.02	2.94	24.17	3.34
S2.0 (T = 5)	34.64	1.15	32.75	1.53
S2.0 (T = 10)	32.23	1.50	30.34	1.88
S2.0 (T = 15)	30.67	1.74	28.79	2.12
S2.0 (T = 20)	29.69	1.90	27.81	2.28
S2.1 (T = 5)	33.60	1.31	31.72	1.69
S2.1 (T = 10)	31.02	1.70	29.14	2.08
S2.1 (T = 15)	29.39	1.96	27.50	2.34
S2.1 (T = 20)	28.35	2.13	26.47	2.51
S2.2 (T = 5)	32.96	1.48	31.07	1.86
S2.2 (T = 10)	30.27	1.89	28.38	2.28
S2.2 (T = 15)	28.57	2.17	26.68	2.55
S2.2 (T = 20)	27.50	2.36	25.62	2.74

Table A.5: Results of S4. These results are based on data from TED2.EN-DE.Long.