Karlsruhe Institute of Technology

Cognitive Systems Lab

SMT-based Text Generation for Code-Switching Language Models

Diploma Thesis
at the
Cognitive Systems Lab
Department of Informatics
Karlsruhe Institute of Technology

of
cand. inform.
Fabian Blaicher

Supervisors:

Prof. Dr.-Ing. Tanja Schultz
Dipl.-Inform. Tim Schlippe

This thesis was in part carried out at
Nanyang Technological University, Singapore
under guidance of
Prof. Haizhou Li and Prof. Chng Eng Siong

Begin: 1st Juli 2010
End: 9th May 2011

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Karlsruhe, den 9. Mai 2011

Acknowledgements

This thesis could not have been written without the support of a number of people to whom I want to express my gratitude. First of all, I would like to thank my supervisor Prof. Tanja Schultz for her helpful advise and the support for the exchange at Nanyang Technological University (NTU) in Singapore. I also would like to thank Asst. Prof. Chng Eng Siong and Prof. Haizhou Li for the discussions and the support for the stay at NTU. Further, I want to thank Dipl-Inform. Tim Schlippe, for the long and frequent discussions which helped me steering through my thesis. For the good cooperation and the help in Singapore I especially want to thank Dau-Cheng Lyu Ph.D. I am also grateful to my proofreaders Dr. Robert Geisberger, Dipl-Inform. Dominic Heger and Dipl-Inform. Jan Gebhardt. I would also like to thank Hu Rong who showed much patience during my work. And finally, I would like to thank my parents who supported me during the whole course of my studies.

Abstract

In this thesis we will present a statistical machine translation (SMT) - based text generation approach for building code-switching (CS) language models (LMs). CS is the use of more than one language during a spoken discourse. The Mandarin-English SEAME corpus [30] is the basis for our experiments. This corpus contains challenging spontaneous speech. Initially, we analyze the recognition of CS 2-grams and find that only 25% are recognized correctly. Motivated by an analysis of the differences in recognition performance using Chinese words and Chinese characters, we investigate the impact of Chinese word segmentation: The perplexity drops by 32% with a segmentation generated by the Stanford Chinese Word Segmenter, but the MER does not improve. For our baseline LM, we interpolate a monolingual English NIST LM, a monolingual Chinese GALE LM, and the SEAME training transcriptions (*train*) LM. We create class-based LMs with automatically-induced classes and with part-of-speech (POS) classes, which are commonly used if little training data is available. Both methods show no improvement when compared to the 3-gram *train* LM. Our main approach *Search and Replace* (S&R) for SMT-based text generation extracts monolingual English segments from *train*, translates these segments to Chinese and searches the translations in a large monolingual Chinese text corpus. If the Chinese translations are found, they are replaced with their English counterparts from SEAME. Analogously, we create a CS text based on an English corpus. Our approach combines CS data with monolingual texts with the aim to build robust CS LMs. Instead of randomly translating segments, we insert segments from SEAME into the monolingual English and Chinese texts. The inserted SEAME segments represent actual code-switching. The mix error rate (MER) of the S&R LM is 0.35% worse than the one of our baseline, but we increase the CS 2-gram coverage from 36% to 57% and the CS 3-gram coverage from 8% to 11%. Further, we follow existing methods [50] to improve the probability distribution of multilingual LMs and apply them to our S&R approach. However, they are not successful. For further improvement of the S&R approach, we limit the replacements to frequent segments in *train*. Then, we incorporate context information: Found segments are only replaced if the word preceding the segment is a trigger word or has trigger POS tag. A trigger is a token found in *train* preceding a CS. Additionally, we investigate another probability improvement approach which limits the number of replacements per segment. These limits are derived from the segment frequencies in *train*. The combination of the last approach with the trigger POS approach results in a 0.62% relative mix error rate improvement. A human evaluation of the improved S&R text suggests that the translations do not fit to the context. We use the best segment translation from the 5-best translations to improve the CS context-fit of S&R and the S&R improvement with trigger POS and segment frequency limits. By using the 5-best translations the ASR shows a relative improvement of 0.45% over the baseline which is lower than the 0.62% improvement of the LM built from the 1st-best translations. Finally, we analyze monolingual corpora and find out that the use of additional texts which are similar in style to SEAME for S&R with trigger POS and frequency limits leads to a 0.18% relative MER improvement compared to a new baseline build from the same texts.

Zusammenfassung

In dieser Arbeit befassen wir uns mit Sprachmodellen für Code-Switching (CS). CS ist die Verwendung von mehreren Sprachen oder Stilen innerhalb eines Satzes. Wir stellen eine Methode basierend auf statistischer Übersetzung zur Generierung von CS Sprachmodellen vor. Dabei benutzen wir den SEAME Textcorpus, der chinesisch-englischen CS Text enthält. Dieser Corpus beinhaltet anspruchsvolle, spontane Sprache für automatische Spracherkennung. Eine anfängliche Analyse zeigt, dass 25% von den CS 2-Grammen korrekt erkannt werden. Darauf folgend, untersuchen wir die chinesische Wortsegmentierung. Das Ergebnis zeigt, dass die Verwendung einer Segmentierung generiert mit dem Stanford Chinese Word Segmenter eine Perplexitätsverbesserung von 32% zur Folge hat. Die Nutzung dieser Segmentierung zeigt keinen positiven Einfluss auf die Mix Error Rate (MER). Als Baseline interpolieren wir 3-Gramm Sprachmodelle basierend auf dem SEAME Trainingstext (*train*), und einem monolingualen englischen und einem monolingualen chinesischen Text. Wir untersuchen ob klassenbasierte Sprachmodelle Verbesserung zeigen. Dabei prüfen wir die Eignung von automatisch-hergeleiteten Klassen und Klassen aus Wortarten. Beide Modelle zeigen keine Verbesserung. Unser Hauptansatz zur Erstellung von CS Sprachmodellen ist die *Suche und Ersetze* Vorgehensweise. Dabei werden aus SEAME monolinguale, englische Wortsequenzen extrahiert und nach Chinesisch übersetzt. Diese Wortsequenzen werden in chinesischen Texten gesucht, und, falls gefunden, mit den extrahierten englischen Wortsequenzen ersetzt. Analog werden Texte basierend auf englischen Textcorpora erstellt. Aus diesen beiden neuen CS Texten werden Sprachmodelle generiert und mit dem *train* Sprachmodell interpoliert. Durch die Verwendung von SEAME Wortsequenzen stellen wir sicher, dass wir natürliches CS generieren. Die MER ist 0.35% schlechter als bei unserem Baseline, aber die CS 2-Gramm Abdeckung steigt von 36% auf 57% und die CS 3-Gramm Abdeckung steigt von 8% auf 11%. Wir folgen existierende Methoden [50], um CS N-Gramm-Wahrscheinlichkeiten zu verbessern, die aber keinen positiven Effekt zeigen. Deswegen schlagen wir neue Methoden vor, um CS N-Gramme zielgerichteter zu erstellen. Zuerst ersetzen wir nur noch häufige Wortsequenzen. Danach verwenden wir Kontextinformationen um die CS Generierung zu verbessern. Es werden nur Wortsequenzen nach einem *Trigger* ersetzt. Ein *Trigger* ist ein Wort oder eine Wortart, welche(s) vor einem CS Auftritt. Zusätzlich präsentieren wir eine weitere Vorgehensweise: Dabei wird eine maximale Anzahl von Ersetzungen vorgegeben, die von SEAME Transkriptionen abgeleitet wurde. Eine Kombination der letzten Vorgehensweise und der Ersetzungseinschränkung durch Wortartentrigger (TrigPOS&TF) zeigt eine relative Verbesserung von 0.62%. Unsere Umfrage bezüglich der Natürlichkeit der generierten Texte zeigt, dass künstlich erzeugte CS Sequenzen bezüglich des Kontextes teilweise falsch übersetzt werden. Deswegen verwenden wir die Beste der 5-Best Übersetzungen um den CS Context-Fit von TrigPOS&TF zu verbessern. Das Ergebnis ist 0.45% besser als das Baseline, aber schlechter als TrigPOS&TF, welches nur die erstbeste Übersetzung verwendet. Im letzten Versuch zeigen wir, dass die Verwendung von mehreren Basistexten mit ähnlichem Stil wie SEAME, zu einer weiteren relativen Verbesserung von 0.18% zu einem Baseline mit den gleichen monolingualen Texten führt.

Contents

1. Introduction

1.1 Motivation

Automatic speech recognition (ASR) systems have been deployed to everyday applications. They are used in popular devices like cell phones or car navigation systems. Although current speech recognition systems are robust in recognizing monolingual speech, they are currently not able to cover all phenomenons of spoken conversations. A typical example is code-switching, which occurs in multilingual communities in many regions. Code-switching is the use of more than one language during a discourse. Most of the current ASR systems lack the support of code-switching. The focus regions for our work are Singapore and Malaysia in South-East Asia. Singapore is interesting for code-switching research as it has a multicultural and multilingual community.

The most important aspect of code-switching speech recognition is the correct prediction of the code-switches. This is challenging since people can switch their language at any point of the sentence. In addition, only limited audio and text data which contains code-switching is currently available. We use a new code-switching speech corpus collected from researchers at Nanyang Technological University. This corpus contains Mandarin-English code-switching speech from South-East Asia (SEAME).

An automatic speech recognition system for code-switching needs to be able to handle multiple vocabularies and pronunciations of different languages.

1.2 Main Contributions

We present several approaches to improve the code-switching support of automatic speech recognizers. The focus of our work lies on language modeling.

Initially, we provide an analysis of the SEAME corpus and different factors impacting the ASR recognition performance. Later, we show issues with the Chinese text segmentation.
Furthermore, we have a close look at differences in code-switching behavior among speakers and between topics. Since we do not know the topics of which the speaker talks, we assign topics based on 20 important words selected with TF-IDF.

We evaluate two kinds of class-based language models: Automatically-induced class language models and part-of-speech language models. In the presence of little available training texts, class-based language models provide robust estimates for word sequences.

Our main focus is the estimation of robust code-switching n-grams since only little data for computing reliable code-switch n-gram estimates exists. Code-switching n-grams contain at least one code-switch. Monolingual n-grams in contrast can be easily and successfully estimated from huge available monolingual texts. We combine knowledge from the code-switching corpus SEAME with available monolingual texts. Therefore, we extract monolingual segments from the SEAME corpus, translate them, search them in monolingual texts, and replace them with the original segments. The modified texts, called SMT-based or artificial texts, are used to build code-switching language models. We present multiple refinements of our main approach. The advantage of our work is the leveraging of available monolingual information with little code-switching data and statistical machine translation to generate code-switching language models.

1.3 Outline

The remaining chapters of this thesis are organized as follows:

Chapter 2 introduces fundamental knowledge of code-switching as well as basics of automatic speech recognition and statistical machine translation.

Chapter 3 is devoted to related literature on speech recognition and code-switching.

Chapter 4 presents an overview of the SEAME corpus, together with an initial analysis of SEAME and a TF-IDF-based topic analysis.

Chapter 5 considers multiple approaches to statistical machine translation-based language model generation for code-switching language models. We present different approaches to improve the generation of code-switch n-grams and their probabilities.

Chapter 6 concludes our work with a discussion of the analyzed approaches and experiments.

Chapter 7 gives an outlook and recommendations for future work.

2. Fundamentals

This chapter introduces the fundamentals of speech recognition and statistical machine translation in a nutshell. The reader more interested in details is relegated to the mentioned literature.

2.1 Code-Switching

Code-Switching (CS) can be defined as the "usage of more than one language, variety, or style by a speaker within an utterance or discourse " [5]. A popular example was given by Poplack [37]:

Sometimes I'll start a sentence in Spanish **y termino en español** [...and I finish in Spanish]

Code-Switching is the use of more than one language during a speech exchange. This phenomenon can be found in many multilingual communities, e.g. in the United States [13] with Spanish and English, in Hong Kong [28] with Cantonese and English or in Singapore [30] with Mandarin and English. Linguists have analyzed code-switching for a long time and comprehensive literature is available [18, 28, 39].
There are special distinctions of code-switching which need a brief discussion. The use of one single word or named entity of a different language is often called word-borrowing [19] or insertion [32], e.g. using "San Francisco" in a Chinese sentence. The literature [28] sometimes distinguishes between code-switching and code-mixing, whereas code-switching describes an inter-sentential change of the language between sentences and code-mixing describes the intra-sentential change of the language. We do not follow this distinction as it is not used consistently in literature. Our work focuses on intra-sentential code-switching, which occurs frequently in our SEAME corpus.

2.2 Automatic Speech Recognition

Automatic Speech Recognition (ASR) converts an unknown speech signal to a series of words $W = w_1 w_2 ... w_n$. Speech is recorded, preprocessed, sampled, and transformed into a sequence of feature vectors and then given to the decoder.

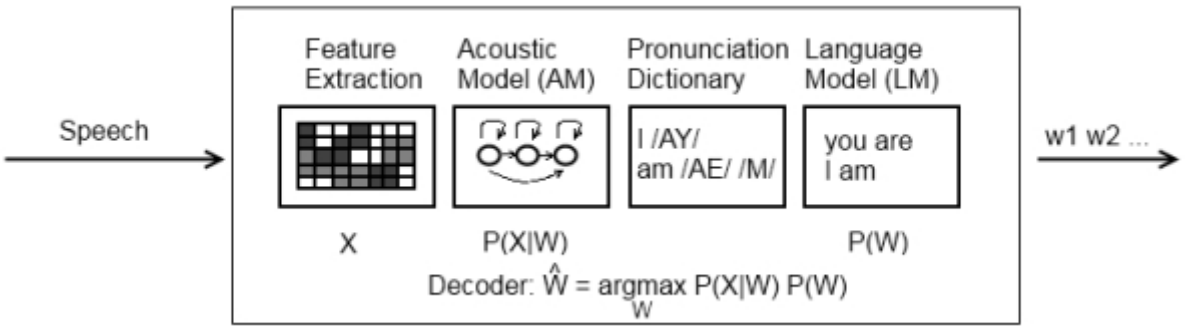

Figure 2.1: Automatic Speech Recognition [42].

An overview of the speech recognition process is depicted in Figure 2.1. The main components of an ASR system are the *acoustic model* P(X|W), the *language model* P(W), the *dictionary*, and the *decoder*. The ASR tries to find the word sequence $\hat{W}$ which fits best to the input data. This is achieved by using the Bayes formula to maximize:

$$\hat{W} = \underset{W}{\operatorname{argmax}}\, P(W|X) = \underset{W}{\operatorname{argmax}}\, \frac{P(X|W)P(W)}{P(X)} \tag{2.1}$$

Acoustic Model

The *acoustic model* (AM) P(X|W) explains the likelihood of a sound unit X given input speech. The sound units used in ASR are usually phonemes. Each phoneme is modeled by one hidden Markov model, see Figure 2.2. Multiple HMMs can be joined together to model words or sentences. For example, the phoneme sequence of "hello" can be **/hh/ /ah/ /l/ /ow/**. The HMMs have a series of usually three states which are connected in a left-to-right topology. The emissions of the HMM states are feature vectors and the emission probabilities are modeled by continuous probability density functions, and usually defined by multivariate Gaussian mixture density functions [53]. Rabiner [38] gives a detailed discussion of HMMs in speech recognition. The model is called *hidden* Markov model since the state sequence is not known, only the sequence of features is observed. All parameters of an HMM are trained from real speech data and the corresponding transcriptions.

Language Model

The *language model* contains probability estimates $\hat{P}(W)$ of word sequences W. This type of language models is called statistical language model [40], but there exist also grammar-based [22] language models. In this work, we focus on statistical n-gram language models. Language models contain n-grams, which are probability estimates

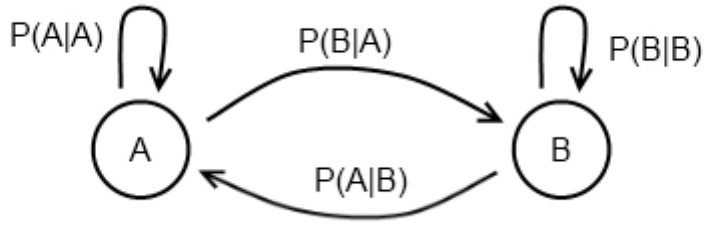

Figure 2.2: An exemplary hidden Markov model.

of word sequences, or differently of words given their predecessors. Language model n-grams are estimated from a training text, with:

$$\hat{P}(w_i|w_{i-N+1}...w_{i-1}) = \frac{\mathrm{C}(w_{i-N+1},...,w_i)}{\mathrm{C}(w_{i-N+1},...,w_{i-1})} \tag{2.2}$$

$\mathrm{C}(w_1,...,w_k)$ is the number of word sequence $w_1,...,w_k$ occurrences in the text. Theoretically, one would use an indefinite history of words to estimate w_i, but this is not feasible. In practice most n-gram models use a history of three or four words dependent on the available training data. The problem of using this strategy is that there can be many valid n-grams which are not present in the training data. The solution to this problem is achieved by using language model smoothing techniques. Smoothing redistributes the probability mass, so that low probabilities such as 0 are adjusted upwards and higher probabilities are adjusted downwards [11].

Apart from simple word-based language models, there are two types of class-based language models. The first one consists of linguistic motivated classes, while the second one consists of automatically derived classes. The class n-gram probabilities are given [4, 17] by:

$$P(w_i|w_{i-N+1}...w_{i-1}) = P(X_i|X_{i-N+1}...X_{i-1})P(w_i|X_i) \tag{2.3}$$

where X is a class. The advantage of class-based language models is that they need less text data for reliable word sequence estimates. This is due to the lower number of classes in comparison to the number of words in the dictionary.

The performance of language models is measured by the language model *perplexity* (PPL). It can be considered as the average number of equally likely words following a given word. The perplexity of the language model P on the development text $\mathrm{W} = w_1...w_n$ is defined as [54]:

$$PPL(P,W) = 2^{\hat{H}(W)} = \hat{P}(w_1,...,w_n)^{-\frac{1}{n}} \tag{2.4}$$

with:

$$\hat{H} = -\frac{1}{n} log_2 \hat{P}(w_1,...,w_n) \tag{2.5}$$

Where $\hat{P}(w_1,...,w_n)$ is the probability estimate assigned to the word sequence $(w_1,...,w_n)$ by the language model P. A lower perplexity indicates a better prediction of the language model. Although the ASR performance is dependent of various components, it has been shown that lower perplexity values correlate with lower word error rates [24].

Another measure for language models is the n-gram coverage. This measure tells us how many n-grams in the evaluation data are covered by the n-grams from the used language model.
The out-of-vocabulary (OOV) rate measures the average percentage of words not covered by the language model. These words are mapped to the unknown word token <UNK> and assigned a single probability during the language model estimation. A high OOV rate has a negative impact on speech recognition, because the word is substituted with a similar sounding word. Furthermore, a recognition error spreads to neighboring words and degrades the overall recognition performance [1].
The n-gram language models in this work are estimated with the Stanford Research Institute Language Modeling Toolkit (SRILM) [46].

Pronunciation Dictionary

The pronunciation dictionary contains a list of word entries with one or more pronunciations represented by sequences of sound units, e.g. phonemes. Here is an exemplary entry of our dictionary in Janus format [12]:

{word} {{w_ENG WB} er_ENG {d_ENG WB}}

Each entry consists of a word in curly brackets, followed by a space, and the pronunciation, again in curly brackets. The first and last phonemes are tagged with curly brackets and a *WB*. WB stands for word boundary. The dictionary we use also supports multiple pronunciation variants for words, e.g. for dialects or accents. For words with multiple pronunciations, a pronunciation index is added in brackets. To distinguish similar phones in different languages, language identifiers are added to each phoneme. The dictionary is the link between the acoustic and the language model.

Decoder

The decoder searches for the best word sequence hypothesis $\hat{W}$ fitting to the speech signal. This means finding the word sequence $\hat{W}$ from all possible W which maximizes the P(W|X), given the acoustic model, language model and dictionary [53].

The performance of an ASR is measured with the *word error rate* (WER). This measure is computed after an alignment of the reference sentence and the hypothesis from the ASR. The definition is:

$$WER = \frac{(\text{INS} + \text{SUB} + \text{DEL})}{\#\text{words in reference}} \cdot 100\% \tag{2.6}$$

Hereby INS is the number of word insertions, SUB is the number of word substitutions and DEL is the number of word deletions. As Chinese does not have the concept of words, we compute the error rate based on Chinese characters and English words. This error rate is called measure mix error rate (MER) [7].

Another performance measure is the word accuracy (WA), which is defined as:

$$WA = 1 - WER \tag{2.7}$$

The weight of the language model in the Janus ASR system can be adjusted by a language model weight parameter (LW). We analyzed LW $\in$ {28,30,32,34,36,38,40,42,44,46,48,50}. A high LW parameter indicates a higher relevance of the language model. In addition, our ASR applies a word penalty (LP) to long words in the recognition process. We analyzed LP $\in$ {-8,-6,-4,-2,0,2,4,6,8,10}. For all experiments the MERs of the ASR are computed with all used LW and LP combinations. The final LW/LP combination is the one which minimized the MER on the development text adn should be used for the decoding of the test data. The LW and LP parameters are included in the Bayes formula 2.1 as:

$$\underset{W}{\operatorname{argmax}}\, P(X|W)P(W)^{LW}LP^{|W|} \tag{2.8}$$

We use the Janus Recognition Toolkit (JRTK) [12] for decoding. This automatic speech recognition system was developed at the Karlsruhe Institute of Technology and Carnegie Mellon University, USA.

2.3 Statistical Machine Translation

A *statistical machine translation* (SMT) system translates a sentence from one language to another language. Figure 2.3 shows the process with the main components of our system: A translation model, a language model and a decoder. Most systems also use a word reordering model (distortion model). Given the translation model

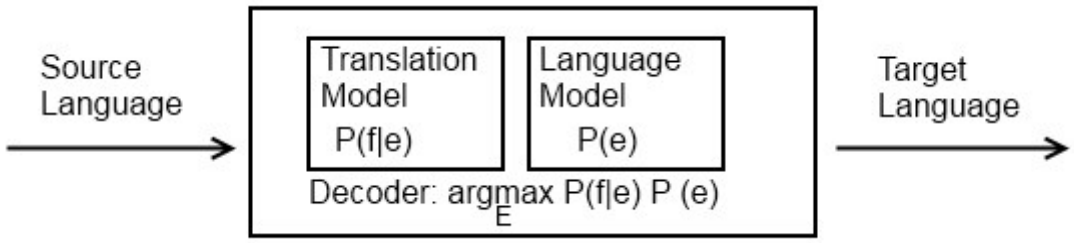

Figure 2.3: Statistical machine translation system.

and the language model, the decoder finds the best translation $\hat{e}$ for a given sentence f by applying the Bayes theorem by computing $\hat{e} = \underset{E}{\operatorname{argmax}}\, P(f|e) \cdot P(e)$. E is the set of all possible translations. The language model $P(e)$ has the same function and structure as mentioned in ASR Section 2.2 and assigns a probability to a given phrase. The translation model $P(f|e)$ assigns a probability to a phrase tuple (f, e). The model used in this work is a bilingual phrase table with probabilities of the possible phrase pairs. The probabilities are trained from large bilingual text corpora.

The performance of SMT systems could be evaluated by humans, but this is expensive. Therefore, systems are evaluated with the bilingual evaluation understudy (BLEU) [35] score. The idea is to compare the output translation of the SMT system with one or more good human translations.

The weights of the SMT system input features are important, in our case the weights of the language model and the phrase table. Och [33] introduced the *minimum error*

rate training (MERT) to find optimal feature weights. This is done by an algorithm for the efficient training of an unsmoothed error count.

3. Related Work

Code-switching has been studied by linguists for decades and has recently received more attention from the speech recognition community. All important speech corpora known to the author were collected during the last 6 years. We give an overview of code-switching speech corpora and previous language modeling approaches for code-switching. Many of the proposed ASR systems use classes and part-of-speech tags in their language models to cope with the data shortage.

Currently there are only few corpora with code-switching speech. Researchers conducted their experiments with their own data, and there was no effort to compare the approaches on a single data set. We give an overview of code-switching speech corpora, which we consider important:

CUMIX (2005) [8]: This data collection contains 17 hours of read Cantonese-English code-switching speech from 80 speakers. The texts originate mostly from newsgroup articles, online diaries and transcriptions from radio phone-in programs [20]. In 93% of the code-switching utterances, one or two English words are embedded into a Cantonese sentence.

Mandarin-English I (2010) [52]: A Mandarin-English speech corpus of 45.2 hours recorded in 2006. It consists of lectures held at the National University Taiwan. The prevailing language of the single speaker is Mandarin, which accounts for 85% of all words.

Mandarin-English II (2009) [6]: Recorded speech of one female speaker during different meetings. It consists of about 3 hours of spontaneous speech. Mandarin constitutes for 78% of the corpus.

Mandarin-Hakka (2010) [48]: The audio data of this corpus contains read speech in monolingual Hakka, monolingual Mandarin and code-switching Hakka-Mandarin. The texts are mainly written in Hakka, which accounts for 75% of the words. Articles from the internet, books, magazines and an information retrieval database (CIRB) were used as text sources for Hakka and Mandarin.

Mandarin-Taiwanese (2006) [29]: The training set of this read speech corpus is composed of monolingual Taiwanese and monolingual Mandarin speech data from 100 speakers. Whereas the testing set consists of read code-switching speech from 16 speakers. In each Mandarin sentence is at least one Taiwanese word embedded.

SEAME (2010) [30]: A Mandarin-English corpus with 31 hours from South-East Asia with 59% Chinese words. This corpus was used as the main code-switching speech data in our experiments and will be discussed in the next chapter.

Spanglish (2007) [13]: 40 minutes of spontaneous Spanish-English code-switching speech from 3 speakers. The average number of code-switches per sentence is 0.3. The author notes that the corpus is small and relegates to a planned collection of more ressources.

The SEAME corpus, which is the target of our research, is much larger than most corpora: it is the second largest corpus. Its word share of the dominating language (Mandarin) is relatively low (59%) compared to other corpora, such as Mandarin-English I (85%) and II (78%) or Mandarin-Hakka (75%). This means SEAME contains more accented speech than other corpora.

There has been much research about multilingual speech processing. Especially during the last years, many researchers published approaches how to handle code-switching speech. We give a short review of important research in this field.

One particularly interesting paper regarding language modeling was published by Fügen et al. [14]. They proposed a language modeling method for combining multiple monolingual language models into one multilingual language model. An additional meta-layer combined multiple language model scores. The new language model was able to handle code-switching at utterance level. This technique made it possible to build multilingual speech recognizers which allowed seamless language switches at utterance level.

Code-switching does not only occur between distinct languages, but also between dialects. Lyu et al. [29] developed an ASR system for code-switching speech of the dialects Taiwanese and Mandarin. Therefore, they compared a multi-pass decoder with a one-pass system on the Mandarin-Taiwanese corpus. Their multi-pass system consisted of language boundary detection (LBD), language identification (LID) as well as a monolingual Taiwanese and a monolingual Mandarin ASR system. Their one-pass system used a three-layer network consisting of an acoustic model layer, a lexical layer and a grammar layer. The grammar layer was represented by a 2-gram language model. This approach took advantage of the characteristics of both dialects sharing the same orthography system. Thus, the number of entries in the pronunciation dictionary was twice as large. The one-pass system with a vocabulary of 20K words showed a character error rate of 20.02% in comparison to the multi-pass system with a character error rate of 31.76%. Since we work with two languages with distinct words, we cannot transfer their approach with one language model and enhanced dictionary to our setup. Nevertheless, we also follow a single-pass approach without LBD, LID or monolingual recognizers.
Tsai et al. [48] built an ASR system for code-switching speech of the dialects Hakka

and Mandarin. Part of their corpus was manually translated and tagged with part-of-speech (POS) tags. Their focus was to build a reliable language model in the light of lacking text data. For this goal, they proposed an improved formula for computing n-grams:

$$P(w_i|w_{i-1}) = \begin{cases} P_{BASELINE}(w_i|w_{i-1}), & \text{count}(w_i|w_{i-1}) \geq k \\ \alpha_1(w_{i-1}) * P_{POS}(w_i|w_{i-1}) & \\ +\alpha_2(w_{i-1}) * P_{TRANS}(w_i|w_{i-1}), & \text{otherwise} \end{cases} \tag{3.1}$$

where $P_{BASELINE}$ was a 2-gram language model, P_{POS} was a language model using part-of-speech information and P_{TRANS} was a language model using translation information. Both P_{POS} and P_{TRANS} supported code-switching in both directions and therefore support unseen code-switches. Their best result showed an improvement of 5.3% relative to 62.12% character accuracy on read speech. This was attributed to the use of part-of-speech 2-grams in their language model. These part-of-speech 2-grams were estimated from a Hakka-Mandarin text and a monolingual Mandarin text.

Chan et. al [8] collected the CUMIX corpus for their experiments. Initially, they trained their acoustic model with the monolingual Cantonese CUSENT corpus and the monolingual English corpus TIMIT. By using CUSENT and CUMIX instead, the recognition performance increased by 4% to 55.67%, measured by combined English word and Chinese syllable accuracy.

Later, Chan et al. [9] built an improved Cantonese-English code-mixing ASR system. They analyzed four different 3-gram language model types. For those language models, they additionally collected texts from newspapers and the web. The first language model was a Cantonese language model, where all English words were considered as out-of-vocabulary. In the second code-switching language model all English words shared the same probability. The third was a class-based language model, consisting of 13 classes. The classes were assigned based on part-of-speech information or meaning, e.g. companies. The fourth translation-based language model translated English words to their Cantonese equivalent if available; otherwise the classes from the 3rd language model were used. They evaluated the language model performance with the phonetic-to-text (PTT) conversion rate and the class-based language model performed best with a PTT of 91.52%. Their final experiment with the complete ASR and the class-based language model achieved a Chinese syllable and English word accuracy of 56.04%. The authors demonstrated that including part-of-speech information improved the recognition performance. Their translation language model was constrained by the lack of translations for all words. The errors in the recognition were attributed to the acoustic model, which had problems with phone changes, syllable fusion and Cantonese accents of English words. They also showed that integrating LBD improved the recognition of the code-switch words.

Cao et al. [7] focused in their 2010 published paper on language modeling for Cantonese-English. Their work was based on the CUSENT and CUMIX corpora. For their first language model they translated parts of the n-grams of a Cantonese language model and merged the resulting n-grams with the monolingual n-grams. The translations were based on a manually created translation dictionary. This language model increased the accuracy by 2.8%, measured by English words and Cantonese characters. The second language model was based on clustering English words by their meaning and part-of-speech. This resulted in another 0.4% accuracy

improvement. The combination of their language modeling approaches revealed an absolute increase of 3.2% to the final accuracy of 73.7%. The better recognition of English words accounted mostly for the increase. The researchers stated that more code-switch n-grams in the language model led to better perplexity as well as recognition performance. Similar to [9] the use of part-of-speech features enhanced the language model performance. However, the creation of code-switching n-grams with a manual translation dictionary was even more successful.
Solorio [13] built a Spanish-English code-switching language model from the Spanglish corpus with Good-Turing discounting. The transcriptions had a vocabulary of 1.5K words. Their language model evaluation showed a perplexity of 49.4.
In their follow up work [45], Solorio et al. developed a Spanglish part-of-speech tagger on the base of monolingual Spanish and English part-of-speech taggers. They applied different heuristics and machine learning approaches to combine the monolingual part-of-speech tagger results. The best performance was achieved with a support vector machine, resulting in a 93.48% mean accuracy.
Later, they analyzed code-switch prediction in [44]. Their best prediction system showed an F1-score of 0.24. The author noted, that the F1-score of the prediction was acceptable, but the F1-score was not necessarily the ideal measure. Therefore, they conducted a human evaluation to see if automatically generated code-switching sentences were grammatical and natural sounding: The code-switch prediction led to sentences nearly as good as real code-switch sentences and much better than randomly generated code-switches. Regarding the prediction features, the authors considered Beginning-Inside-Outside information not useful. For the successful prediction they prefered the following features: word form, language, part-of-speech tag, and detailed information (probability, lemma) from the part-of-speech tagger.
Burgmer [6] worked with the Mandarin-English II corpus. In his work he combined an English and a Chinese part-of-speech tagger to a Mandarin-English code-switching part-of-speech tagger. Further, he analyzed multiple features, e.g. part-of-speech tags or words, for predicting code-switching points. His system achieved an F1-score of 0.45, 0.60 for switches from English and 0.23 for switches from Chinese. In summary, the author considered code-switch prediction feasible, but pointed to the issue that code-switching was voluntary. This means it is questionable if perfect prediction could be achieved. His work indicated that part-of-speech tags were better for predicting Chinese-to-English switches, than single words. In contrast, he showed that English-to-Chinese switches were better predicted by single words. He also stated that Chinese accents of English words posed additional challenges. We refer to his work in Section 5.5.3.2.
Gebhardt [16] used a factored language model (FLM) to build an ASR system for the Mandarin-English II corpus. First, he investigated how code-switching could be predicted with current machine learning methods. Second, he built a two-pass recognizer: the first pass used the CMU - InterACT 2008 Mandarin Transcription System [21] and in the second pass the FLM applied to rescore the hypotheses. The FLM included the following four factors: language tags, part-of-speech tags, code-switch probabilities and words. His system showed an improvement of 1.6% relative to 59.8% MER on the evaluation set. His result suggested that code-switch probability was not helpful as an FLM factor. More interestingly, his research indicated that the language and part-of-speech of a word gave important information about a following code-switch and improved the FLM performance.
Yeh et al. [52] used word-based 2-and 3-gram language models, an automatically-

induced class-based 2-gram and part-of-speech language models. The Chinese part-of-speech tags were assigned with the Academia Sinica part-of-speech tagger. The English part-of-speech tags were assigned from a dictionary. Their first part-of-speech language models used different classes for English and Chinese. The second language model contained 10 classes for both English and Chinese together. The classes were merged manually. The third language model clustered the Chinese words by part-of-speech and each English word was treated as a class. The fourth language model (LM4) was similar to the third, except that only Chinese words with inadequate counts were clustered and the language model was only used for back-off. Initial experiments with these language models showed all small improvements. In their final experiment with their best acoustic model they interpolated an out-of-domain language model with their best previous language models. The result showed that the part-of-speech based language model achieved the highest improvement of 0.7% relative to 79% MER. The class-based approach did not show any improvement, whereas RFLM showed little improvement. The success of the part-of-speech language model was attributed to the good recognition of Mandarin. This is interesting, since the recognition of English words should be improved. Similar to their first part-of-speech approach, we use language-dependent part-of-speech tags. Our automatically-induced class-based language model is built in the same way as their class-based language model.
Lyu built a Mandarin-English code-switching speech corpus (SEAME)[30]. This corpus was used as the main code-switching speech data in our experiments and is discussed in the next chapter.

The research indicated that little text data was a problem for language modeling. Embedded words with accents of the matrix language seemed to be a issue, too.
The authors from [44] and [6] showed that code-switch predictions feasible. Part-of-speech tags and words were successfully used as features for the prediction. Human evaluation supported the theory that code-switches could be predicted by machine learning methods.
With regard to language modeling for code-switching it could be seen that automatically derived class-based language models as well as various part-of-speech based language models showed improvement. The translation of parts of a monolingual language model suggested to be a step in the right direction. Especially the generation of many code-switch n-grams seemed to be important.
A drawback of the proposed approaches was the manual translation and manual part-of-speech tagging of code-switch texts.

4. Corpora and Corpora Analysis

In this chapter, we introduce the text and speech corpora which form the base for our experiments. This overview of the corpora is followed by an initial error analysis. Thereafter, we discuss issues with the Chinese segmentation and present a TF-IDF-based topic analysis.

4.1 Corpora

Our automatic speech recognizer is trained and evaluated on a Mandarin-English code-switching speech corpus from South-East Asia (SEAME). Additionally, we use monolingual Chinese text from the Global Autonomous Language Exploitation (GALE) Project and English text from the National Institute of Standards and Technology (NIST) for our SMT-based code-switching language models. The SEAME training set transcriptions are abbreviated with "*train*" in this thesis.

The SEAME corpus [30] contains 31 hours of spontaneously spoken Mandarin and English code-switching speech. It is recorded in interview and conversation settings. The participants are either from Universiti Sains Malaysia, Malaysia or Nanyang Technological University, Singapore. The English transcriptions are all in lower case. Chinese is in Hanzi Unicode encoding.

Since the speakers come from a multicultural environment, they also use words from other languages, e.g. Japanese or Malay. These foreign words are tagged with the surrounding character "#", e.g. #sashimi# which is Japanese for raw fish. Proper nouns are tagged with the surrounding character "%", e.g. %jurong%, which is a quarter of Singapore. Many proper nouns refer to places in Singapore. Particles are tagged with [ ], e.g. [ma]. Most of the particles are interjections, for example *mh*, *ah* or *yeah*. But there is also the Chinese particle *ma*, which indicates a question in the Chinese language. Table 4.1 presents an overview of the tagged words.

For the recordings of the SEAME corpus, the participants receive a list of possible topics for their discussion. The topics involve university, family, friends, sports and other spare time activities. Unfortunately, this topic information of the conversations was not archived.

	Tag	Token count	Type count
Foreign words	#	820	337
Proper nouns	%	224	102
Particles	[ ]	23K	110

Table 4.1: Words with special tags counted on the complete corpus.

The SEAME corpus has 101 speakers and is divided in the training (train), development (dev) and testing (test) set. Table 4.2 shows the number of hours of speech and the number of speakers for the complete SEAME and the 3 subsets and SEAME Part2. The table also presents the percentages of monolingual English, Chinese and code-switching sentences. Most of the sentences contain code-switching. The development set contains more code-switching sentences than the test set. The latter has less English and more Chinese sentences.
At the end of our work, we obtain further transcriptions from the same data collection. These are called SEAME (Part2). The shares of monolingual English, Chinese and code-switching sentences of Part 2 text are similar to those of the *train* text.

	#hours	#speakers	#sentences	%CS	%English	%Chinese
SEAME	31	101	26K	76%	8%	16%
Train	26	83	22K	76%	9%	15%
Dev	2.5	8	2K	82%	2%	16%
Test	2.5	10	2K	77%	1%	22%
Part2	30	74	28K	73%	10%	17%

Table 4.2: The different sets of SEAME.

Table 4.3 illustrates the word distribution between the sets of spoken language. The distribution among languages is rather unbalanced with $\frac{1}{3}$ English words and $\frac{2}{3}$ Chinese words. In the development and test sets there are more Chinese words than in the training set. This is suboptimal, as balanced sets are more suitable for training and evaluation. It can be seen that approximately 8% of words belong to the word group *other* which consists of proper nouns, particles and foreign words.

	#words	%English	%Chinese	%Other
SEAME	313K	33%	59%	8%
Train	263K	35%	57%	8%
Dev	24K	28%	63%	9%
Test	26K	23%	70%	8%
Part2	342K	33%	59%	8%

Table 4.3: Word distribution in the SEAME corpus.

An analysis of the monolingual segment lengths shows that they are generally very short. A monolingual segment is the longest sequence of consecutive words in one language. Figure 4.1 shows that 71% of the English segments are only one or two

words long. The Chinese segments in contrast are longer, see Figure 4.2. This observation of many short embedded words in a code-switch corpus is similar to observations in other code-switch studies [8, 44].

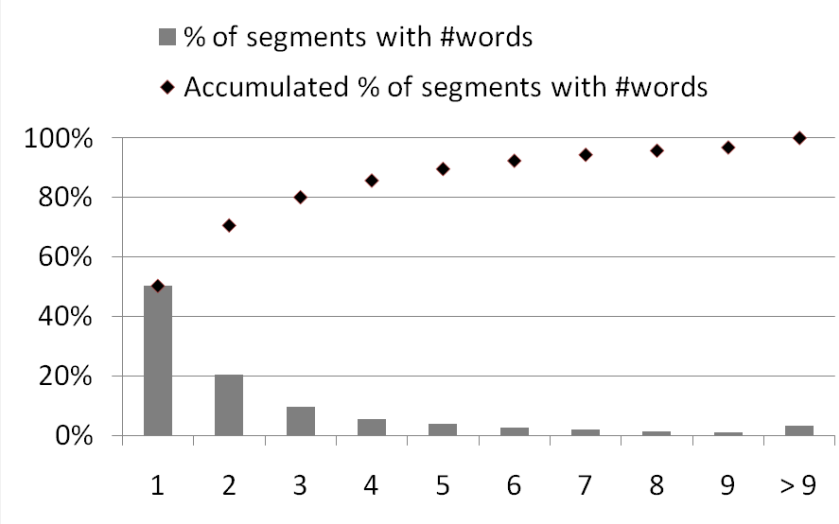

Figure 4.1: English segment lengths distribution. The x-axis shows the number of words per segment.

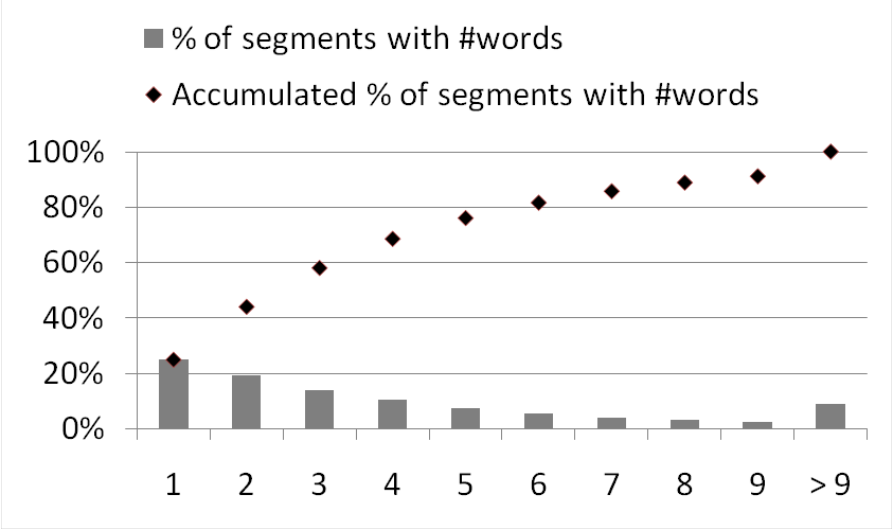

Figure 4.2: Chinese segment lengths distribution. The x-axis shows the number of words per segment.

The average number of code-switches (∅CS rate) in *train* is 2.4 times per sentence, with an average sentence length of 12 words. Figure 4.3 shows the descending sorted ∅CS rates for all 83 speakers in the training set. There is a large range between the ∅CS rates among the speakers: The highest ∅CS rate of a speaker is 4.3, while the lowest is 0.82.

We build the SMT-based language models from artificial code-switch texts. These are generated by automatically translating parts of subsets of the monolingual English NIST [15] and Chinese GALE [34] text corpora. Only a subset of the GALE text is available to us. Since we want balanced English and Chinese texts, we create a subset of the English NIST text with a similar amount of sentences as the GALE

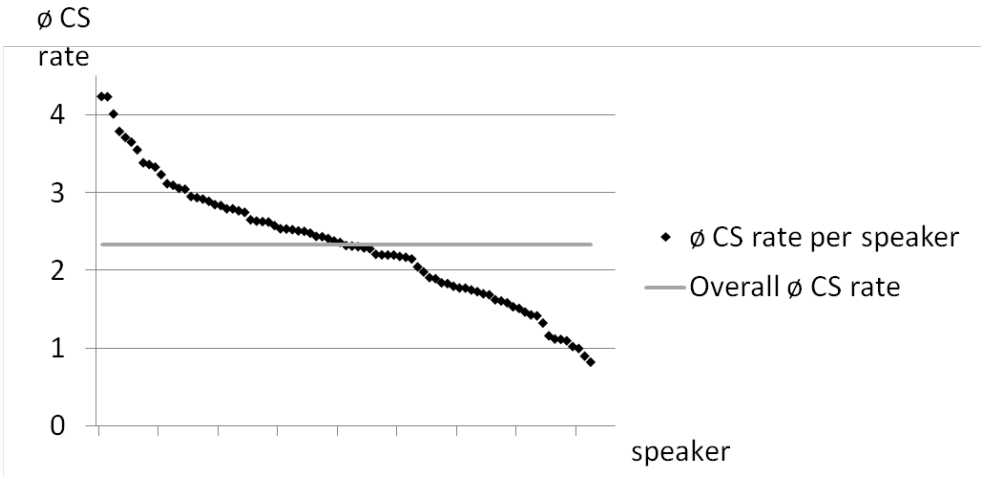

Figure 4.3: Descending sorted ∅CS rates of the speakers in the training set.

subset. Therefore, we select each 24,000th sentence from the larger English NIST text and concatenate them to the English NIST subset.
Working with a subset of the texts has an additional advantage, that the SMT-based text generation time decreases from several weeks to 1 day on our machines. We use an Intel Core2 Quad CPU Q9400 at 2.66 GHZ with 8GB RAM for our text generation experiments.

	#sentences	#words
English NIST	8.4M	255M
English NIST subset	345K	9.2M
Chinese GALE subset	395K	9.6M

Table 4.4: English NIST and Chinese GALE texts used in our experiments.

The NIST text originates from newswire, whereas the GALE text comes from collected web text, broadcast news and broadcast conversations.

4.2 Initial Analysis

Our ASR system with the SEAME training transcriptions (*train*) language model has a mix error rate of 50.50% on the development set. Therefore, we analyze how different factors affect the mix error rate: First, we investigate how different speakers affect the recognition performance. Second, we compare the recognition of tagged foreign words, proper nouns and particles to Chinese characters and English words. Finally, we focus on the baseline recognition performance of the code-switching parts.

Speakers can exhibit diverse speaking behaviors, which may result in significant variations in recognition performance. For example, people with varying backgrounds speak with different speeds or use different dialects. Figure 4.4 presents the mix error rates of the 8 speakers in the development set. The mix error rates vary between

39% and 72% among the speakers. But not all speakers are equally important, since some of them account for a larger share in the corpus. For example, speaker 3, who shows the worst recognition performance, is with 74 (4%) sentences responsible for only a relatively small part of the corpus.

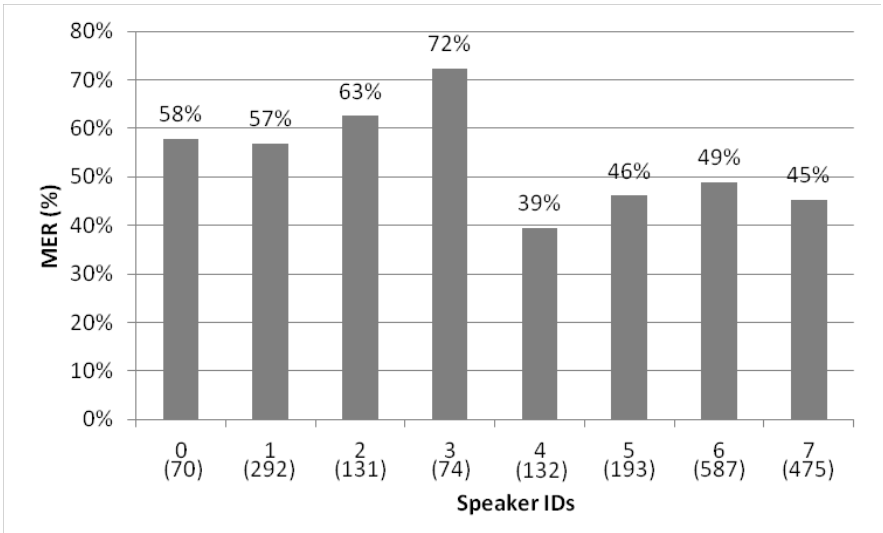

Figure 4.4: MER over the development set speakers. The number of sentences is in brackets.

Foreign words, particles and proper nouns are tagged in the corpus. Since the acoustic model and the language model are trained with little data for these words, we assume that they might be recognized wrong. Figure 4.5 presents the number of correctly and wrongly recognized tagged words. There are only 39 foreign words and 30 proper nouns in the development set. But most of them are not recognized correctly. From the 1955 particles, only 33% are recognized properly. The recognition rates for non-tagged Chinese characters and English words are higher than for the tagged words. 62% of Chinese is recognized correctly compared to 40% of the English words.

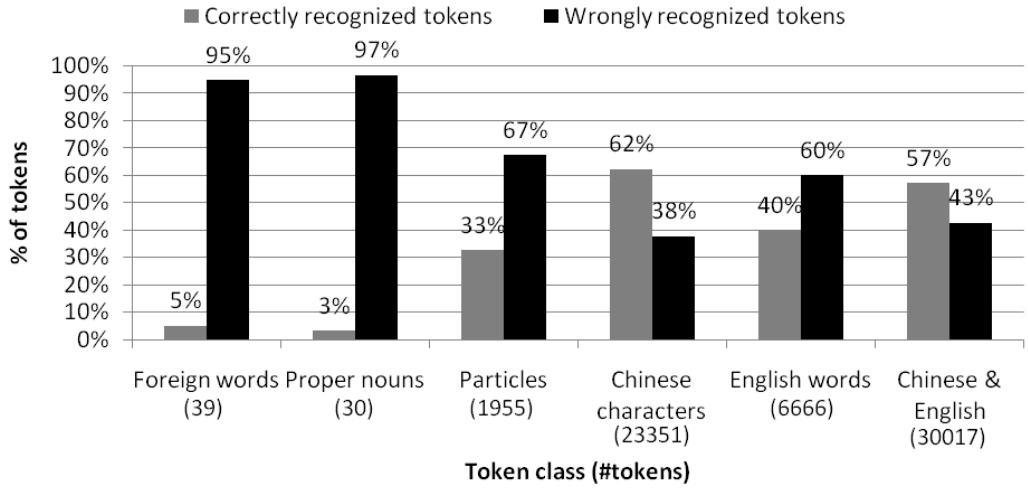

Figure 4.5: Percentage of correctly or wrongly recognized tokens.

A high number of code-switches per sentence (∅CS) can be another likely driver of a high mix error rate. The issue is that code-switches are estimated from little data and the acoustic model and the language model have to cope properly with the language switches. Since we want to analyze and improve the recognition of code-switching, we look at the two words covering a code-switch. We name these two words *code-switch 2-grams*. Figure 4.6 shows the number of correctly and wrongly recognized *code-switch 2-grams*. Only 25% of these are recognized correctly, 75% are wrong. This result indicates that code-switching is a big problem for our recognizer. Since there are many code-switches in our data, we need to improve the recognition performance of code-switching.

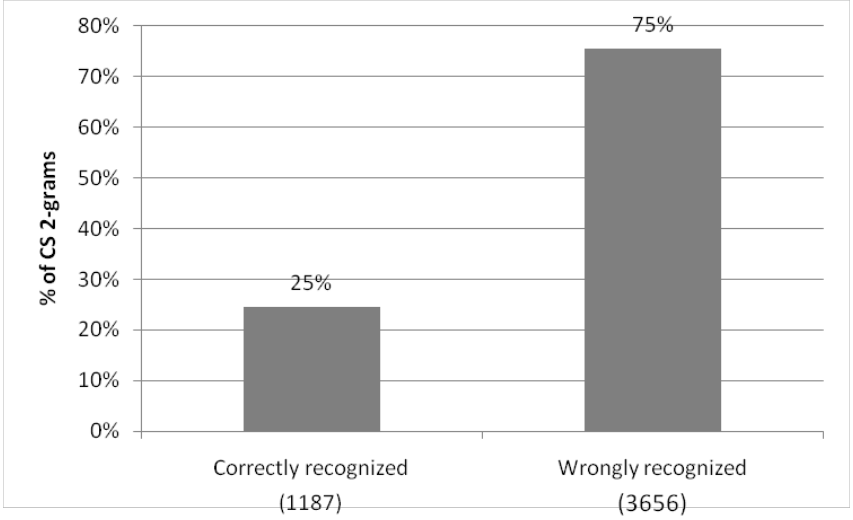

Figure 4.6: Comparison of correctly and wrongly recognized CS 2-grams.

Conclusion

Our investigation shows that different speakers show different mix error rates in the development set. A deeper look into the hypotheses from the recognizer reveals, that Chinese is recognized better than English or tagged words. The 39 foreign words and 30 proper nouns are recognized poorly and only a third of the 1955 particles are recognized correctly. The analysis of *code-switch 2-grams* shows that only 25% are recognized correctly. Since there are many code-switches in our data, our goal is to improve their recognition.

4.3 Chinese Segmentation Issues

The SEAME *train* language model has a perplexity of 552 on the development text. One explanation for such a relatively high perplexity is the Chinese word segmentation: The ASR scoring results show a difference between error rates computed on Chinese words and Chinese characters. The error rate on the development text of 50.50% evaluated on Chinese characters and English words is lower than the mix error rate of 61.73% with Chinese ”words” and English words. The SEAME corpus has a Chinese vocabulary of 5.4K words, which are composed of 2K different characters. Chinese has no concept of word segmentation. Characters are written without any

spaces. Since there are no official segmentation rules, there are different possibilities how to segment text. Inconsistent segmentation leads to poor n-gram estimates and finally to a high perplexity or mix error rate. Using n-grams of single characters is not the solution as this means the n-grams are estimated from less context.

	Original segmentation	Stanford segmentation	Change in %
SEAME	5,386	6,270	16%
Train	4,806	5,477	14%
Dev	1,559	1,444	-7%
Test	1,808	1,797	-1%

Table 4.5: Chinese vocabulary sizes (#words) with the original and Stanford segmentation.

To analyze the impact of a homogeneous segmentation, we remove all spaces between Chinese characters in the code-switching training and development transcriptions. Then, we use the Stanford Chinese Word Segmenter [47] for resegmentation. The tagger offers two segmentation models: The first one is trained on the Chinese Treebank segmentation and the second one is trained on a Beijing University segmentation. The last model provides results with smaller vocabulary sizes and lower OOV rates [10]. We use the Beijing University model as it provides us with a lower perplexity.

Table 4.5 shows that the vocabulary size of Chinese words increases from 5.4K to 6.3K. The perplexity of the resegmented *train* language model on the resegmented development text drops by 32%, see Table 4.6. At the same time, the OOV rises slightly from 3.86% to 3.99%. The drop in perplexity seems promising for ASR improvement. But the issue is that the new language model could not immediately be used with our speech recognizer, since the available pronunciation dictionary does not cover 1.4K of the newly segmented Chinese words. Therefore, it is necessary to create new pronunciations.

	OOV	PPL
Original	3.86%	552
Resegmented	3.99%	376
Change	-4%	32%

Table 4.6: Comparison of the two segmentations.

Manual pronunciation generation is expensive. Therefore, we decide to generate the pronunciations automatically. An analysis of our baseline dictionary shows that 99% of the Chinese multi-character words have pronunciations which are built from combined pronunciations of the existing single-character pronunciations. The problem is that many single-character words have multiple pronunciations, such that a generation of all combinations would lead to an explosion of possible pronunciations. We generate new pronunciations for new multiple-character words based on the first

single-character pronunciation found in the dictionary for each of the characters in the Chinese words.

With the resegmented language model and the new generated pronunciations evaluated on the resegmented development set, the ASR result is with -0.8% relative worse than the baseline system. The reason may be wrong pronunciations originating from the higher OOV rate.

4.4 TF-IDF Analysis

Language model performance is strongly dependent on the relation between the training text and the evaluation text. Language models which have the same characteristics, e.g. domain and style, as the evaluation data are mostly superior to a general or out-of-domain model. It has been shown that including texts, which are relevant for the evaluation data, improves language model perplexity [25, 26].

The use of Code-switching can be domain-dependent [43]. For example in Hong Kong, English-Cantonese code-switching is widely spread in computer discourse, business discourse, food discourse and fashion or showbusiness discourse. Therefore, we believe that topic-dependent language models might model the n-grams for each topic better.

We already mentioned that the topic information of the conversations were not noted. But we know that the conversations have different topics. Therefore, we analyze the topics and build different *topic* language models. With these topic language models, we try to improve the language model perplexity. We search the transcriptions for topic words for each speaker and use texts exclusively from *train*, which have the same topic, to build *topic*-specific language models.

We use *term frequency-inverse document frequency* (TF-IDF) [23] to find important topic words from the speaker transcriptions (documents) which are used as the basis of our topic assignment. The idea is to find words which occur frequently in one document, but less in other documents. These words represent the relevant topics in the document. The term-frequency (TF) is defined as follows:

$$tf_{i,j} = \frac{\text{\#words j in document i}}{\text{total \#words in document i}} \tag{4.1}$$

The inverse document frequency (IDF) is defined as:

$$idf_j = log\left(\frac{N}{\text{\#documents with word j}}\right) \tag{4.2}$$

where N is the total number of documents. The TF-IDF weight is therefore:

$$w_{i,j} = tf_{i,j} \cdot idf_j = \frac{\text{\#words j in document i}}{\text{total \#words in document i}} \cdot log\left(\frac{N}{\text{\#documents with word j}}\right) \tag{4.3}$$

Words which appear only in few documents have a high IDF. Whereas, words which occur in many documents have a low IDF [41]. Based on TF-IDF, we compute the

weights of all words from all speaker transcriptions. After that, we manually assign one topic to each speaker based on the 20 words with the highest weights.

For 26 of the 83 *train* speakers we cannot assign topics based on these words, since they contain too differing or too general words. We assign these speakers the topic "*unclassified*". All in all, we assign 12 different topics to the speakers. The speaker transcriptions with the same topics are concatenated together.

Tables 4.7, 4.8 and 4.9 show the assigned topics, number of speakers of these topics, and the ∅CS rates in the training, development and test texts. We observe devia-

Topic	#speakers	∅CS rate
unclassified	26	2.3
university	16	3.0
friends	12	2.0
sports	9	1.9
travel	4	2.6
family	4	2.2
internet	3	3.8
sleeping	2	3.0
Singapore	2	2.4
politics	1	1.1
movies	1	2.2
health	1	3.1
food	1	2.8

Table 4.7: Topics with number of speakers and ∅CS rates from the training text.

tions from the SEAME ∅CS rate of 2.4. Some topics have much higher ∅CS rates, e.g. *internet* or *university*, while the ∅CS rates for others are lower, e.g. *friends*. A higher ∅CS rate in the topic *internet* may be explained with the use of technical terms and the habit of talking more English at university. The lower ∅CS rates of the topics *family* and *friends* may be explained by the fact that many Singaporeans and Malaysians speak Mandarin at home.

Topic	#speakers	∅CS rate
unclassified	4	2.4
university	2	2.8
travel	1	2.3
friends	1	1.6

Table 4.8: Topics with number of speakers and ∅CS rates from the development text.

Topic	#speakers	∅CS rate
unclassified	2	3.5
food	2	1.8
university	2	2.5
travel	1	1.6
sports	1	1.6
sleeping	1	2.4
business	1	2.4

Table 4.9: Topics with number of speakers and ∅CS rates from the test text.

We build *topic* language models from the concatenated *topic* texts, which are extracted from the *train* text. The language models are interpolated with the *train* language model. Table 4.10 presents the evaluation of these *topic* language models, the interpolated *topic* with *train* language models and the *train* language model on the speaker transcriptions of the development set. The table shows the speaker ID, assigned topic and number of sentences of the speaker transcriptions. Additionally, the perplexities (PPL) of the *topic* language models, interpolated *topic*+*train* language models and the *train* language model on the development set speakers are illustrated. Further, the table presents the number of sentences used for the *topic* language models. The interpolated language models show slightly lower or equal perplexities than the *train* language model. The vocabularies of the interpolated and *train* language models are the same due to the interpolation.

Speaker	Topic	#sent. per spk.	Topic PPL	Int. Topic +Train PPL	#sent. per topic	Train PPL
0	unclass.	70	473	411	4,903	427
1	university	292	579	434	4,396	437
2	unclass.	131	538	516	4,903	537
3	unclass.	74	577	506	4,903	528
4	friends	132	900	698	3,580	704
5	university	193	595	395	4,396	395
6	unclass.	587	724	619	4,903	621
7	travel	475	731	609	964	612
All	all	-	-	-	-	552

Table 4.10: The evaluation shows that the interpolated language models are better than the *train* language model.

Our results reveals that different topics show different code-switching behavior in terms of ∅CS rates. Knowledge about the topics can be used to build topic-specific texts and language models. These language models do not not contain new sentences since all texts are taken from the training transcriptions. However, they contain more data related to the topics. The result indicate a small improvement by the interpolated language models compared to the *train* language model. It would be

interesting to see an analysis of language models which include more topic-related texts from other corpora.

5. Experiments and Results

In this chapter, we propose different methods for building code-switching language models. We present our evaluation methodology, our baseline system, and our translation system. First, we build automatically-derived class-based and part-of-speech class-based language models. Then, we introduce our *Search and Replace* approach, which combines existing code-switching text with large monolingual text corpora and generate code-switching texts and language models. Therefore, we extract monolingual segments from our SEAME code-switching corpus, translate them, search the translations in the monolingual corpora and replace the found segments with the ones from SEAME.

After the basic *Search and Replace* approach we propose different replacement strategies to improve the insertion of code-switching by incorporating different kinds of information. Initially, we follow previous approaches to improve the probability distribution of our language models. Thereafter, we restrict the replacement of segments to frequent segments, replace segments only after certain trigger words or trigger part-of-speech, or limit the replacements per segment to segment frequencies from the SEAME training text. Further, we combine promising approaches and use multiple translations to find the segments which fit best into the context. In the last experiment we analyze the effect of new texts with similar style to SEAME on our experiments.

5.1 Evaluation Methodology

Language models are usually evaluated with perplexity and ASR system are evaluated with word-error-rate. Additional to these standard measures, we introduce and use more precise measures for the recognition performance of code-switching speech.

We use the perplexity (PPL), mix error rate (MER), language model weight (LW) and word penalty (LP) as introduced in Section 2.2 and include the ∅CS rate in our analysis. Moreover, we introduce a set of measures for the evaluation of code-switching in our experiments:

Number-of-replaced-segments (NRS): The SMT-based texts are built by extracting monolingual segments from *train* and combining the segments with

monolingual corpora. The *train* sentences are split into monolingual sequences of words, which are called segments. SEAME and monolingual texts have sometimes differences in styles or topics. A measure for the style and topic similarity of the SEAME segments and text corpora is the total *number-of-replaced segments*, which describes how many segments where replaced during the SMT-based text generation.

Used-segments-ratio (USR): Another related measure is the *used-segments-ratio* which is defined as:

$$\text{used} - \text{segments} - \text{ratio(USR)} = \frac{\#\text{unique found segments}}{\#\text{unique searched segments}} \tag{5.1}$$

The number of found segments is the number of extracted segments from *train*, which have found in the monolingual text. The number of searched segments, for example for English texts is the number of extracted Chinese segments from *train* reduced by the number of wrong translations. We count Chinese segments, since they are translated to English and then used for the search. Translations are considered wrong, when English translations of Chinese contain Chinese words or the other way round. These cases can occur when our SMT system cannot translate the input segments.

If NRS or USR are low, only a few segments were replaced or few were used, which can originate for example from differences in style, topic or from wrong translations. Such wrong translations can have a different word order or unlikely word sequences, which cannot be found in proper monolingual corpora. We analyze those aspects in the Sections 5.5.5 and 5.5.6.

Code-switch n-gram coverage: This measure is derived from the n-gram coverage. It describes how many code-switch n-grams in the development text are covered by code-switch n-grams of a language model.

Recognized code-switch n-grams (ReCS[n]): The mix error rate as measure is too broad to see improvements in the code-switch n-gram recognition, since it measures the recognition of all words and not only words around code-switches. Therefore, we evaluate the percentage of *recognized code-switch 2-grams*. This is the number of code-switch 2-grams which are correctly recognized by the ASR divided by the total number of code-switch 2-grams in the development text. Section 2.2 showed that an ASR depends on different components. Therefore, it is not possible to directly attribute high or low *recognized* code-switch n-grams to the language model. The performance is also highly dependent on the acoustic model and the dictionary. Nevertheless, it is an important measure, which we plan to improve.

5.2 Baseline Language Model

The baseline ASR acoustic model is trained on the SEAME corpus [30] and the language model is estimated from the SEAME training transcriptions (*train*). For the baseline 3-gram language model, we interpolate [46] the *train* and two monolingual

language models. The monolingual language models are estimated from subsets of the Chinese GALE text and the English NIST text.

To find the best smoothing technique, we compare methods from Witten-Bell, Kneser-Ney, and the method from Kneser-Ney modified by Chen and Goodman. Table 5.1 shows that the method from Kneser-Ney results in the lowest perplexity and is therefore used in all experiments.

	Modified Kneser Ney	Kneser Ney	Witten Bell
Train (3-gram)	575	552*	659

Table 5.1: Comparison of smoothing techniques for the *train* language model. The best method is marked with a "*".

We limit the vocabulary of all language models to the *train* vocabulary. This allows us to correctly compare the perplexities and mix error rates of different language models, since we can attribute changes to differences in n-gram estimates and eliminate the impact of additional vocabulary. The size of the *train* vocabulary is 11.4K words, while there are 6.1K English words, 4.8K Chinese words, 278 words from foreign languages, 87 proper nouns, and 99 particles. The particles are mostly discourse particles with no semantic meaning, e.g. *ha* which is an expression of laughter. The restriction of the vocabulary results in an out-of-vocabulary (OOV) rate of 3.83% for all language models.

	PPL	Rel. Δ	MER(%)	Rel. Δ	OOV
Train	552	-	50.50	-	3.83%
Train+P2	511	8.0%	48.44	4.08%	1.99%
Baseline	516	6.5%	50.11	0.78%	3.83%

Table 5.2: Comparison of *train*, *train*+Part2 and the baseline language models in terms of PPL, MER and OOV rate. The changes are relative to *train*.

We obtained the weights for language model interpolation by perplexity optimization on the development text with the SRI LM Toolkit. The *train* language model has an interpolation weight of 89% in comparison to 8% for the Chinese and 3% for the English language model. We believe the high weight of *train* originates from the high ∅CS rate in the development text and the different styles between SEAME, and the monolingual texts as shown in Section 4.1. Table 5.2 shows, that interpolating the monolingual texts with *train* improves the language model perplexity by 6.5% and mix error rate by 0.78%. The *train* language model weight (LW), which leads to the best performance on the development set, is 36 with word penalty (LP) 10. For the baseline language model LW is 38 and LP is 10.

After conducting most experiments, we obtained additional SEAME training transcriptions (Part2) 4.2. Thus they are not included in the SMT-based experiments. The Part2 text corpus has 28K sentences in comparison to the SEAME text corpus in our text with 26K sentences. However, we analyze the differences between the *train* and an interpolated language model with *train* and Part2 language models.

An analysis of the Part2 vocabulary reveals the size of 14K words. The interpolation weights of these language models are 51% for *train* and 49% for Part2. Table 5.2 shows that the interpolation of the Part 2 language model improves the perplexity by 7.4% and the mix error rate by 4.1%. The vocabulary is not restricted in this setup, such that the interpolated language model has a vocabulary of 18K words. The OOV rate decreased from 3.83% to 1.99% on the development text.

We also conduct an *oracle* experiment, where the 3-gram language model is built from the development text and then evaluated on the same set. This setup shows the upper boundary or best possible performance with the current acoustic model since it contains the correct n-grams. We evaluate this *oracle* language model with the full development text vocabulary and with a restricted vocabulary (R). We restrict the vocabulary to the *train* vocabulary for comparability to the baseline and the other experiments. The language model with the full vocabulary of the development text shows a better perplexity and mix error rate. In both ASR experiments, the best language model weight (LW) is 50. This is the highest possible weight in our experiments and reflects the importance of the development text language model. The word penalties are -8 for the unrestricted and -6 for the restricted language model. Table 5.3 shows the perplexities and mix error rates of the *oracle* language models in comparison to the *train* and baseline language models. The *oracle* (restricted) shows a perplexity improvement of 94% (93%) to 31 (37) from 552. The mix error rate improves from 50.50% by 55% (43%) to 22.29% (28.35%). Table 5.3 illustrates

	PPL	MER	Rel. Δ	OOV	Vocab.
Train	552	50.50	-	3.83%	11.4K
Baseline	516	50.11	0.78%	3.83%	11.4K
Dev (R)	37	28.35	43.43%	3.83%	2.7K
Dev	31	22.29	55.51%	0%	3.3K

Table 5.3: Comparison of the development text (oracle) language models to the *train* and baseline language models. The changes are relative to the *train* language model. The vocabulary of the (R) language model is restricted to the *train* vocabulary.

that the language model is an important factor in our speech recognizer. Our *oracle* language model improves the performance by more than 55%. The difference between the *oracle* experiments indicates that the out-of-vocabulary rate has also strong influence on the recognition.

5.3 Statistical Machine Translation

We use the Moses Statistical Machine Translation Toolkit for our experiments [27]. Moses' main components are a phrase table and a language model. The Chinese-English and English-Chinese phrase tables are estimated from the parallel NIST Chinese-English corpus. For differences in word order between the translation languages, a word-reordering (distortion) model is implemented. We divide the parallel corpus in a training, tuning and evaluation set. Table 5.4 shows the breakdown of the different sets. The English and Chinese language models for our SMT system are built from the training texts.

	Chinese:			English:		
	Train	Tune	Eval	Train	Tune	Eval
#words	228M	28K	28K	255M	32K	32K
#sentences	8.4M	1K	1K	8.4M	1K	1K

Table 5.4: Chinese and English training (Train), tuning (Tune) and evaluation (Eval) texts of the SMT system.

As introduced in Section 2.3, we used *minimum error rate training* (MERT) [2] to find and use the optimal weights of the phrase table, the language model and of distortion (reordering) model. MERT minimizes the BLEU score of the SMT system on the *tuning* text. The Table 5.5 shows that MERT improved the English-to-Chinese translation quality on the *evaluation* text by 19% and the Chinese-to-English translations by 41%.

	Engl.-to-Chin.	Rel. Δ	Chin.-to-Engl.	Rel. Δ
Moses w/o. MERT	7.78	-	11.79	-
Moses w. MERT	9.27	19%	16.58	41%
Resegmented	16.16	108%	-	-
Google translate (resegm.)	32.77	321%	18.62	58%

Table 5.5: Improvement of the BLEU score of Moses due to MERT and resegmentation, and a comparison to Google translate. Changes are relative to the system without MERT.

Our output translations are in lower case and contain punctuation marks. They are removed since we use the translations as search patterns for SEAME which does not contain any punctuation. Our phrase tables were large. Thus, they have to be converted to a binary format to fit into memory.

Since this initial system is slow on our machines, we tune it for higher speed, while keeping a similar BLEU score. One factor for speed improvement is a reduction of the number of lookups (L) in the parallel phrase table during a translation. Moses offers a parameter to limit these lookups. We conduct several experiments to find a balance between speed and BLEU score. The result is shown in Table 5.6. We achieve a 10% speed improvement for English-to-Chinese translations and a 23% improvement for Chinese-to-English translations, while having the same BLEU score.

Since the quality of our SMT system determines the quality of the new language models, we compare our system to another translation program. For this system we use Google translate [1]. We evaluate the performance of both systems on the SMT evaluation set. The Chinese output from Google translate is unsegmented. Therefore, we use the Stanford Chinese Segmenter [49] to normalize the Google translate output, our Moses output and the SMT evaluation text to make the results comparable.

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

	L=∞	L=1	Rel. Δ	L=5	Rel. Δ	L=10*	Rel. Δ
English-to-Chinese:							
BLEU	9.27	7.72	-17%	9.19	-1%	9.27	0%
Duration (min.)	70	11	84%	36	49%	63	10%
Chinese-to-English:							
BLEU	16.58	12.02	-28%	15.75	-5%	16.58	0%
Duration (min.)	78	11	86%	33	53%	60	23%

Table 5.6: Comparison of BLEU scores and speed by the number of phrase table lookups (L). The changes are relative to unlimited (∞) lookups. The best trade-off is achieved with a maximum of 10 lookups.

Table 5.5 shows that our SMT system performs worse than Google translate for both Chinese-to-English and English-to-Chinese translations. Our best systems reach BLEU scores of 16 for English-to-Chinese and Chinese-to-English translations. For Chinese-to-English translations is Googles BLEU score 12% higher compared to our SMT sytem. For English-to-Chinese translations, Google shows an outperformance of more than 100%. However, we still decide to use the Moses system, as we have the full control of all parameters. For example, Moses outputs an n-best list of translations for each input sentence. In Section 5.5.5 we illustrate an approach which finds the best fitting translation for a given sentence from the 5-best translations. Google translate, however, does not offer information about n-best translations. Furthermore, Google translate is slow as there is no official API: This requires to write a grabber which extracts translations from the web interface. The grabber cannot translate all input sentences at once and is therefore relatively slow. Google offers two web interfaces: One interface where input text is translates immediately and another interface where text files can be uploaded. Google limits the upload file size to 1 MB per upload.

5.4 Class-based Language Models

Class-based language models are often used to build language models from little text data. These language models contain class n-grams instead of word n-grams. Some class-based language models use linguistically motivated classes, for example verbs, nouns, etc. The advantage is that there are usually less classes than words, and there are multiple words in one class. Therefore, the class n-grams are estimated from multiple words, which leads to more robust estimates. During the perplexity computation and the ASR decoding, the classes are expanded to words with class-to-word mappings.

5.4.1 Automatically-induced Class Language Model

One class-based approach to improve the SEAME code-switching language models is to use automatically-induced classes [46]. This class-based approach uses distributional statistics to induce word classes. The classes are automatically-induced with the Brown-algorithm [4]. This algorithm initially assigns each word to a distinct class and then merges the classes based on the average mutual information between

two classes. This procedure terminates, when a predetermined number of classes is reached. We used the SRI LM Toolkit [46] for the generation of the class-based language models and the interpolation of class-based and word-based language models. The automatically-induced class-based language models generated with the SRI LM Toolkit are 2-gram language models.

For the clustering algorithm, the number of classes is required as input. We do not know a priori the optimal number of classes. Thus, we create class-based language models with different numbers of classes and compute the perplexities on the development text. Table 5.7 shows that 500 is the optimal number of classes from the set 50, 100, 200, 300, 400, 500, 600, whereas the optimum for the interpolated class-and *train* language model is 300. The interpolated language model contains word n-grams, for $n \in 1, 2, 3$ and class n-grams, for $n \in 1, 2$.

#classes	50	100	200	300	400	500	600
Class	708	657	597	572	567	565.284*	565.346
Class+*train*	486	481	476.6	476.5*	482	486	489

Table 5.7: Perplexity of class-based language models with different number of classes. The optima are marked with a "*".

The main advantage of class-based language models is that they give probability estimates for word n-grams not seen in the *train*. Table 5.8 shows the code-switch n-gram coverages. The code-switch 2-gram and 3-gram coverages of the class+*train* language model are higher than those of the baseline.

There are cases when word n-grams give better estimates than class n-grams. For class 1-grams, the probability mass is equally distributed over all words in the class. Further, each class n-gram (n > 1) has a probability estimate assigned, which is derived from the occurence count of the class 1-grams. Some of the word sequences (with n words) in the training set have a higher probability than the corresponding class n-gram. On the other hand, word sequences exist with lower probabilities than the corresponding class n-gram. Hence, word n-grams can give preciser estimates.

	CS 2-gram coverage	CS 3-gram coverage
Baseline	26%	5%
Class+*train*	77%	8%

Table 5.8: The interpolated class-based and *train* language model improve the coverages of code-switch n-grams relative to the baseline.

This problem can be seen in Table 5.9. The class-based language model shows a worse perplexity than the class+*train* language model, which also contains word n-grams. The usage of the pure class-based language model with our recognizer reveals a worse mix error rate than the version interpolated with *train*. But the interpolated language model shows an improvement of 7.8% in perplexity and of 1.8% in ReCS[2] compared to the baseline. The language model weight LW is 40 with an LP of 2. The code-switch n-gram coverages and the perplexities improve, but this cannot be translated into a better mix error rate.

	PPL	Rel. Δ	MER (%)	Rel. Δ	ReCS[2]	Rel. Δ
Train	552	-	50.50	-	24.51%	-
Baseline	516	-	50.11	-	23.77%	-
Class	565	-9.5%	61.96	-23.65%	15.55%	-34.6%
Class+*train*	477	7.8%	50.32	-0.42%	24.20%	1.8%

Table 5.9: The class-based language model is much worse than the interpolated class+*train* language model. Changes are relative to baseline.

5.4.2 Part-of-Speech Language Model

Another technique for improving language models are part-of-speech (POS) language models. POS language models include grammatical information. These language models are class-based language models, where each POS tag is a class. A POS tag is a part-of-speech identifier, for example "VV" for verb. These classes work similar to automatically-induced class language models. They are able to cope with little data and can estimate the probabilities of unseen n-grams. The advantage of POS language models over the automatically-induced class language models which we described in Section 5.4.1 is that they do not rely on automatically derivation of classes. Additionally, the number of classes does not have to be determined by experiments. The optimal number of POS tags is not known as well, but Marcus et al. [31] showed in an overview of POS tagging research, that the number of tags for the English language was in a relatively small range between 87 and 197 tags. In our setup is the number of classes equivalent to the number of supported part-of-speech tags by the POS tagger.

Train is tagged with Burgmer's POS tagger [6]. The POS tags are extracted and used to train a POS 3-gram language model. We interpolate this language model with the *train* language model to include the word n-grams.

Burgmer proposed to merge both Chinese and English part-of-speech tag sets together. For example, he merged all 6 English verb form tags and 4 Chinese verb form tags to 1 general "verb" form tag. We do not follow this as the number of POS is already very low: The unified tag set has 28 POS tags, while the Chinese tag set has 33 tags [51] and the English tag set has 36 POS tags and 12 other tags (e.g. for punctuation) [31]. The most important reason, for not merging the POS tags, is that the merge would lead to information loss. For example the gerund of an English verb (POS tag: VBG) gives more grammatical information than a universal "verb" tag. Additionally, our tags carry language information, since the English tags start with a "_" and the Chinese tags started with a "#".

	CS 2-gram coverage	CS 3-gram coverage
Baseline	26%	5%
Class+*train*	77%	8%
POS+*train*	98%	92%

Table 5.10: The code-switch n-gram coverages of POS+*train* are higher than for the baseline or class+*train* language models.

Table 5.10 presents the code-switch n-gram coverages. Using the merged POS language model increases the coverages over the baseline and the automatically-induced class+*train* language models. The strong increase results from the lower number (81) of classes in comparison to the automatically-induced class+*train* language model (300), and a vocabulary of 11.4K. While the low POS number gave less n-grams, the number of words per class increases and leads to a combinatory increase of possible word 2-and 3-grams. This finally resulted in an increase of the coverages.

	PPL	Rel. Δ	MER (%)	Rel. Δ	ReCS[2]	Rel. Δ
Train	552	-	50.50	-	24.51%	-
Baseline	516	-	50.11	-	23.77%	-
Class+*train*	477	7.8%	50.32	-0.42%	24.20%	1.8%
POS+*train*	[1]720	-39.5%	50.64	-1.05%	23.85%	0.4%

Table 5.11: Comparison of the POS+*train* language model to the automatically-induced class language model, baseline and *train* language models. The class- and POS-based language models use differently generated class-to-word mappings[1]. Changes are relative to the baseline.

The perplexity computation of class-based language models[1] requires a mapping between the classes and the words. This mapping containes estimates how likely classes are expanded to their member words. The SRI LM Toolkit computes these estimates for automatically-induced class language models, but not for manually created class language models, in our case POS language models. Our class j-to-word i expansion is estimated with [2]:

$$\hat{P}_{i,j} = \frac{\#\text{words i in class j} + 1}{\#\text{occurrences of class j} \ + \ \#\text{classes}} \tag{5.2}$$

Table 5.11 shows the performance of POS+*train* language model in comparison to the baseline and automatically-induced class+*train* language models. Althought the POS+*train* language model has a high CS n-gram coverage, it performs worse than the baseline in perplexity (-39.5%) and mix error rate (-1.05%). The automatically-induced class+*train* language model is more suitable than the POS+*train* language model for our data.

5.5 SMT-based Text Generation

5.5.1 Search and Replace

We concentrate on the improvement of code-switching support for ASR. This is achieved by generating new code-switch texts, from which we built code-switching language models. The aim of the new code-switching texts is to better estimate code-switching n-grams. Robust, monolingual n-grams can already be estimated from a vast amount of available, monolingual texts.

For our experiments, we leverage the existing code-switching *train* transcriptions with monolingual text corpora. An overview of this process is shown in Figure 5.1.

[2]Proposed by Andreas Stolcke during an eMail exchange

The first step (1) is to extract all monolingual segments from *train*. This meant splitting each sentence in monolingual parts, which we call segments, and save them to English or Chinese segment lists. After the extraction, we translate the English segments into Chinese and Chinese into English (2), search for the translations in monolingual text corpora (3), and replace all found segments in the monolingual text corpora with the original segment from the *train* (4). The monolingual texts are also called base texts in this work. Our approach restricts new code-switching inserted into English (Chinese) texts to Chinese (English) segments from *train*, but leads to new code-switch n-grams reaching over the boundaries of the replaced segments. We call this experiment the *Search and Replace (S&R)* approach.

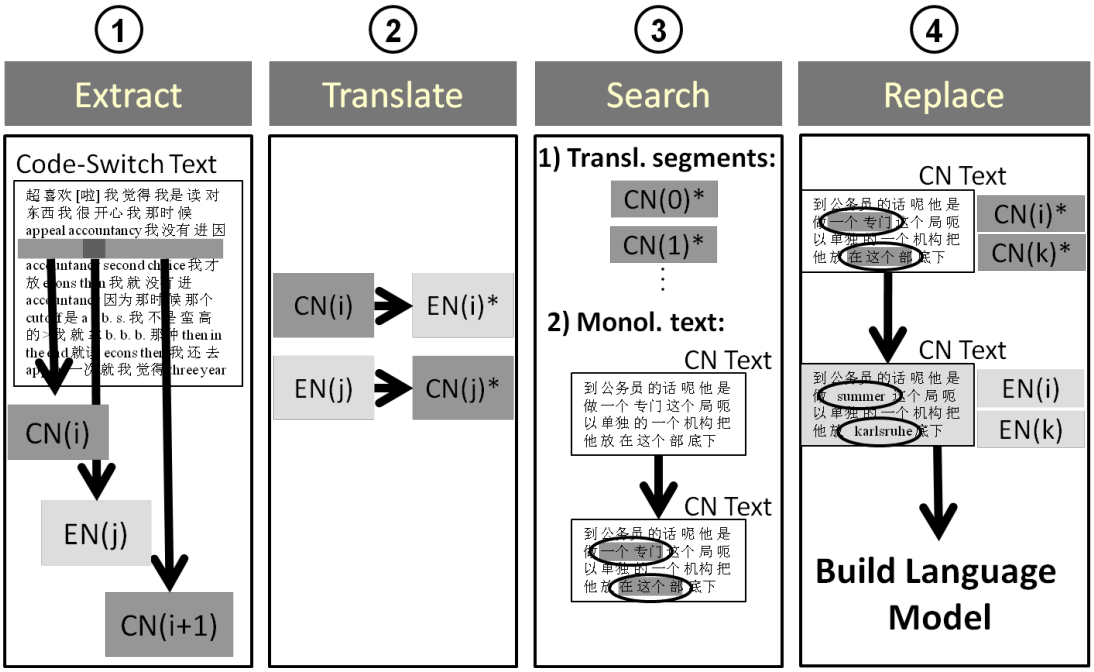

Figure 5.1: The *Search and Replace (S&R)* approach.

In the first step, we extract all unique monolingual Chinese (26K) and English (17K) and segments. The use of the original segments guarantees that the generated code-switching is natural. Some parts of sentences cannot be translated by our Moses system. In this case, the Moses output contains words in the source language: An analysis shows that 777 (4%) of the English-to-Chinese translations are not translated correctly and contain English words. This is also the case for 402 (2%) of the Chinese-to-English translations. These sentences are removed and not used as patterns for the search.

Table 5.12 presents the *number-of-replaced-segments* and the *used-segments-ratio* (USR) for both monolingual texts. All found segments are replaced with the corresponding translations. However, only a small fraction of the extracted *train* segments, 8% of monolingual English and 22% of monolingual Chinese, are found in the monolingual texts. Especially longer segments are not found. This originates from differences in style or topics between SEAME and the base texts. Erroneous translations are also a possible reason.
The *number-of-replaced-segments* is high compared to the number of sentences: 3.4M

in the Chinese text which contains 395K sentences and 2.8M in the English text which contains 345K sentences. The ∅CS rates are 10.43 for the text based on Chinese and 11.25 for the English-based text. This leads to an overestimation of code-switch n-grams, due to the high number of code-switches in contrast to the lower ∅CS rate in *train* of 2.4. The origin of the high ∅CS is the replacement strategy which translates all found segments. These segments are replaced, without any restrictions. It is possible, that multiple successive segments in one sentences are replaced with their respective translations.

	English-based	Chinese-based
Number-of-replaced-segments	2.7M	3.4M
Used-segments-ratio	8%	22%
∅CS rate	11.3	10.4

Table 5.12: The used-segments-ratio (USR) of the English-and Chinese-based S&R texts was low, but the total *number-of-replaced-segments* was high and lead to higher ∅CS rates than in *train* (∅CS rate=2.4).

We build language models from Chinese and English SMT-based code-switching texts. From these texts, we use both code-switching and monolingual sentences to build language models. The interpolation of both with the *train* language model (*S&R+T*) results in higher code-switch n-gram coverages than the baseline language model on the development text. The total number of code-switch 2-grams (3-grams) increases heavily with 1926% relative (4043%). The coverage of code-switch 2-gram increases by 58% relative to 57% and the coverage of code-switch 3-grams increases by 38% relative to 11%. The numbers of generated code-switch 2-and 3-grams are higher than the 4K code-switch 2-grams and 7K 3-grams in the development text. Table 5.13 presents an overview of the generated code-switches and the code-switch n-gram coverages.

	#CS 2-grams	CS 2-gram cov.	#CS 3-gram	CS 3-gram cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
Rel. Δ	1926%	58%	4043%	38%

Table 5.13: The *S&R+T* approach shows an increase in the number and coverages of code-switch n-grams.

Table 5.14 illustrates the perplexities and the mix error rates on the development set. Both drop from the baseline to the *S&R*. The *recognized code-switch 2-grams (ReCS[2])* increased by 2.35% to 1,178 (24.32%). For *S&R+T*, the language model weight is 40 and LP is 8.
The drop in perplexity and mix error rate can be explained with too frequent generation of code-switches per sentence. This leads to an overestimation of code-switching. Therefore, the following experiments cope with improving the probabilities of n-grams, in particular code-switch n-grams.

	PPL	Rel. Δ	MER (%)	Rel. Δ	ReCS[2]	Rel. Δ
Baseline	516	-	50.11	-	23.77%	-
S&R+T	533	-3.17%	50.29	-0.35%	24.32%	2.35%

Table 5.14: Comparion of the *S&R+T* to the baseline language model.

5.5.2 N-gram Frequency Distribution Sharing

The previous experiment increases the code-switch coverages, but shows little improvement in perplexity and mix error rate. Therefore, the next step is to improve the probabilities of the code-switch n-grams. We use the text generated from the *S&R* approach as base, since it has a high code-switch n-gram coverages and adjust the n-gram probabilities with different strategies. This is achieved by sharing the n-gram frequencies between *train* and *S&R*.

5.5.2.1 Full N-gram Frequency Distribution Sharing

The generated *S&R* language model showed a good improvement in code-switch coverages. But the high code-switch rates indicated erroneous probability estimates. Therefore, we want to modify the n-gram probabilities to improve *S&R*.
Topkara and Schultz [50] built a bilingual German and English language model for a multilingual speech application. Their training data was unbalanced with regard to text corpora size: The English text was 218 times larger than the German text. To built one multilingual language model they balanced the probability distribution functions. The key for the improvement was to assign frequencies from English n-grams to German n-grams. This assignment was done based on the frequency rank of n-grams. Their approach improved German recognition by 17% absolute to 31% WER in comparison to their baseline experiment, but decreased the English WER from 15.5% to 17%. Their baseline language model was built from concatenated English and German text corpora.

We follow this approach (*TK-0*) to improve the *S&R* probabilities and map the *train* n-gram frequencies to the *S&R* n-grams. Figure 5.2 shows an illustration of the frequency adjustements. The *train* frequencies are assigned to the *S&R* frequencies with the same frequency rank. A comparison of the *train* and *S&R* n-gram frequencies reveals that there are 120K 2-grams and 220K 3-grams in *train* compared to 1.5M 2-grams and 4.4M 3-grams in *S&R*. As the *S&R* language model has more n-grams than the *train* language model, we assign the lowest *train* 1-gram frequency to the remaining *S&R* 1-grams. This is done analogous with the 2-and 3-grams. The English- and Chinese-based TK-0 language models and *train* are interpolated (*S&R(TK-0)+T*).

	PPL	Rel. Δ	MER (%)	Rel. Δ	ReCS[2]	Rel. Δ
Baseline	516	-	50.11	-	23.77%	-
S&R(TK-0)+T	537	-4.1%	50.24	-0.26%	24.18%	1.7%

Table 5.15: PPL and MER of the baseline and *S&R(TK-0)+T* language models.

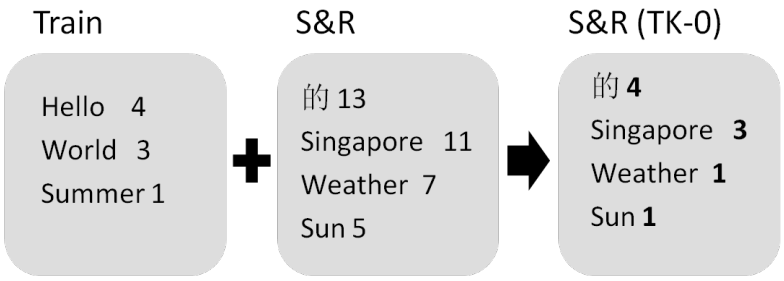

Figure 5.2: Illustration of the full n-gram frequency distribution sharing

The n-gram coverages and the ∅CS rates are the same as in the *S&R* experiment. Table 5.15 shows that the TK-0 experiment has a worse perplexity and mix error rate, although ReCS[2] increased (LW=42, LP=10).
There are two differences to Topkaras setup: First the frequencies of the English n-grams are from a large corpus, whereas in our case the *train* n-gram frequencies are estimated from little text data. The second aspect regards the importance of the frequency rank in Topkara's approach. In our case, we do not know if the *S&R* n-gram ranking is correct before the modification. So the assignment of frequencies is based on a biased frequency ranks.

5.5.2.2 N-gram Frequency Distribution Sharing with Adjustement for Code-Switching

Topkara and Schultz [50] proposed another method (TK-1) in the setup introduced in the last experiment for improving their multilingual language model. Their English n-gram frequencies were again used to improve the German n-gram frequencies. In that case, the English 1-gram frequencies were assigned to German 1-gram frequencies based on their frequency rank. The German higher order n-gram frequencies, n>1, were adjusted based on lower order n-gram frequencies. The frequencies of 2-grams, were adjusted as followed:

$$f(A,B)_{new} = f(A)_{old} \cdot ratio(A) \tag{5.3}$$

where $f(A,B)$ is the frequency of the 2-gram AB, and $f(A)$ is the frequency of the 1-gram A. The index *new* describes the adjusted n-gram frequencies, whereas the index *old* describes the original frequencies. $ratio(A)$ is defined as the 1-gram X frequency from English, with the same frequency rank as the 1-gram A from the German text, divided by the frequency of A.

We propose a similar approach, taking the nature of code-switching n-grams into consideration. We adjust higher order n-grams by their weighted ratios of lower order n-grams. The new frequencies for 2-grams are computed as follows:

$$f(A,B)_{new} = \frac{1}{2} \cdot f(A,B) \cdot ratio(A) + \frac{1}{2} \cdot f(A,B) \cdot ratio(B) \tag{5.4}$$

$ratio(A)$ is here defined as the 1-gram X frequency from *train*, with the same frequency rank as the 1-gram A from the *S&R* text, divided by the frequency of A. This approach has an advantage for code-switching n-grams since the new higher order code-switch n-gram frequencies depend on the frequencies of their lower order

	PPL	Rel. Δ	MER (%)	Rel. Δ	ReCS[2]	Rel. Δ
Baseline	516	-	50.11	-	23.77%	-
S&R(TK-1)+T	536	-3.9%	50.30	-0.37%	24.30%	2.2%

Table 5.16: Comparison of the *TK-1* approach to baseline.

n-grams. The lower order n-grams (e.g. 1-grams) contained in higher order n-grams may have different languages. 3-grams are computed analogously.

The result of this approach is shown in Table 5.16: The perplexity increased by 3.9% relative and the mix error rate by 0.37% relative (LW=38, LP=10). ReCS[2] increased by 2.2% to 1,177 (24.30%). This worse performance originated from the same two issues mentioned in the last experiment: The amount of *train* data was little and we relied on the ranking of the *S&R* n-grams. These aspects guided us to try different approaches.

5.5.3 Advanced Segment Replacement Strategies

The *Search and Replace* experiment showed higher code-switch 2- and 3-gram coverages of 58% and 38%. Nevertheless, the perplexity and mix error rate did not improve as much. The reason was the high number of generated code-switches which resulted in overestimation of code-switching. Additionally, the high usage of code-switching left less data for monolingual n-gram estimation. The next experiments improves code-switch n-gram probabilities, while keeping high code-switch n-gram coverages. For this purpose, we propose multiple advanced segment-replacement strategies.

The general strategy lies in a smart restriction of the number of generated code-switches with the goal to generate better code-switch n-grams. To achieve this, we first generate code-switches only from frequent segments. In the following experiments, we use information of the code-switching context to see if code-switching can be inserted. The third strategy sets target frequencies for segment replacements in the SMT-based text. These frequencies are derived from the segment frequencies in *train*.

5.5.3.1 Minimum Segment Occurrence Threshold

With the *Minimum Segment Occurrence Threshold* (Threshold) approach, we want to improve code-switching probabilities and lower the high number of code-switch n-grams of *S&R*, while keeping high code-switch coverages. This is achieved by only searching and replacing exclusively frequent segments, which have a minimum occurrence count in *train*. The used minimum occurence count will be discussed later. The number of search segments should decrease and result in a lower *number-of-replaced-segments*. The aim is to keep only frequent and common segments, which in turn decrease the number of code-switch n-grams and improve the n-gram probabilities.

Table 5.17 shows the number of segments dependent on the threshold. For our experiments, we use a threshold of 2. The number of Chinese segments for the monolingual English text drops from 17.4K to 3.1K. The number of English segments

	#Chinese segments	#English segments
Threshold = 1	26.6K	17.4K
Threshold = 2	1.9K	3.1K
Threshold = 3	0.9K	1.8K
Threshold = 4	0.6K	1.3K

Table 5.17: The number of segments varies with the minimum occurrence count (threshold) for Chinese and English segments.

for the monolingual Chinese text decreases from 26.6K to 1.9K. However, we do not generate a *Threshold* with threshold 3, since we believe that the number of segments with threshold > 2 was too low. Many segments occur only once in *train*.

	#segments	Rel. Δ	USR	Rel. Δ	NRS	Rel. Δ	∅CS
English-based:							
S&R	26.6K	-	8%	-	2.8M	-	11.3
Threshold 2	1.9K	-93%	41%	413%	2M	-28%	9.2
Chinese-based:							
S&R	17.4K	-	22%	-	3.4M	-	10.4
Threshold 2	3.1K	-82%	60%	172%	2.3M	-32%	8.7

Table 5.18: Comparison of *Threshold* to *S&R*, with regard to the number of segments, *used-segments-ratio* (USR), *number-of-replaced-segments* (NRS) and average language turns (∅CS).

Table 5.18 shows that the *number-of-replaced-segments* (NRS) falls. *Threshold* shows a lower number of code-switches and a higher USR of the segments than *S&R*. The *used-segments-ratio* (USR) describes how many of the searched segments are found. Therefore, it is a measure of the segment fitting to the monolingual base text and the translation quality. The higher USRs, of the English-based text (+413%) and the Chinese-based text (+172%), are a result of the nature of the segments: In general the frequent segments are shorter than the infrequent ones, and appear more frequently in the monolingual texts. The ∅CS rates drop to 8.7 for the Chinese-based text and to 9.2 for the English-based text. This ∅CS rate is still much higher than the SEAME ∅CS rate of 2.4.

The English- and Chinese-based language models of the SMT-based *Threshold*, are interpolated (*Threshold+T*) with the *train* language model. Table 5.19 highlights that the number of generated code-switch n-grams of *Threshold* compared to *S&R* drops by 23% for 2-grams and by 14% for 3-grams, whereas the code-switch coverages increases. The coverages are much higher than the ones the baseline.

Table 5.20 shows that both perplexity and mix error rate are worse than for the baseline, but better than for the *Search and Replace* approach. ReCS[2] of 1,170 decreased (1.7%) from *S&R* with 1,178. This is unfortunate, because ReCS[2] measures the number of correctly recognized code-switch 2-grams. The ∅CS rates of the English and Chinese texts are higher than for SEAME, but lower than for *S&R*. The language model *Threshold+T* weighted higher with 40 (LP=10) than *S&R*

	#CS 2-grams	CS 2-gram cov.	#CS 3-grams	CS 3-gram
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
Threshold+T	533K	58%	2.5M	12%
Rel. Δ	1468%	61%	3471%	50%

Table 5.19: The *Threshold* approach shows high coverages of the code-switch n-grams, with fewer code-switch n-grams than *S&R*. Changes are relative to the baseline.

(LW=38). The main progress of the *Threshold* approach are the fewer code-switch n-grams compared to *S&R* with a similar coverages on the development set. But no improvement over the baseline in perplexity and mix error rate is achieved. The conclusion is that the number of code-switch n-grams is still too high and code-switching is still overestimated.

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	Rel.Δ	∅CS(E)	∅CS(C)
Baseline	516	-	50.11	-	23.77%	-	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	2.3%	11.2	10.4
Thres.+T	524	-1.6%	50.23	-0.24%	24.16%	1.7%	9.1	8.7

Table 5.20: Comparison of the *Threshold+T*, *S&R+T*, and baseline language models. PPL and MER decrease slightly. The ∅CSs drops for English (E)- and Chinese(C)-based code-switch texts. Changes are relative to baseline.

5.5.3.2 Context-sensitive Segment Replacement Strategies

Trigger Words

In this experiment we use context information to identify, if a code-switch can be inserted into a monolingual sentence. The goal is to create less code-switch n-grams with more robust probability estimates, while keeping high code-switch coverages. The context-sensitive segment-replacement should lead to robust code-switch n-gram probabilities.

We include context information with trigger words (TrigWords). These words are directly preceding a code-switch. Later we will use part-of-speech tags instead. We follow earlier work, which predicted code-switching points with different methods: Solorio [45] predicted English-Spanish code-switching based on multiple features, such as the word or part-of-speech tag preceding the code-switch. Burgmer [6] followed Solorio and analyzed the prediction of English-Cantonese code-switching points with similar features on a small corpus. Chinese switches were better predicted by trigger words, English switches by part-of-speech triggers. Burgmer reported better results in terms of F1-score than Solorio, but neither of the approaches were implemented into ASR systems.

First, we conduct an analysis of the relationship between trigger words and code-switching of SEAME. Therefore, we extract all unique trigger words from *train*. We found 5K trigger words. They cover 91% of all language switches in the development text. Unfortunately, these words also cover 89% of all words in the development text and 91% of all words in *train*. Only 20% of the trigger word occurrences in *train* are followed by a code-switch. This indicates that trigger words have a limited prediction power.

Solorio and Burgmer saw reasonable results with part-of-speech-based code-

Word	Meaning	Frequency	CS rate
then	-	3,599	55%
那个	that	2,654	52%
的	associative particle	6,913	20%
啊	discourse particle ”ah”	4,354	19%
是	to be	3,063	25%
一个	one	1,300	48%
去	to go	2,342	26%
有	to have	2,454	23%
很	very	2,353	22%
but	-	1,416	36%

Table 5.21: List of the top 10 trigger words by code-switch rate.

switch prediction. Hence, we build a artificial text using trigger words to improve our replacements. Table 5.21 shows the list of the 10 trigger words with the highest code-switch ratio. The code-switch ratio of word w is defined as:

$$\text{CS rate(w)} = \frac{\#\text{occurrences of w followed by a CS}}{\#\text{occurrences of w}} \tag{5.5}$$

These words are similar to those found by Burgmer [6]: 4 out of his top 6 Chinese trigger words are in our top 10 list, whereas ”,” is his top switch word. He stated, that his most frequent English words triggering a CS (e.g. ”,”, baby and Goa) might not be representative.

Text generation

We search for the extracted and translated segments in the monolingual text. When a segment is found, the algorithm checks, if the word preceding the current segment is in the list of trigger words. If it is a trigger word, then the segment is replaced and the search continued. If it is not in the trigger word list, the search continues without replacing. The English and Chinese SMT-based language models are then interpolated with the *train* language model (*TrigWords+T*).

Table 5.22 illustrates that the number of code-switch 2-grams (3-grams) increases compared to the baseline, but decreases by 27% (4%) in comparison to *S&R*. The coverage for the code-switch 2-grams (3-grams) of the *TrigWords+T* language model

with 56% (25%) relative is higher than for the baseline. Compared to *S&R* the coverages decreases only slightly. This decrease can be attributed to a stricter replacement method for the segments.

	#CS 2-gr.	CS 2-gr. cov.	#CS 3-gr.	CS 3-gr. cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
Threshold+T	533K	58%	2.5M	12%
TrigWords+T	506K	56%	2.4M	10%
Rel. Δ	1388%	56%	3329%	25%

Table 5.22: *TrigWords+T* shows similar coverages of the code-switch n-grams, with less total code-switch n-grams than *S&R+T* or *Threshold+T*. Changes are relative to the baseline

Table 5.23 illustrates that although we have lower ∅CS rates and more reasonable code-switch n-gram probabilities, we keep high code-switch coverages. The perplexity and mix error rate are worse than for the baseline. ReCS[2] increases to 1,185 (3.0%) compared to 1,151 of *S&R*. Therefore, we consider this approach superior to the *S&R* or *Threshold* approaches, as it produces lower ∅CS rates and code-switch n-grams with similar coverages. The language model weight is lower with 38 (LP=10) than in the *Threshold* approach.

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	516	-	50.11	-	23.77%	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	11.2	10.4
Threshold+T	524	-1.6%	50.23	-0.24%	24.16%	9.1	8.7
TrigWords+T	531	-2.9%	50.15	-0.08%	24.47%	6.7	6.9

Table 5.23: Comparison of *TrigWords+T*, *Threshold+T*, *S&R+T*, and the baseline language models in terms of PPL, MER and ReCS[2]. ∅CS rates decreas for the English(E)- and Chinese(C)-based texts.

Trigger Part-of-Speech Tags

Instead of using context information in the form of trigger words, we also analyze the use of part-of-speech tags as triggers (TrigPOS). The goal is to generate fewer code-switch n-grams, while achieving high coverages and robust n-gram probabilities. Part-of-speech tags are used as semantic features for language analysis. Burgmer [6] and Solorio [44] showed that these tags are useful features for code-switching analysis and prediction.

The manual annotation of POS tags is cumbersome. Therefore, we use an available POS tagger. Burgmer [6] developed a Chinese-English code-switch POS tagger based on monolingual Chinese and monolingual English POS taggers [47]. His POS tagger splits code-switching text in monolingual language islands and executes the

monolingual tagger on them. A language island is a series of monolingual words with embedded words in the other language. These embedded islands are either single words or two consecutive words. The motivation is that the POS taggers are able to properly tag unknown words, or in our case words in a different language, based on their context. Assigning POS tags based on context is more appropriate than only tagging single words without context.

We tag *train* and analyze the trigger part-of-speech tags, which are located before code-switches: Table 5.24 shows the 10 POS tags, which occur the most frequent before code-switches in *train*. They cover 70% of the code-switches in *train* and 70% of the code-switches in the development text. Unfortunately, they cover only 43% of all POS tags in *train*. This is more promising than the trigger words, where 90% of all words are covered. The top 10 trigger POS tags in terms of CS rate had mostly a code-switch ratio between 20% and 30%. This is not high, but code-switching is voluntary, and the speakers can choose to code-switch.

POS	Definiton	Frequency	CS rate
ON	Onomatopoeia	2	100%
OR	Proper noun	3,831	61%
SB	Short bei4 (被)-construction	51	33%
DEG	Associative de (的)	2,219	31%
MSP	Other particle	232	29%
NN	Other noun	51,487	28%
VV	Other verbs	40,539	24%
M	Measure word	1,477	22%
VC	Copula	2,811	22%
DT	Determiner	10,231	22%

Table 5.24: List of the top 10 trigger POS tags from *train* sorted descending by code-switch rate with POS definitions.

Text generation

We use the trigger part-of-speech tags in the same way, as the trigger words in the last experiment. We search for the extracted and translated segments, and if the word, preceding the found segment, has an assigned part-of-speech tag which is a trigger tag, it is replaced with the source segment. Both English-and Chinese-based *TrigPOS* language models and the *train* language model are interpolated (*TrigPOS+T*). Table 5.25 presents the interpolated language model, which has the same code-switch 2-and 3-gram coverages as the *S&R+T* language model. In comparison with the *TrigWords+T* approach, the new language model contains more code-switch 2- and less code-switch 3-grams, and the coverages are slightly higher.

Table 5.26 shows the resulting perplexities and mix error rates. The perplexity of *TrigPOS+T* is 2.3% relative better than the *TrigWords* approach, but still worse than the baseline. The mix error rate of the ASR with the *TrigPOS+T* language

	#CS 2-grams	CS 2-gram cov.	#CS 3-grams	CS 3-gram cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
TrigWords+T	506K	56%	2.4M	10%
TrigPOS+T	569K	57%	2.1M	11%
Rel. Δ	1574%	58%	2900%	38%

Table 5.25: Comparison of the covered code-switch n-grams of *TrigPOS+T*, *TrigWords+T*, *S&R+T*, and baseline language models. Changes are relative from *TrigPOS+T* to the baseline.

model (50.07%) improve to all previous experiments. The comparison of *S&R* to *TrigWords* and *TrigPOS* shows that both trigger approaches significantly decrease the number of generated code-switch n-grams. ReCS[2] is higher than for the baseline and *S&R*, but 0.6% absolute lower than *TrigWords*. The ∅CS rates are higher in the Chinese text and lower in the English text than in the *TrigWords* experiment. The use of trigger POS was marginally better in terms of perplexity and mix error rate to the *TrigWords*. This is achieved with the same code-switch n-gram coverages as *S&R*, and much less code-switch n-grams.

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	516	-	50.11	-	23.77%	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	11.2	10.4
TrigWords+T	531	-2.9%	50.15	-0.08%	24.47%	6.7	6.9
TrigPOS+T	519	-0.6%	50.07	0.07%	24.34%	5.2	7.8

Table 5.26: Comparison of the baseline, *S&R+T*, *TrigWord+T*, and *TrigPOS+T* language models.

5.5.3.3 Target Segment Frequency Approach

In the next approach, we intend to generate better probabilities by adopting the number of the monolingual segments from *train*. This should result in similar probabilities for code-switch n-grams and monolingual n-grams in both texts. As it is unlikely to build a text with the exact same number of segments in the SMT-based texts and in SEAME, we use means to improve the search for reaching this target. Finding the exact same number of *train* segments in the base texts is not plausible. This would require the base text to be exactly the same as *train*, only translated to English or Chinese.

Text generation

We modified the *S&R* algorithm, such that each segment has an assigned target frequency. This target frequency is based on the number of segment occurrences in *train*. The latter and the monolingual texts differ in size. Thus we normalize the

number of segments with the total number of sentences per text. The normalized target segment frequency for each segment i is:

$$segment\ frequency_i = \frac{\#\text{segments i}}{\#\text{sentences}} \tag{5.6}$$

The segment frequency describes how often the segment occurs in the text, relative to the number of all sentences in the text. Those frequencies are precomputed before the search. They have to be computed separately for the English and Chinese monolingual texts. However, the differences in style and topic between the monolingual text and *train* suggest that the target for each segment cannot be reached.

The *target frequency (TF)* algorithm searches and replaces all found segments until the target frequency of each segment is reached. As it is important to reach the target frequency, we modify the segment search. Initially, the algorithm searches for the longest segment, than the second longest, et cetera. This is necessary because short segments are more frequent. If they are replaced first, they prohibit long sentences to be found. The order of the monolingual sentences is also randomized to make sure that some segments are not only replaced at the beginning of the corpus where one topic is prevalent, while different topics at the end of the text would be left without code-switching. The percentage of reached target replacements for the English text is 67% and 73% for the Chinese text. The resulting two language models from the English and Chinese texts are interpolated with the *train* language model (*TF+T*).

	#CS 2-grams	CS 2-gram cov.	#CS 3-grams	CS 3-gram cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
TrigPOS+T	569K	57%	2.1M	11%
TF+T	211K	52%	528K	11%
Rel. Δ	521%	44%	654%	38%

Table 5.27: Comparison of the code-switch n-gram coverages of *TF+T*, *TrigPOS+T*, and the baseline language models. Changes are relative from TF+T to the baseline.

Table 5.27 presents a comparison of the code-switch n-gram coverages of the *TF+T*, *TrigPOS+T*, and baseline language models. *TF+T* has a higher code-switch 2-gram (+44%) and 3-gram (+38%) coverage than the baseline and a lower code-switch 2-gram coverage than *TrigPOS*. The important aspect is that the number of code-switch n-grams is lower than in all previous SMT-based approaches, while the coverages are equally high for code-switch 2-grams (52%) and 3-grams (11%). The lower counts of code-switch n-grams and the high coverages should lead to more reliable code-switch n-grams.

Table 5.28 shows the evaluation results. The perplexity is 1% better than the baseline. The mix error rate is 0.26% better. The *TF+T* has also a better perplexity and mix error rate than *TrigPOS*, with slightly lower *ReCS2*. The language model weight is 36 and the word penalty parameter is 10. The ∅CSs for the monolingual English based and Chinese texts are 0.7 and 1.3. Therefore, they are much lower than in previous approaches and lower than in SEAME. The high code-switch coverages, few total code-switch n-grams, as well as the improvements in perplexity

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	516	-	50.11	-	23.77%	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	11.2	10.4
TrigWords+T	531	-2.9%	50.15	-0.08%	24.47%	6.7	6.9
TrigPOS+T	519	-0.6%	50.07	0.07%	24.34%	5.2	7.8
TF+T	511	1.0%	49.98	0.26%	24.03%	0.7	1.3

Table 5.28: Comparison of the *TF+T*, *TrigPOS+T*, and baseline language models. Changes are relative from TF+T to the baseline.

and mix error rate show that this is the best approach so far. Although, ReCS[2] is slightly lower and the language model weight is also relatively low.

5.5.3.4 Context-sensitive Segment Replacement Strategy with Target Segment Frequencies

The last three sections introduced methods for improving code-switch n-gram generation. The next step is to combine the best approaches. The *TrigPOS* and *TF* approaches showed the most promising results. *TrigPOS* used context information to improve the SMT-based text generation, whereas *TF* set target frequencies for the segment replacements.

Thus, we develop an algorithm which combines the *TrigPOS* approach with the *TF* approach. During the search and replacements, only those segments are replaced, which are preceded by a trigger. And the number of replacements are restricted to a target frequency for each segment. For this experiment the percentage of reached target replacements is 50% for the English-based text and 63% for the Chinese-based text. This is lower than in the *TF* approach.

	#CS 2-gr.	CS 2-gr. cov.	#CS 3-gr.	CS 3-gr. cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	*57%	2.9M	*11%
TrigPOS+T	569K	*57%	2.1M	*11%
TF+T	211K	52%	528K	*11%
TrigPOS&TF+T	*203K	48%	*442K	10%
Rel. Δ	497%	33%	531%	25%

Table 5.29: Comparison of the code-switch coverages of the *TrigPOS&TF+T*, *TF+T*, *TrigPOS+T*, and baseline language models. The best values of the artificial language models are marked with a "*".

Table 5.29 shows the code-switch 2-gram (3-gram) coverages of 48% (10%) of the experiments. The *TrigPOS&TF+F* leads to the lowest number of code-switch 2-and 3-grams. This low number results in lower code-switch 2-and 3-gram coverages than previous experiments.

Table 5.30 shows that *TrigPOS&TF+T* (LW=38, LP=10) achieves the best perplexity and mix error rate of all previous approaches. ReCS[2] was high, but slightly lower than for the *S&R* and *TrigPOS+T* experiments. The ∅CS rates are very

	PPL	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	516	50.11	-	23.77%	-	-
S&R+T	533	50.29	-0.36%	24.32%	11.2	10.4
TrigPOS+T	519	50.07	0.07%	*24.34%	5.20	7.8
TF+T	511	49.98	0.26%	24.03%	0.7	1.3
TrigPOS&TF+T	*510	*49.80	*0.62%	24.16%	0.5	1.2

Table 5.30: Comparison of the *TrigPOS&TF+T*, *TF+T*, *TrigPOS+T*, *S&R+T* and baseline PPLs and MERs, together with ReCS2. The best values are marked with a *.

low. The combination of the *TrigPOS* and *TF* approaches is important and leads to improvement of the language model.

Statistical Significance

TrigPOS&TF+T is the best approach so far. To judge the statistical significance of this approach, we conduct a statistical t-test [3]. This test shows a significance of 94.51% and an error probability of 5.49%. Therefore, this approach is not significant at the 5% significance level.

5.5.4 Human Evaluation

So far, we base our evaluation of the SMT-based language models on a quantitative evaluation of the results. For a better understanding of the problems with our new language models, we conduct a survey with humans to review the naturalness of our approach. People at Nanyang Technological University evaluated code-switch sentences from *train* and from the *TrigWords&TF* approach. We asked the students and researchers to rate each sentence on a scale of 1 to 5, where 5 means the sentence is natural and sounds similar to a human conversation. The score 3 is used for a neutral valuation, and 1 is unnatural and appears computer-generated. We also asked the participants to mark the sentences, where they seem wrong and were encouraged to give comments.

The six individuals participating in the survey are of Taiwanese, Chinese and Malay origin, who are fluent in Mandarin and English. The Malay participant is of Chinese roots. All participants use code-switching regularly, mostly at work. However, there are no Singaporean participants, which would have been considered to have both Mandarin and English as mother tongue.

The sentences are randomly chosen from *train* and from the *TrigWords&TF* text and mixed into a list of 30 sentences. From each text corpus are 15 sentences used. The selected sentences contains both short and long sentences.

Figure 5.31 illustrates that the sentences from *train* are not always rated natural. It can be seen that the sentence from *TrigWords&TF* are generally rated less natural than the ones of *train*. The participants rated 66% of *train* with 4 or 5 in contrast to 58% for *TrigWords&TF*. They also note that the sentences, especially the few short sentences, are difficult to evaluate without the context of the conversation. This may explain why the *train* sentences are not always rated as natural/human.

	not human/ not natural		neutral neutral		human/ natural
	1	2	3	4	5
Train	7%	7%	20%	23%	43%
TrigWords+TF	12%	14%	16%	22%	36%

Table 5.31: The evaluation of the participants shows that the SMT-based sentences are rated less natural than *train*.

The participants found one important problem in the SMT-based sentences: There are English segments embedded in Chinese sentences, which are mistranslated with regard to the context. The Chinese segments can have different meanings depending on the context. In some instances the words are translated to one possible meaning, which may be correct for a different context, but does not fit in the current context. This is also noted for embedded Chinese segments in Englist texts. The conclusion is that we need to take the context more into consideration. It is especially important to find the translations, which fit best for our code-switching.

5.5.5 Context-dependent Code-Switching Text Generation

We learned from the human evaluation that the meaning of some inserted segments does not fit to the context of the sentence. For example, there were segments of the monolingual Chinese text replaced by English SEAME segments, although the inserted English segments and its Chinese translations did not mean the same thing for the given context. But the translations could be correct for a different context. The problems lies in the translation: There are often different possible translations for one word or sentence, but the correctness depends on the context. Sometimes, words in one language can be translated to multiple words in another language.
To achieve a context fit improvement of the code-switches, we propose to find and use the best translation of the 5-best translations from the Moses SMT system with the *S&R* approach.

The first aspect is the improvement of the context fit to the inserted code-switching. A context fit improvement should be achieved by generating 5 versions of each sentence with the 5-best translations and finding the one with the best code-switching. Therefore, all 5 sentence versions are rescored and the best are used to build a new language model. The scoring will be discussed later. The ”best” code-switching can also mean no code-switching.
The second aspect is, that we only find a few unique translated segments in the monolingual texts. *S&R* has a used-segments-ratio of 8% for the English base text and a used-segments-ratio of 22% for the Chinese base text. This may stem from bad translations. If we improve the used-segments-ratio, this could result in more and better replacements and better code-switching in terms of coverages and probabilities.

So far, we use the 1st-best translation for the search. Our new approaches use multiple translations for the search. It is important to remember that the inserted segments are still the segments extracted from *train*.

First we use all 5-best translations and move to more refined variants, where we only used the best translations for each sentence or for each segment.

5.5.5.1 Preliminary 5-best Translation Experiment

The preliminary 5-best translation experiment (Pre 5-best) analyzes the impact of 5-best translations on *S&R*. The 5 translations are inserted in the same way as explained in Section 5.5.1. We expect higher code-switch n-gram coverages and high ∅CS rates.

The first step is to generate 5 English-based *S&R* texts from the 5-best translations. The monolingual segments extracted from *train* are translated and the 5-best translation variants are obtained. The i-th *S&R* text is built from the i-th best translations, with $i \in \{1, 2, 3, 4, 5\}$. 5 Chinese-based *S&R* texts are built analogously. After the text generation, the 5 English SMT-based texts are concatenated and one large language model is built. The same procedure is repeated for the 5 Chinese SMT-based texts. These 2 language models are interpolated with the *train* language model (*Pre 5-best+T*).

The code-switch coverages of *Pre 5-best+T* are given in Table 5.32. The 5 variants of the *S&R* texts increase both the code-switch 2-gram (3-gram) coverages by 100% (100%) to 72% (16%) in comparison to the baseline. The high coverages come at the cost of an increased number of code-switch 2-grams +6665% and 3-grams +17043% relative to the baseline.

	#CS 2-gr.	CS 2-gr. cov.	#CS 3-gr.	CS 3-gr. cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
Pre 5-best+T	2.3M	72%	12M	16%
Rel. Δ	6665%	100%	17043%	100%

Table 5.32: The 2-gram and 3-gram coverages of the *Pre 5-best+T* language model are higher than both *S&R* and the baseline language models. Changes are relative to the baseline.

The performance of the language model and the ASR system are shown in Table 5.33. We see a drop of 1.94% in perplexity, but a small improvement of 0.24% in mix error rate. ReCS[2] increases by 1.6% relatively compared to the baseline. The ∅CS rate of the English-based text is higher than *S&R+T* and the ∅CS rate of Chinese-based text is slightly lower. The language model weight is low at 36 and the word penalty is 10.

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	516	-	50.11	-	23.77%	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	11.2	10.4
Pre 5-best+T	526	-1.9%	50.00	0.24%	24.14%	12.4	10.2

Table 5.33: The *Pre 5-best+T* language model shows a worse PPL, and results in a better MER, with lower ReCS[2].

This approach generates many code-switches and shows better code-switch 2-gram and 3-gram coverages. The code-switch 2-gram coverage even reaches 72%. The high ∅CS rates indicates an overestimation of code-switching. We try to improve the 5-best language model by focusing on the best translations with regard to context and later on the combination with *TrigPOS&TF*.

5.5.5.2 Context-dependent N-best Translation Language Model

Hitherto, we used all 5-best translations for the search and did not differentiate between the quality of the translations or the context fit of the search segments to the context. We propose to find the variant of each code-switching sentence, which have the most appropriate code-switching. Further, these best variants are used to create language models.

We use the English-based and Chinese-based 5-best *S&R* texts from the last experiment. The approach is described for the English-based texts, and is analog for the Chinese-based texts.
Figure 5.3 shows the rescoring and selection process. The modified algorithm loops over all 5 sentence variants built from the 5-best translations of all sentences of the English-based SMT-text. The 5 sentence variants of each sentence are rescored and the variant with the best scores are saved. After that, the algorithm continues with the next 5 sentence variants. Subsequently, the file with the best sentences was used to build a new code-switching language model.

We use two approaches for finding the sentence variant with the best code-switching. In both setups is the perplexity used as the decision measure. First, we rescore the generated sentences with the *train* language model. Second, we use a part-of-speech *train* language model. The next two sections describe rescoring methods for selecting the sentences with the most appropriate code-switching and the resulting language models.

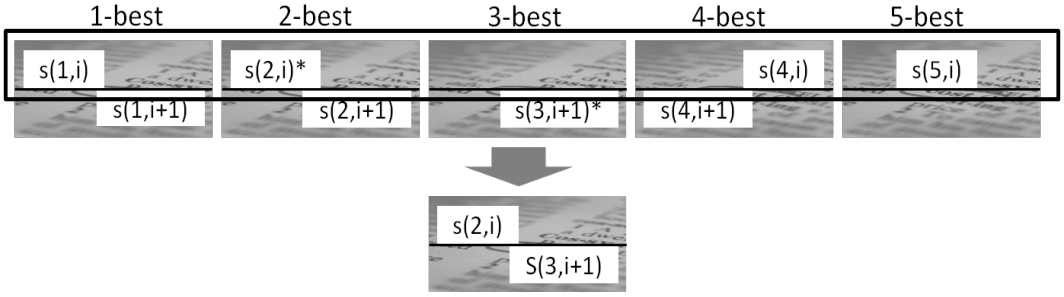

Figure 5.3: Each of the sentences of the 5 *S&R* texts is rescored. And the best are selected and saved. These sententence are used to build a new language model.

Train Language Model-based Best Sentence Selection

For building the code-switching text from the best translations per sentence, we apply the *train* language model for perplexity-based rescoring and selection of the best sentence. The 5 variants of the English and the 5-variants of the Chinese *S&R* texts form the basis. Analogously, we only describe the approach for one base text.

We propose an algorithm, which loops for each sentence over the 5 sentence variants, rescore each sentence, select and save the best sentence. Each sentence is rescored with a word-based 3-gram language model (wb) built from *train*. The sentence variant with the lowest perplexity is saved to a file. The resulting text contains all sentence variants with the lowest perplexities. The *train* language model is built from SEAME code-switching transcriptions, and is therefore suitable for rescoring our generated code-switching text. The resulting English- and Chinese SMT-based code-switch texts are used to generate language models, which are interpolated with the *train* language model (*Best(wb)+T*, wb for word-based).

Figure 5.4 depicts the percentage of sentences generated from the n-best (1st-, 2nd-,..., 5th-best) segment translations, which are used for the new language model. The Chinese-to-English segment translations are used for the English base text. The English-to-Chinese segment translations are used for the Chinese base text. The order of Moses n-best translations from 1 to 5 corresponds to the importance rated by Moses, where 1 is the best. Thus, sentences generated from a translation with a low n are expected to be better and used more often.
The distribution of the sentences shows that according to our rescoring, the sentences using the 1st-best translations are not always the ones, which lead to a low perplexity. For the sentences using the Chinese-to-English translations, the 1st-best translations are not the ones which are used most. Nevertheless, the trend that lower n-best translations with a smaller n are used more often than higher n translations is revealed: For example, the 4th-best and 5th-best translations are used less than the 2nd-best or 3rd-best. For the sentences using the English-to-Chinese translated segments, the 1st-best translations are used most. The trend to a higher use of translations from lower n, can be seen, too.

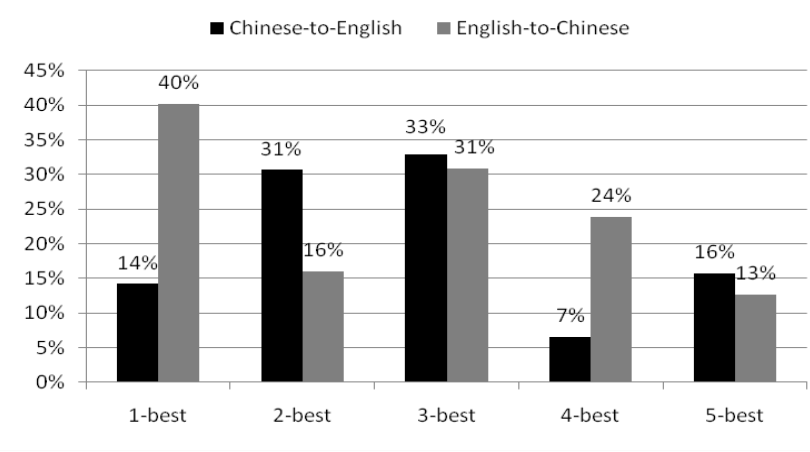

Figure 5.4: Percentages of sentences using the i-th-best translations. Recored and selected with the *train* language model.

Table 5.34 presents the code-switch n-gram coverages. The numbers of code-switch n-grams of the *Best (wb)+T* are higher than for *S&R*. The coverages are higher than for the baseline and higher than *S&R*, which uses only the 1st-best translation. Nevertheless, the 2-gram coverage of *Best (wb)+T* is 11% absolute lower than *Pre 5-best+T*. The advantage of *Best (wb)+T* is that the high coverages are achieved without the high numbers code-switch n-grams of *Pre 5-best+T*.

	#CS 2-gr.	CS 2-gr. cov.	#CS 3-gr.	CS 3-gr. cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
Pre 5-best+T	2.3M	72%	12M	16%
Best(wb)+T	890K	61%	3.1M	12%
Rel. Δ	2518%	72%	4329%	50%

Table 5.34: The coverages of the *Best(wb)+T* in comparison to *S&R+T* and baseline language models. Changes are relative to the baseline.

Table 5.35 reveals that the perplexity of *Best(wb)+T* drops 2.1% to 527, compared to the baseline with 516. However, *Best(wb)+T* performs better than the 1st-best using *S&R+T*. The mix error rate of *Best(wb)+T* improves in relation to the baseline and *S&R*. ReCS[2] increases and is higher than in the last experiment and *S&R+T*. The language model weight is 38 (LP=8).

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	516	-	50.11	-	23.77%	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	11.2	10.4
Pre 5-best+T	526	-1.9%	50.00	0.22%	24.14%	12.4	10.2
Best(wb)+T	527	-2.1%	50.05	0.13%	24.59%	12.5	10.2

Table 5.35: Comparison of the *Best(wb)+T*, *Pre 5-best+T*, and baseline language models

The use of different n-best translations shows improvement over the baseline and (1st-best) *S&R* in terms of code-switch 2-gram and 3-gram coverage. The improvement of the language model over *S&R* is revealed by a better perplexity. However, the perplexity is lower than of the baseline. Besides, the MER improves from the baseline by relative 0.13% and from the *S&R* by relative 0.48%. ReCS[2] improves by 3.5% relative from the baseline. We analyze another scoring and selection approach in the next experiment, since the scoring is the crucial step of this approach.

Part-of-Speech Language Model-based Best Sentence Selection

The rescoring and selection of the 5 *S&R* is critical for the improvement with n-best translations of the language model. Our second rescoring method evaluates all sentences with a 3-gram *train* part-of-speech (POS) language model. We tag *train* with Burgmers [6] part-of-speech tagger. The output is a text file, where each word is followed by a POS tag. From the tagger output we remove the words, which resulted in POS sequences. They are used to build a language model. This model has the advantage, that it uses more general, linguistically motivated tags and can better cope with unseen n-grams. This is useful, since we generate new n-grams. Those are expected to be difficult to score for the *train* language model.

Our approach found the sentences with code-switching, which fits best to the context. The procedure is similar to the previous one: First we run the POS tagger on the 5 *S&R* texts. After the tagging, we remove the words from the text. This results

in a file only containing the POS tags of the *train* sentences. Second, we compute the perplexities of all POS tag sequences with the *train* POS language model. Third, we select the n-best sentence variant with the lowest perplexity for each sentence. Finally we concatenate the best sentences to a new text.
From the resulting English-and Chinese SMT-based code-switching texts we compute the language models, which are interpolated with the *train* language model (*Best(pb)+T*, pb for POS-based).

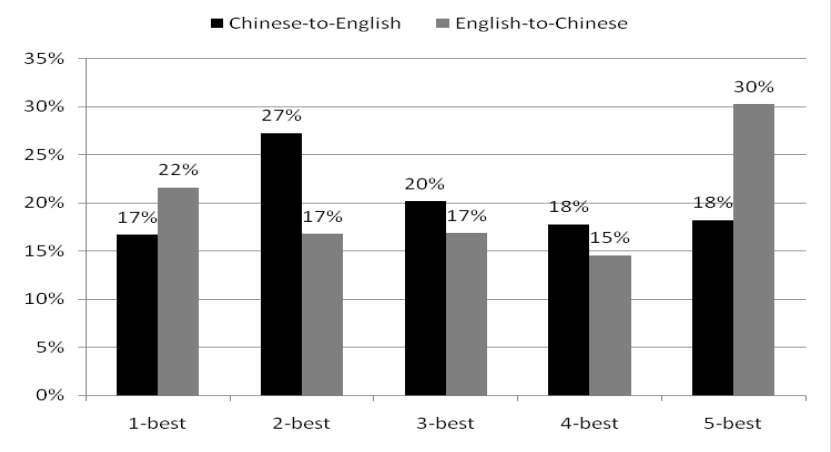

Figure 5.5: Percentages of sentences with the lowest PPLs using the n-best translations.

Figure 5.5 illustrates the percentages of sentences, which are build from the 1st-best, 2nd-best,..., 5th-best segment translations, and used after the rescoring. For sentences which use the Chinese-to-English translations the 1st-best translation lead not always to the best perplexity. The sentences using the remaining translations reveal that the ones which are build from a lower i-th best ($i \in 2, 3, 4, 5$) translation have lower perplexities than those with higher i-th best translations: The English-based *S&R* sentences built from the 2nd-best translations has the lowest perplexity of all n-best variants 27% of the time. This is more often than the 3rd-best (20%), 4th-best (18%) and 5th-best (18%). The usage of English-to-Chinese n-best translations illustrates the same trend, that the i-th-best translations are better than (i+1)th-best ($i \in 1, 2, 3, 4$). The sentences built from the 5th-best translations, in other words the worst, show an outlying high usage.

	#CS 2-gr.	CS 2-gr. cov.	#CS 3-gr.	CS 3-gr. cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
Best(wb)+T	890K	61%	3.1M	12%
Best(pb)+T	970K	62%	3.6M	12%
Rel. Δ	2753%	72%	5043%	50%

Table 5.36: The 2-gram coverage of the *best(pb)+T* language model is higher than *S&R+T*, *best(wb)+T* and the baseline language models. The number of code-switch n-grams is higher than for *best(wb)+T*. Changes are relative to the baseline.

Table 5.36 shows that the 2-gram coverage of *best(pb)+T* is higher than the baseline and the *best(wb)+T* language models. The number of code-switch n-grams increases compared to the baseline and *Best(wb)+T*.

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	*516	-	50.11	-	23.77%	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	*11.2	10.4
Best(wb)+T	527	-2.1%	*50.05	*0.13%	*24.59%	12.5	*10.2
Best(pb)+T	528	-2.3%	50.19	-0.16%	24.16%	12.9	10.6

Table 5.37: *Best(pb)+T* has a worse PPL and MER than *Best(wb)+T*. Changes are relative to the baseline. The best values are marked with a "*".

Table 5.37 illustrates that the perplexity and the mix error rate of *Best(pb)+T* (LW=38, LP=10) are worse than the baseline. *Best (wb)+T*. ReCS2 shows no improvement over *Best(wb)+T*. The *train* language model appears to be more suitable for the rescoring of the generated code-switch sentences than the POS language model.

A drawback of the POS language model-based rescoring approach is that the scoring is based on a POS tagger, which is not directly trained on Chinese-Englisch code-switching texts. The tagger is instead based on a combination of monolingual English and monolingual Chinese tagger.

The usage pattern of sentences rescored with the POS language model (Figure 5.5) shows a weaker trend than the usage of the translations scored with the *train* language model 5.4. This is especially the case for the English-to-Chinese translations: The *train* language model-rescored sentences show the trend of a decreasing use of n-best translations. In contrast, the usage pattern of the POS language model-rescored sentences resemble an equal distribution, which means that all translations are equally good. This is contrasting to the results from the *train* language model 5.5, where the trend that a lower number indicates a better translations was shown. We believe that a preference of lower n-best translations over higher n-best translations makes sense, e.g. 1st-best preferred to 5th-best, which is in line with the Moses SMT system. The lack of the trend, together with the worse perplexity, mix error rate and ReCS[2] leads to the conclusion, that the word-based language model rescoring are more suitable for rescoring than the POS-based language model.

In the last two experiments we improved the *S&R* approach with n-best translations. In earlier experiments, we improved *S&R* with a restriction of the segment replacements. Therefore, we combine the n-best translation-based experiment with earlier segment replacement strategies to see if we can achieve further improvement. There are two methods for combining the n-best translation *S&R* with the best segment replacement strategy *TrigPOS&TF*.
The first would be to run the rescoring not on the 5-best variants of *S&R*, but on 5 n-best variants of *TrigPOS&TF*. The shortcoming would be, that the target frequencies could be breached. The reason is that in this case we would limit the number of segment replacements for *each* of the 5 *TrigPOS&TF* texts. A combination of sentences from the 5 *TrigPOS&TF* to a new text can result in segments used too often, and therefore violating the target number of replacements. This would be a

step back from the *TrigPOS&TF* advancements.
The second method for combining the n-best translation *S&R* with *TrigPOS&TF* is to use the word-based rescoring approach to find the best translations per segment, and not for each sentence. These segments are then used to improve the 1st-best *TrigPOS&TF*.

5.5.5.3 Context-dependent N-best Translation Language Model combined with Advanced Segment Replacement Strategies

We combine the advantages of the 5-best *S&R* approach with the *TrigPOS&TF* approach. In the experiment before, we analyzed which sentence had the best code-switching and which translations a responsible. In this experiment, we investigate which segments are responsible for better code-switching. For *TrigPOS&TF* it was necessary to know the best segments, since we needed to set target number of replacements per sentence. We use the 5-best *S&R* SMT-based texts from the experiment before and evaluate the scores of each translated segment. The *train* language model is used for rescoring as it gave better results in earlier experiments.

For the scoring, we use a 2 dimensional array with 5 entries in the first dimension and the number of monolingual segments as the second dimension. Each row in the array corresponds to the 5 translations of 1 segment. All values are set to zero in the beginning. We rescore each segment separately. The final list of best segment translations, consisting of 1st-best, 2nd-best, et cetera, is used to create *TrigPOS&TF*. We describe the procedure for one SMT-based text in the following paragraphs:
Our algorithm loops over the 5 sentence variants of the SMT-based text and selects the variant with the lowest perplexity. In the selected sentence, all replaced segments are identified. For these segments the array IDs are looked up, and the score for each segment is increased by 1. This leads to an array with scores for all translated segments.
Subsequently, the algorithm loops over all array entries, and saves the translations with the highest scores. Figure 5.6 shows the used segments. In some cases, no n-best translation of a segment is scored. These segments are discarded. The Chinese-to-English translations shows a light trend, that the i-th best translation is better than the (i+1)th-best translation. Nevertheless, the 1st-best translations are not covered by this trend.
For the English-to-Chinese translations the trend is the same. But the outliers in this case are the 5th-best translations. These best translations together with the corresponding segments are used to pursue the *context-sensitive segment replacement strategy with target segment frequencies* from Section 5.5.3.4. From the resulting English and Chinese-based texts, language models are built and interpolated with the *train* language model (*BestSeg(wb)+T*). This language model is built exactly as *TrigPOS&TF*, but with the best translations.

We also want to see if better translations improve the used-segment-ratio (USR). The use of better n-best translations together with a high used-segments-ratio indicates that bad translations have been the reason why only a few segments are found. In the case of a low used-segment-ratio it is possible that the style of the segment does not fit to the base texts.
Table 5.38 shows that the used-segments-ratio increased for both English- and

Chinese-based texts: 21% absolute for the English case and 3% for the Chinese case. The new used-segments ratios are still low, which suggested, that different base texts should be analyzed.

	English-based	Chinese-based
S&R	8%	22%
BestSeg(wb)	26%	25%

Table 5.38: Used-segments-ratio (USR) of the English- and Chinese-based S&R and BestSeg(wb) texts.

It is important that the *BestSeg(wb)+T* language model is built from the n-best translations, which are scored together with their context. This context is lost, when building a new *TrigPOS&TF* language model with the n-best translations. We use the rescoring approach to find the best SMT translation.

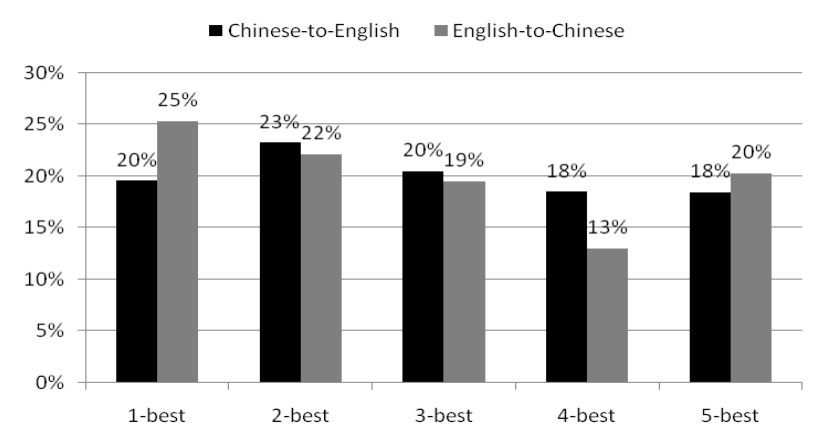

Figure 5.6: Percentages of the used n-best segment translations.

Table 5.39 shows that the n-gram coverages of *BestSeg(wb)+T* are higher than the ones of the baseline. The code-switch 2-gram coverage is also higher (2% absolute) than *TrigPOS&TF+T*, which uses only the 1st-best translations. However, the code-switch n-gram coverages are lower than for *S&R* and *S&R* with n-best translation approaches.

Table 5.40 illustrates the performance of the *BestSeg(wb)+T* language model (LW=38, LP=10): The perplexity improves against the baseline and is the same as *TrigPOS&TF*, which uses only the 1st-best translations. The word error rate is better than the earlier 5-best experiments, but worse than *TrigPOS&TF*. The ∅CS rates show that the current approach estimated more code-switching than *TrigPOS&TF*.

This strategy uses a context-dependent rescoring for segments to find the best segment translations. The inclusion of these different translated segments in *BestSeg(wb)+T* does not show improvement in perplexity or word error rate over 1st-best *TrigPOS&TF*. But in terms of code-switch coverage and perplexity the scoring

	#CS 2-gr.	CS 2-gr. cov.	#CS 3-gr.	CS 3-gr. cov.
Baseline	34K	36%	70K	8%
S&R+T	689K	57%	2.9M	11%
TrigPOS&TF+T	203K	48%	442K	10%
Best(wb)+T	890K	61%	3.1M	12%
Best(pb)+T	970K	62%	3.6M	12%
BestSeg(wb)+T	231K	50%	552K	10%
Rel. Δ	579%	39%	689%	25%

Table 5.39: The 2-gram coverage of the *BestSeg(wb)+T* language model is higher than *TrigPOS&TF+T* with only 1st-best translations. Changes are relative to the baseline.

	PPL	Rel.Δ	MER	Rel.Δ	ReCS[2]	∅CS(E)	∅CS(C)
Baseline	516	-	50.11	-	23.77%	-	-
S&R+T	533	-3.3%	50.29	-0.36%	24.32%	11.2	10.4
TrigPOS&TF+T	510	1.2%	49.80	0.62%	24.16%	0.5	1.2
Best(wb)+T	527	-2.1%	50.05	0.13%	24.59%	12.5	10.2
Best(pb)+T	528	-2.3%	50.19	-0.16%	24.16%	12.9	10.6
BestSeg(wb)+T	510	1.2%	49.89	0.45%	23.99%	0.9	1.4

Table 5.40: Comparison of *BestSeg(wb)+T* to *Best(wb)+T*, *Best(pb)+T*, *TrigPOS&TF+T*, and baseline language model. Changes are relative to the baseline.

approach is superior. The used-segments-ratio shows a low number of found segments. As we improve the segment translations, the used-segments-ratio remains relatively low. This suggests the analysis of different base texts with different styles.

5.5.6 Style-motivated Integration of supplemental Monolingual Texts

During the last experiments, we found that between 8% and 26% of the translated segments were found during the search. Additionally, the styles of the English NIST text (from newswire) and SEAME (conversations) were quite different. This suggests to look for more appropriate texts, in terms of style. We discuss various monolingual texts and combine them with earlier approaches. Table 5.41 gives an overview of the different texts in terms of size. The Chinese GALE and English NIST subsets have already been introduced in Section 4.1. The IWSLT and BTEC texts contain spoken dialogues from travel situations. The Wall Street Journal corpus (WSJ)[36] contains text from the homonymous newspaper. The style is newswire. The text is carefully selected and normalized. The Quaero text is from European parliament speeches and web data, for example podcasts. To gather more spontaneous speech transcriptions, we develop a web crawler for DVD subtitles. We crawl 178 English movie subtitles from a website.

The English texts has to be cleaned and normalized before they can be used. For example, most of the non-character and non-digit characters are removed. The SEAME corpus does not have any punctuation.

Chinese texts	#sentences	#words	English texts	#sentences	#words
NIST	8.4M	228M	NIST	8.4M	255M
NIST Subset	348K	820K	NIST Subset	345K	9.2M
BTEC	20K	152K	BTEC	20K	153K
IWSLT	10K	89K	IWSLT	10K	118K
GALE Subset	395K	9.6M	WSJ	1.6M	37M
			Quaero	48K	1.3M
			DVD Subtitles	225K	954K

Table 5.41: Comparison of monolingual Chinese and English base texts.

We analyze the fit of different monolingual language models for SEAME. This is done by computing the perplexity of monolingual language models on the development text. For this reason, we build language models from monolingual English and Chinese texts. The vocabulary of the language models is restricted to the *train* vocabulary. Then all language models are evaluated on the SEAME development text.

We do not show the results of the perplexities of the language models on the monolingual segments from the SEAME development text: First, because we focus on the improvement of code-switching, and second the monolingual segments are very short. Most of them are only one word long.

Figure 5.7 shows the evaluation of the English BTEC, IWSLT, NIST subset, WSJ, Quaero, and DVD subtitle language models. The out-of-vocabulary rates are very high, since the development text contains code-switching. In the development text are more Chinese than English words, which was shown in Section 4.4. Those Chinese words are not included in our English language models. The NIST subset, which we used so far, had the highest perplexity. Text from newswire has a different style than conversational SEAME. This suggests to conduct our experiments with different English base texts.
Figure 5.8 shows the evaluation of the Chinese BTEC, IWSLT, NIST subset, and GALE subset language models on the development text. The out-of-vocabulary rates are high, because the development text also contained English. The GALE subset, which we use in our experiments, show a low perplexity compared to the NIST language model. The reasons are the different styles of NIST, GALE and SEAME. GALE contains conversational texts, whereas NIST contains text from newswire. The BTEC and IWSLT language models show lower perplexity, but are built from very little data: BTEC has 20K sentences and IWSLT has 10K sentences. Therefore, we decide not to use any different Chinese base texts.

Next, we generate *TrigPOS&TF* texts, based on all monolingual English texts, except for the NIST subset. From the resulting SMT-based texts language models are built. These language models are interpolated with *TrigPOS&TF*, based on GALE, which was already described in Section 5.5.3.4 and the *train* language model. This new language model, based on multiple English texts, is called *TrigPOS&TF&T(multEN)*. Since we use more monolingual English texts, we generate a new baseline language model, interpolated from the *train*, GALE subset, and the BTEC, IWSLT, WSJ, Quaero, and DVD subtitle language models.

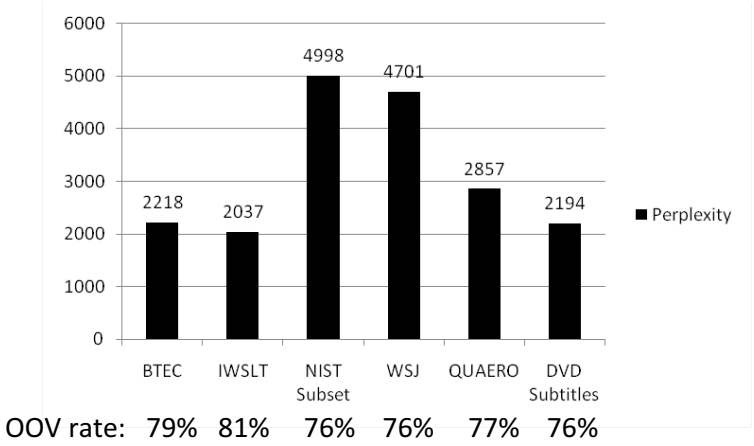

Figure 5.7: PPLs of monolingual English LMs evaluated on the dev. text.

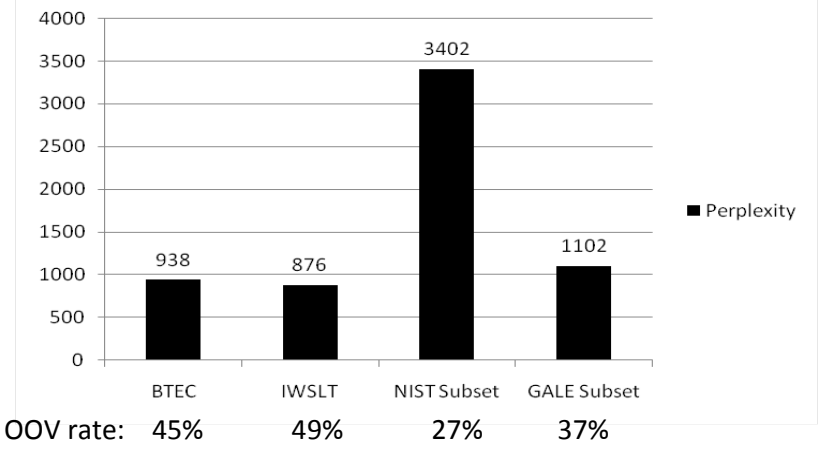

Figure 5.8: PPLs of monolingual Chinese LMs evaluated on the dev. text.

	PPL	MER	ReCS[2]
Baseline	516	50.11	23.77%
TrigPOS&TF+T	510	49.80	24.16%
Baseline(multEN)	514	49.84	24.26%
Rel.Δ to baseline	0.39%	0.55%	2.1%
TrigPOS&TF+T(multEN)	507	49.75	24.03%
Rel.Δ to baseline	1.74%	0.72%	1.1%
Rel.Δ to baseline(multEN)	1.36%	0.18%	-0.9%

Table 5.42: Comparison of the baseline, *TrigPOS&TF+T*, the baseline with multiple English texts, and *TrigPOS&TF+T(multEN)+T* language models.

Table 5.42 presents the perplexities and mix error rates of the baseline, *TrigPOS&TF+T*, the baseline with multiple English texts, and *TrigPOS&TF(multEN)+T* language models. *TrigPOS&TF(multEN)+T* gives the best improvement in terms of perplexity (1.74%) and mix error rate (0.72%) to the baseline. The baseline with multiple English texts results in an improvement of 0.55% over the old baseline, which is slightly worse than the 0.62% improvement of *Trig-*

POS&TF+T language model.
The improvement of the *TrigPOS&TF* approach to the baseline of 0.62% drops to 0.18% from *TrigPOS&TF(multEN)* to the new baseline.
This experiment illustrates that more monolingual texts - especially with the same style - improves perplexity and mix error rate on the SEAME code-switching development data. A collection of more English and Chinese texts with conversational style similar to SEAME should be collected to improve the language model. The *TrigPOS&TF* approach shows an improvement over the new baseline, but less than for the experiments with one English-based language model.

6. Conclusion

This work addressed the problem of language modeling for Chinese-English code-switching speech. We introduced the fundamentals of speech recognition and statistical machine translation. For a better understanding of the problems of language modeling for code-switching speech, we presented an overview of the research in this field.

At the beginning, we conducted several analyses of the SEAME corpus. They showed that 59% of the words in the corpus were Chinese and 33% were English. The average number of code-switches per sentence was 2.4 times. We also found variations between the mix error rates of different speakers. Additionally, there were 8% tagged foreign words, proper nouns and particles. Their recognition rates were poor. Most importantly, we showed that 2-grams, which contained code-switching, were a challenge for the baseline system, since only 25% were recognized correctly. A further analysis showed that a homogenous Chinese word segmentation, generated with the Stanford Chinese Word Segmenter, of the language model and development text led to a 32% relatively lower perplexity compared to the baseline segmentation. Surprisingly, no improvement in mix error rate was obtained.

For topic analysis, we employed the TF-IDF measure to assign topics to all speaker transcriptions. Some topics revealed differences in code-switching behaviour. For example, the topics *internet* or *university* indicated more code-switching than the topic *family*.

We used two kinds of class-based language models for code-switching speech recognition. First, we analyzed automatically-induced class language models. Second, we built part-of-speech language models. Both approaches showed much higher code-switch n-gram coverages, but this did not result in lower mix error rates.

We proposed methods to combine code-switching text with large monolingual corpora to build new code-switching texts and language models. For our main approach, *Search and Replace*, we extracted all monolingual English and Chinese segments from the SEAME training text. The extracted English segments were translated to Chinese and the extracted Chinese segments were translated to English. These English translations of Chinese segments were then searched in a monolingual English

text corpus and the found segments were replaced by the original Chinese segments. English segments were identically inserted into a Chinese text. This method generated a high *code-switch coverage*, but did not result in perplexity or mix error rate improvements.

For the evaluation we used the measures perplexity, mix error rate, *∅CS rate* and introduced the new measures *number-of-replaced-segments*, *used-segments-ratio*, *code-switch n-gram coverage*, and *recognized code-switch n-grams*.
All of our experiments were compared to a baseline language model, built from the SEAME training transcriptions, interpolated with monolingual English and Chinese texts.
Further probability modifications from Topkara et al. [50] were applied without improvements. The *Search and Replace* approach was advanced with multiple segment replacement restrictions: We replaced only segments which occured frequently in the training text. Further, we replaced segments only after code-switch trigger words or trigger part-of-speech tags. Additionally, the number of replacements per segment was limited to an upper bound, which was derived from *train*. The mix error rate increased by 0.62% relative from the baseline, which is not statistical significant.

To better understand the generated code-switching texts, we carried out a human evaluation study at Nanyang Technological University. The results showed that the translations did sometimes not fit to the context.
Therefore, we proposed a method to find the translations which fit better to the context. The fit was evaluated with word-based and part-of-speech based language models. This method was also combined with the advanced segment replacement strategies. We achieved a relative improvement of 0.45% to the baseline.
The final experiment used supplemental English texts together with advanced replacement strategies. The different English texts contained more conversational text than the first experiments. This resulted in a 0.72% relative improvement to the baseline generated from *train*, and monolingual English NIST and Chinese GALE language models. For a correct comparison to a baseline, we built a new baseline from *train*, Chinese GALE and all of the used English texts. In comparison to the new baseline, this last language model showed an improvement of 0.18%.

The proposed approaches showed improvements in terms of *code-switch n-gram coverage*, perplexity, mix error rate and *recognized code-switch 2 grams*. However, it would be interesting to see further strategies to insert natural code-switching into monolingual text corpora.

7. Recommendations

There are multiple ways to improve the performance of code-switching language models. Since SEAME contains spontaneous utterances, the speech recognizer could be enhanced with support for word fragments.

The analysis of the topics indicates that there are differences in code-switching between different topics. Therefore, topic-dependent code-switching language models with different ∅CS rates could also lead to an improvement.

Our proposed approach generates a high code-switching coverage. But there is room for improving the probabilities. The existing part-of-speech language model could be improved by a part-of-speech tagger, solely trained on code-switching data. As our system relies on a statistical machine translation system, one could improve this system to increase the translation quality. Another way to enhance our approach is to improve the scoring method for the context-fit of the code-switches. New methods could be developed. The succesful inclusion of monolingual texts can be extended by crawling colloquial English or Chinese from Singaporean news sites or web forums.

A further step would be to generate code-switching texts without the segments from SEAME. This could be achieved by directly translating parts of monolingual texts. The knowledge which parts to translate is crucial for this idea. Code-switches to English or Chinese could be motivated by linguistic features, e.g. part-of-speech trigger.

A large monolingual language model could be used as base to build a CS language model, where monolingual n-grams are converted to code-switching n-grams with statistical machine translation. For each monolingual 2-gram one could generated 2 code-switch 2-grams. The first 2-gram could be created by translating the first word of the monolingual 2-gram and the second 2-gram by translating the second word. The translated words must be added to the 1-gram list of the language model. The probabilities for both 1-and 2-grams could be used from the corresponding monolingual n-grams. A similar strategy could be executed for 3-grams or other higher order n-grams. The final language model would contain the original monolingual n-grams, the new code-switch n-grams ($n > 1$) and the new 1-grams.

Bibliography

[1] I. Bazzi, J. Glass, and A. C. Smith. Modeling Out-Of-Vocabulary Words For Robust Speech Recognition. *Proc. Interspeech*, Beijing, China, 2000.

[2] N. Bertoldi, B. Haddow, and J. Fouet. Improved Minimum Error Rate Training in Moses. *Prague Bulletin of Mathematical Linguistics*, No. 91:7–16, 2009.

[3] J. Bortz. Statistik für Human-und Sozialwissenschaftler. Springer, 2005.

[4] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. Class-based n-gram models of natural language. volume 18, pages 467–479. MIT Press, December 1992.

[5] B. E. Bullock and A. J. Toribio. The Cambridge Handbook of Linguistic Code-switching. Cambridge University Press, 2009.

[6] C. Burgmer. Detecting Code-Switch Events based on Textual Features. 2009. Diploma thesis, CSL, KIT.

[7] H. Cao, P. Ching, T. Lee, and Y. T. Yeung. Semantics-based language modeling for Cantonese-English code-mixing speech recognition. In *Proc. International Symposium on Chinese Spoken Language Processing*, pages 246–250, Nantou, Taiwan, 2010.

[8] J. Y. C. Chan, P. Ching, and T. Lee. Development of a Cantonese-English Code-mixing Speech Corpus. *Proc. Interspeech*, Lisboa, 2005.

[9] J. Y. C. Chan, P. Ching, T. Lee, and H. Cao. Automatic speech recognition of Cantonese-English code-mixing utterances. *Proc. Interspeech*, Pittsburgh, PA, 2006.

[10] P. Chang, H. Tseng, and G. Andrew. Stanford Chinese Segmenter. 2008.

[11] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proc. Annual meeting on Association for Computational Linguistics*, pages 310–318. Association for Computational Linguistics, Santa Cruz, California, 1996.

[12] M. Finke, P. Geutner, H. Hild, T. Kemp, K. Ries, and M. Westphal. The Karlsruhe-Verbmobil speech recognition engine. In *Proc. International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 83–86, Munich, Bavaria, Germany, April 1997.

[13] J. France and T. Solorio. Baby-Steps Towards Building a Spanglish Language Model. In *Proc. International Conference on Computational Linguistics and Intelligent Text Processing*, pages 75–84. Springer-Verlag, Mexico City, Mexico, 2007.

[14] C. Fügen, S. Stüker, H. Soltau, F. Metze, and T. Schultz. Efficient Handling of Multilingual Language Models. In *Proc. Workshop of Automatic Speech Recognition Understanding*, pages 441–446. ASRU, St. Thomas, 2003.

[15] J. Garofolo. NIST Open Machine Translation (OpenMT) Evaluation. http://www.itl.nist.gov/iad/mig/tests/mt/, 2011.

[16] J. Gebhardt. *Speech Recognition on English-Mandarin Code-Switching Data using Factored Language Models*. 2011. Diploma thesis, CSL, KIT.

[17] J. T. Goodman. A bit of progress in language modeling. *Computer Speech and Language*, 15(4):403 – 434, 2001.

[18] J. Gumperz. *Discourse Strategies*. Cambridge University Press, 1982.

[19] M. Higa. Sociolinguistic Aspects of Word Borrowing. *Topics in Culture Learning*, pages 75–85, 1973.

[20] B. C. Hok-shing. Code-Mixing in Hongkong Cantonese-English Bilinguals: Constraints and Processes. *CUHK Papers in Linguistics*, pages 1–24, 1993.

[21] R. Hsiao, M. Fuhs, Q. Jin Y. Tam, and T. Schultz. The CMU-InterACT 2008 Mandarin transcription system. In *Proc. International Conference on Acoustics, Speech and Signal Processing*. IEEE, Las Vegas, 2008.

[22] X. Huang, A. Acero, and H. Hon. *Spoken language processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall, 2001.

[23] K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21, 1972.

[24] D. Klakow and J. Peters. Testing the correlation of word error rate and perplexity. *Speech Communication*, 38:19–28, September 2002.

[25] R. Kneser and J. Peters. Semantic clustering for adaptive language modeling. In *Proc. International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 779–782. IEEE, April Munich, Bavaria, Germany, 1997.

[26] R. Kneser and V. Steinbiss. On the dynamic adaptation of stochastic language models. In *Proc. International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 586–589. IEEE, Minneapolis, Minnesota, April, 1993.

[27] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Frederico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: open source toolkit for statistical machine translation. In *Proc. Annual Meeting of the Association for Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics, 2007.

[28] D. C. S. Li. Cantonese-English code-switching research in Hong Kong: a Y2K review. 19:305–322, 2000. Special issue: Hong Kong English: Autonomy and Creativity.

[29] D. Lyu, R. Lyu, Y. Chiang, and C. Hsu. Speech Recognition on Code-Switching Among the Chinese Dialects. *Proc. International Conference on Acoustics, Speech and Signal Processing*, Toulouse, France, 2006.

[30] D. Lyu, T. Tan, E. Chng, and H. Li. SEAME: a Mandarin-English Code-Switching Speech Corpus in South-East Asia. *Proc. Interspeech*, Makuhari, Japan, 2010.

[31] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*, 19:313–330, June 1993.

[32] P. Muysken. *Bilingual Speech: A Typology of Code-mixing*. Cambridge University Press, 2000.

[33] F. J. Och. Minimum error rate training in statistical machine translation. In *Proc. Annual Meeting on Association for Computational Linguistics*, pages 160–167. Association for Computational Linguistics, Sapporo, Japan, 2003.

[34] J. Olive. Global Autonomous Language Exploitation. http://www.darpa.mil/Our_Work/I2O/Programs/Global_Autonomous_Language_Exploitation_(GALE).aspx, 2011.

[35] K. Papineni, S. Roukos, T. Ward, and W. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proc. Annual Meeting on Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics, Philadelphia, PA, 2002.

[36] D. B. Paul and J. M. Baker. The design for the wall street journal-based CSR corpus. In *Proc. Workshop on Speech and Natural Language*, pages 357–362. Association for Computational Linguistics, Harriman, New York, 1992.

[37] S. Poplack. *Sometimes I'll start a sentence in Spanish y termino en español: toward a typology of code-switching*. Mouton de Gruyter, 1979.

[38] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proc. IEEE*, pages 257–286, 1989.

[39] S. Romaine. *Bilingualism*. Blackwell, 1995.

[40] R. Rosenfeld. Two decades of statistical language modeling: where do we go from here? *Proc. IEEE*, 88(8):1270 –1278, August 2000.

[41] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513 – 523, 1988.

[42] T. Schultz and K. Kirchhoff. *Multilingual Speech Processing*. Academic Press, May 2006.

[43] D. Snow. *Cantonese as written language: the growth of a written Chinese vernacular*. Hong Kong University Press, 2004.

[44] T. Solorio and Y. Liu. Learning to predict code-switching points. In *Proc. Conference on Empirical Methods in Natural Language Processing*, pages 973–981. Association for Computational Linguistics, Honolulu, Hawaii, 2008.

[45] T. Solorio and Y. Liu. Part-of-speech tagging for English-Spanish code-switched text. In *Proc. Conference on Empirical Methods in Natural Language Processing*, pages 1051–1060. Association for Computational Linguistics, Honolulu, Hawaii, 2008.

[46] A. Stolcke. SRILM - An Extensible Language Modeling Toolkit. In *Proc. International Conference on Spoken Language Processing*, Denver, Colorado, September, 2002.

[47] K. Toutanova and C. D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proc. SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, pages 63–70. Association for Computational Linguistics, Hong Kong, 2000.

[48] T. Tsai, C. Chiang, L. Lo H. Yu, Y. Wang, and S. Chen. A study on Hakka and mixed Hakka-Mandarin speech recognition. In *Proc. International Symposium on Chinese Spoken Language Processing*, pages 199–204, Nantou, Taiwan, 2010.

[49] H. Tseng. A conditional random field word segmenter. *Fourth SIGHAN Workshop on Chinese Language Processing*, Jeju Island, Korea, 2005.

[50] Z. Wang, U. Topkara, T. Schultz, and A. Waibel. Towards Universal Speech Recognition. In *Proc. International Conference on Multimodal Interfaces*, pages 14–16. IEEE, Pittsburgh, PA, 2002.

[51] F. Xia. *The Part-Of-Speech Tagging Guidelines for the Penn Chinese Treebank (3.0)*. 2000.

[52] C. F. Yeh, C. Y. Huang, L. C. Sun, and L. S. Lee. An integrated framework for transcribing Mandarin-English code-mixed lectures with improved acoustic and language modeling. In *Proc. International Symposium on Chinese Spoken Language Processing*, pages 214–219, Nantou, Taiwan, 2010.

[53] S. Young. Large Vocabulary Continuous Speech Recognition: a Review. *IEEE Signal Processing Magazine*, pages 45–57, 1996.

[54] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK Book*. 2009.