# A Text Rewriting Decoder with Application to Machine Translation

**Pidong Wang**

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the School of Computing

**NATIONAL UNIVERSITY OF SINGAPORE**

2013

©2013

Pidong Wang

All Rights Reserved

# Declaration

This thesis is an account of research undertaken between August 2008 and August 2013 at the Department of Computer Science, School of Computing, National University of Singapore.

I declare that this thesis is the result of my own research except as cited in the references. This thesis has not been submitted in candidature of any degree in any university previously.

---

Pidong Wang

5th July 2013

# Abstract

The main aim of this thesis is to propose a text rewriting decoder, and then apply it to two applications: social media text normalization for machine translation, and source language adaptation for resource-poor machine translation.

In the first part of this thesis, we propose a text rewriting decoder based on beam search. The decoder can be used to rewrite texts from one form to another. In contrast to the beam-search decoders widely used in statistical machine translation (SMT) and automatic speech recognition (ASR), the text rewriting decoder works on the sentence level, so it can use sentence-level features, e.g., the language model score of the whole sentence.

We then apply the proposed text rewriting decoder to social media text normalization for machine translation in the second part of this thesis. Social media texts are written in an informal style, which hinders other natural language processing (NLP) applications such as machine translation. Text normalization is thus important for processing of social media text. Previous work mostly focused on normalizing words by replacing an informal word with its formal form. To further improve other downstream NLP applications, we argue that other normalization operations should also be performed, e.g., punctuation correction and missing word recovery. The proposed text rewriting decoder is adopted to effectively integrate various normalization operations. In the experiments, we have achieved statistically significant improvements over two strong baselines in both social media text normalization and translation tasks, for both Chinese and English.

In the third part of this thesis, our text rewriting decoder is applied to source language adaptation for resource-poor machine translation. As most of the world languages still remain resource-poor for machine translation and many resource-poor languages are actually related to some resource-rich languages, we propose to apply the text rewriting decoder to source language adaptation for resource-poor machine translation. Specifically, the text rewriting decoder attempts to improve machine translation from a resource-poor language $POOR$ to a target language $TGT$ by *adapting* a large bi-text for a related resource-rich language $RICH$ and the same target language $TGT$. We assumed a small $POOR$-$TGT$ bi-text which was used to learn word-level and phrase-level paraphrases and cross-lingual morphological variants between the resource-rich and the resource-poor language. Our work is of importance for resource-poor machine translation, since it can provide a useful guideline for people building machine translation systems of resource-poor languages.

# Contents

# List of Figures

# List of Tables

# Acknowledgments

This thesis is the majority part of the research work done in my five-year Ph.D. period. In the period, I have received help from many people. I will take this opportunity to thank them.

First of all, I would like to thank my supervisor, Professor Hwee Tou Ng, for his great support during the last five years of my Ph.D. study. Professor Ng always acts as a rigorous examiner of my research, and with his preciseness, he has given me great help on my research.

Sincere thanks to Professor Chew Lim Tan and Professor Khe Chai Sim for serving as my examiners, not only for this thesis, but also for my graduate research paper. I would also give thanks to Professor Kurohashi Sadao from the Kyoto University who helps me as the external examiner of this thesis. Their constructive comments have helped me significantly.

I would also thank the group members from the NUS Natural Language Processing group: Daniel Dahlmeier, Christian Hadiwinoto, Ziheng Lin, Chang Liu, Wei Lu, Preslav Nakov, Long Qiu, Xuancong Wang, Shanheng Zhao, Zhi Zhong, etc. I would give special thanks to Preslav Nakov for his great help when I just started my Ph.D. study. He did not only teach me how to do experiments, but also how to write research papers.

Last but not least, I would like to thank my family: my father Junyi Wang, my mother Hongtao Yu, and my wife Jing Niu, for their invaluable support and understanding throughout my five-year Ph.D. study.

*To my father Junyi Wang, my mother Hongtao Yu, and my wife Jing Niu.*

# Chapter 1

# Introduction

In computational linguistics, machine translation (MT) investigates how to use computers to translate text from one language to another. From the late 1980s, as the computers become more powerful, statistical machine translation (SMT) (Brown et al., 1993) has drawn more and more research attention.

SMT enables people without linguistic expertise to build MT systems, since SMT learns statistical models only from large sentence-aligned bilingual corpora of human-generated translations. We often call such kind of corpora *bi-texts*. SMT is particularly promising because we only need to collect sufficiently large bi-texts to build SMT systems without the requirement of hand-written translation rules and dictionaries. These are often necessary for other MT approaches. Furthermore, the SMT approach is largely language independent. Another advantage is that SMT systems can translate in real time with acceptable translation quality, e.g., Google Translate[1], and Bing Translator[2].

While SMT can be easily used for building translation systems, it still faces the difficulty of collecting sufficiently large, high-quality bi-texts. As a result, most of the 6,500+ world languages still remain resource-poor (Nakov and Ng, 2012).

The remainder of this chapter is organized as follows. We will first discuss social

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

media text normalization, followed by one of its applications, social media text translation. Section 1.3 introduces source language adaptation for resource-poor machine translation. Lastly, the contributions and the organization of this thesis will be presented.

## 1.1 Social Media Text Normalization

Social media texts include SMS (Short Message Service) messages, Twitter messages, Facebook updates, etc. They are different from formal texts due to their significant informal characteristics, so they always pose difficulties for applications such as machine translation (MT) (Aw et al., 2005) and named entity recognition (Liu et al., 2011), because of a lack of training data containing informal texts. Thus, the applications always suffer from a substantial performance drop when evaluated on social media texts. For example, Ritter et al. (2011) reported a drop from 90% to 76% on part-of-speech tagging, and Foster et al. (2011) found a drop of 20% in dependency parsing.

Creating training data of social media texts specifically for a text processing task is time-consuming. For example, to create parallel Chinese-English training texts for translation of social media texts, it takes three minutes on average to translate an informally written social media text of eleven words from Chinese into English. On the other hand, it takes thirty seconds to normalize the same message, a six-fold increase in speed. After training a text normalization system to normalize social media texts, we can use an existing text processing system trained on normal texts (non-social media texts) to carry out the text processing task. So we argue that normalization followed by regular text processing is a more practical approach. Thus, social media text normalization is important for social media text processing.

Most previous work on normalization of social media text focused on word substitution (Beaufort et al., 2010; Gouws et al., 2011; Han and Baldwin, 2011; Liu et al., 2012). However, we argue that some other normalization operations besides word sub-

stitution are also critical for subsequent natural language processing (NLP) applications, such as missing word recovery (e.g., zero pronouns) and punctuation correction.

## 1.2 Social Media Text Translation

Most of the MT research efforts aim at the translation of formal texts, e.g., newswire texts, which are usually well written and hardly contain any typos. Recently, a new trend of MT research is on the translation of social media texts which often contain informal words, typos, and improper punctuation symbols, e.g., "*hav u beeen there b4...*" standing for "*Have you been there before?*"

The SMS translation task in the 2011 Workshop on Statistical Machine Translation (WMT 2011) (Callison-Burch et al., 2011) paved the way for social media text translation. This task was to translate Haitian Creole SMS messages into English using dictionaries or formal bi-texts, such as Bible and Wikipedia. In this task, the best reported system (Costa-jussà and Banchs, 2011) used a source context semantic feature to improve lexical selection. This semantic feature however achieved almost no improvement according to the reported results. The CMU team (Hewavitharana et al., 2011) investigated spelling normalization and attempted to augment the available training corpus using semantic role labeling rules as well as extracting parallel sentences from comparable documents. However, all their three proposed methods failed to improve the baseline system. The LIU system (Stymne, 2011) used SMT to perform SMS normalization which normalizes informal words into their normal forms. Another system of Eidelman et al. (2011) utilized two kinds of lattices to jointly perform SMS normalization and translation.

The SMS translation task in WMT 2011 assumed the availability of some SMS training bi-texts which are however very scare in practice. Most of the world languages have little informal training bi-text.

## 1.3 Source Language Adaptation for Resource-Poor Machine Translation

Although most of the languages in the world are still resource-poor for SMT, fortunately, many of these resource-poor languages are related to some resource-rich language, and they often overlap in vocabulary and share cognates. This offers a good opportunity for improving resource-poor machine translation by using related resource-rich language bi-texts. Example pairs of such resource rich-poor languages[3] include Spanish-Catalan, Finnish-Estonian, Swedish-Norwegian, Russian-Ukrainian, Irish-Gaelic Scottish, Standard German-Swiss German, Modern Standard Arabic-Dialectical Arabic (e.g., Gulf, Egyptian), and Turkish-Azerbaijani.

Resource-poor machine translation has already attracted the attention of a lot of researchers in previous work. Some researchers used paraphrasing to improve resource-poor machine translation (Callison-Burch et al., 2006; Marton et al., 2009), while other work demonstrated the benefits of using a bi-text for a related resource-rich language to improve machine translation of a resource-poor language (Nakov and Ng, 2009; Nakov and Ng, 2012).

Nakov and Ng (2009) proposed various techniques for combining a small bi-text for a resource-poor language (Indonesian or Spanish[4]) with a much larger bi-text for a related resource-rich language (Malay or Portuguese), and the target language of all the bi-texts was English. Their work, however, did not really attempt to *adapt* the resource-rich language bi-text to get closer to the resource-poor one, except very simple transliteration for Portuguese-Spanish that ignored context entirely. Since the simple transliteration could not substitute one word for a completely different word, it did not

[3]The boundary between a language and a dialect is thin, e.g., while normally people talk about Arabic "dialects", many linguists believe that Arabic is a language family, where the "dialects" are languages. The distinction is often political, e.g., Macedonian is considered as a dialect of Bulgarian in Bulgaria but as a separate language in Macedonia.

[4]Pretending that Spanish is resource-poor.

help much for Malay-Indonesian which use unified spelling.

Another piece of work (Marujo et al., 2011) described a rule-based system for adapting Brazilian Portuguese (BP) to European Portuguese (EP), which was used to adapt BP-English bi-texts to EP-English, in order to help EP-English translation. They however reported very small improvements: when training on the adapted "EP"-English bi-text compared to using the unadapted BP-English (38.55% vs. 38.29% BLEU scores); when an EP-English bi-text was used in addition to the adapted/unadapted one (41.07% vs. 40.91% BLEU scores). Furthermore, this previous work did not take into account other language pairs, since it was a rule-based language-adaptation system which heavily relied on language-specific rules. Thus, to easily generalize to other language pairs, a statistical approach is more appropriate.

## 1.4 Contributions

The limitations of previous work are summarized as follows:

- Existing work on social media text normalization has mainly focused on word substitution, neglecting other normalization operations like missing word recovery, punctuation correction, etc.

- Previous work on social media text translation often assume social media training bi-texts which are actually very scare in practice.

- Little work has been done on improving resource-poor language machine translation by adapting bi-texts for related resource-rich languages, except some work using rule-based methods with marginal improvements.

The main objective of this thesis is to propose a general beam-search decoder for text rewriting. The decoder can then be used in social media text normalization and

source language adaptation for resource-poor machine translation. More details will be discussed in the following subsections.

### 1.4.1 A Beam-Search Decoder for Text Rewriting

To overcome the limitations in previous work, we introduce a general beam-search decoder for text rewriting in the first part of this thesis. The decoder will be subsequently applied to social media text normalization and source language adaptation to help resource-poor machine translation.

Motivated by the beam-search decoders widely used in statistical machine translation (SMT) (e.g., Moses (Koehn et al., 2007)), automatic speech recognition (ASR) (e.g., HTK (Young et al., 2002)), and grammatical error correction (Dahlmeier and Ng, 2012), we propose a novel beam-search decoder for text rewriting. Though our decoder also uses beam search, it is different from the traditional decoders used in SMT and ASR. For example, in each iteration of a phrase-based SMT decoder, one additional target phrase is appended to the target sentence which is incomplete before the final iteration. In contrast, our beam-search decoder maintains a complete sentence in each iteration of the decoder. This allows our decoder to use sentence-level features, e.g., the language model score of the whole sentence and the number of potential informal words in the whole sentence.

We apply this decoder to both social media text normalization and source language adaptation. Other NLP applications such as automatic post-processing of ASR output can also benefit from such a text rewriting decoder.

### 1.4.2 Social Media Text Normalization with Application to Machine Translation

To better translate social media texts without social media training bi-text, we propose to apply our text rewriting decoder of Section 1.4.1 to social media text normalization for machine translation. Our social media text normalization decoder can effectively integrate different normalization operations together. This work has been published in the NAACL 2013 conference (Wang and Ng, 2013).

We design a text rewriting decoder to normalize social media texts in two languages: Chinese and English. After normalization, we feed the normalized texts to a regular MT system trained on formal bi-texts. In contrast to previous work, some of our normalization operations are specifically designed for MT, e.g., missing word recovery based on conditional random fields (CRF) (Lafferty et al., 2001) and punctuation correction based on dynamic conditional random fields (DCRF) (Sutton et al., 2004).

To the best of our knowledge, our work is the first to perform missing word recovery and punctuation correction for normalization of social media text, and also the first to perform sentence-level normalization of Chinese social media text. We investigate the effects on translating social media text after addressing various characteristics of informal social media text through normalization. To show the applicability of our normalization approach for different languages, we experiment with two languages, Chinese and English. In the experiments, we achieved statistically significant improvements over two strong baselines: an improvement of 9.98%/7.35% in BLEU scores for normalization of Chinese/English social media text, and an improvement of 1.38%/1.35% in BLEU scores for translation of Chinese/English social media text. We have also created two corpora: a Chinese corpus containing 1,000 Weibo[5] messages with their normalizations and English translations, and another similar English corpus containing 2,000 SMS messages from the NUS SMS corpus (How and Kan, 2005). As far as we know, our corpora are

[5]A Chinese version of Twitter at `www.weibo.com`

the first publicly available Chinese/English corpora for normalization and translation of social media text[6].

### 1.4.3 Source Language Adaptation for Resource-Poor Machine Translation

We also apply our text rewriting decoder of Section 1.4.1 to source language adaptation for resource-poor machine translation. We compare the text rewriting decoder approach with two approaches in our previous work (Wang et al., 2012a): (1) word-level paraphrasing approach using confusion networks; (2) phrase-level paraphrasing approach using pivoted phrase tables.

More precisely, we improve machine translation of a resource-poor language by *adapting* a bi-text of a resource-rich language which is closely related to the resource-poor language. We assume a small bi-text for a resource-poor language $POOR$, and also a large bi-text for a related resource-rich language $RICH$. These two languages are closely related and share vocabulary and cognates, and the two bi-texts have the same target language $TGT$. From the two bi-texts, a statistical approach learns word-level and phrase-level paraphrases and cross-lingual morphological variants between the two languages. These paraphrases and morphological variants are then used to adapt the source side of the resource-rich bi-text from language $RICH$ to $POOR$. After the adaptation, each of the adapted "$POOR$" sentences is paired with its $TGT$ counterpart in the $RICH$-$TGT$ bi-text. As a result, we obtain a synthetic "$POOR$"-$TGT$ bi-text which is then used to improve machine translation from the resource-poor language $POOR$ to $TGT$.

With a resource-rich Malay-English (*ML2EN*) and a resource-poor Indonesian-English bi-text (*IN2EN*), we have achieved very significant improvements over several baselines (7.26% BLEU scores over an unadapted version of *ML2EN*, 3.09% BLEU

[6]Available at `www.comp.nus.edu.sg/~nlp/corpora.html`

scores over *IN2EN*, and 1.93-3.25% BLEU scores over three bi-text combinations of *ML2EN* and *IN2EN*), thus proving the potential of the idea of source-language adaptation for resource-poor machine translation. We have further demonstrated the applicability of the general approach to other languages and domains.

This part of our work provides insights into the importance of utilizing the close relationship between languages to help resource-poor machine translation. Also, it provides the foundation for source language adaptation of bi-texts to improve resource-poor machine translation.

## 1.5 Organization of This Thesis

The remainder of this thesis is organized as follows. The next chapter presents a detailed literature review of related work. Then in Chapter 3, we will describe our beam-search decoder for text rewriting which will be applied to social media text normalization in Chapter 4 and source language adaptation in Chapter 5. Finally, Chapter 6 will conclude the thesis and propose future work.

# Chapter 2

# Related Work

In this chapter, we will briefly review previous work on beam-search decoders, and then discuss related work on social media text normalization and translation. Finally we will present related work on source language adaptation for resource-poor machine translation.

## 2.1 Beam-Search Decoders

Beam search (Russell and Norvig, 2010) is a heuristic search algorithm which tries to search for the best path in a graph. In each iteration, beam search first produces all new hypotheses obtained from the hypotheses in the frontier of the previous iteration, and then sort the new hypotheses in decreasing order of heuristic scores. It only retains a predefined number of best hypotheses at the end of each iteration. The number is called the beam width, which is set to limit the memory usage and runtime of the beam search. The theoretical best hypothesis may not be found by the beam search algorithm, because it may be pruned during the search process.

Beam-search decoders are widely used in many applications, e.g., statistical machine translation (SMT) (e.g., the phrase-based SMT decoder in Moses (Koehn et al.,

2007)) and automatic speech recognition (ASR) (e.g., the hidden Markov model toolkit HTK (Young et al., 2002)). We propose a novel beam-search decoder for text rewriting which will then be applied to social media text normalization and source language adaptation.

The phrase-based SMT decoder (Koehn, 2013) in Moses also employs a beam-search algorithm. Given an input sentence in the source language, the output sentence in the target language is generated left to right in the form of a hypothesis. For example, given the input sentence $s_1 s_2 s_3$, with the translation options: $\{(s_1, t_2), (s_1 s_2, t_2 t_5), (s_2 s_3, t_6), (s_3, t_4)\}$, the search tree is shown in Figure 2.1. Starting from the initial hypothesis, we expand each hypothesis by adding one more target phrase to the output sentence. Before the final iteration, the output sentence in each hypothesis is incomplete. Even though the Moses decoder also uses the language model score as a feature, the score is estimated before the final iteration due to the incompleteness of the output sentence.

HTK[1] (Young et al., 2002) is a toolkit for building and manipulating hidden Markov models (HMMs). It is widely used to build ASR systems. The HVITE of HTK performs ASR through a token passing paradigm to find the best path in the network of HMM states. A token is a partial path in the network from time 0 to time $t$. The number of tokens that each node keeps has a significant impact on time and memory usage. Of course, the number should be limited, since the network is usually very huge. As a result, only promising tokens which have a good chance to be part of the best path are retained in each node, i.e., pruning is carried out. At each time step, a record of the best token overall is kept, and all tokens whose log probabilities fall more than a beam-width below the best token are discarded. By using pruning, we can perform ASR in an acceptable amount of time. The target sentence is generated word by word, so HVITE cannot utilize sentence-level features during decoding.

Since the decoders used in SMT and ASR mostly work on the phrase or word

[1] `http://htk.eng.cam.ac.uk/`

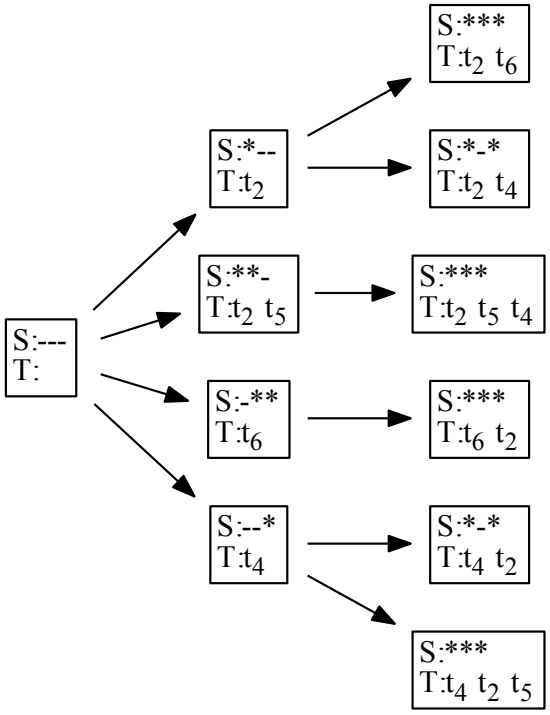


Figure 2.1: **An example search tree of the phrase-based translation decoder in Moses.** A source word (in S:) which has already been translated is marked as an asterisk (*), otherwise it is marked as a dash (-). The generated target sentence is shown in T:. Unknown words are not translated.

level, they cannot utilize sentence-level features during the beam-search process. In contrast, the text rewriting decoder proposed in this thesis works on the sentence level, i.e., the sentence in each hypothesis is a complete sentence. As such, the proposed decoder can use real sentence-level features, e.g., the language model score of the whole sentence.

For example, given the same input sentence and the same translation options as the example of the phrase-based SMT decoder, the search tree of the proposed text rewriting decoder is shown in Figure 2.2. Starting from the initial hypothesis, we expand each hypothesis by replacing a source phrase with a target phrase using one phrase pair from the translation options.

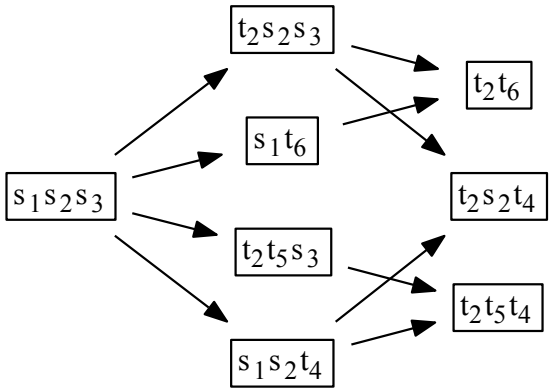


Figure 2.2: **An example search tree of the proposed text rewriting decoder.** Each hypothesis maintains a complete sentence.

## 2.2 Social Media Text Normalization

The first application of our beam-search text rewriting decoder is social media text normalization for machine translation.

Zhu et al. (2007) performed text normalization of informally written email messages using CRF (Lafferty et al., 2001). Due to its importance, normalization of social media text has been extensively studied recently. Aw et al. (2005) proposed a noisy channel model consisting of different operations: substitution of non-standard acronyms, deletion of flavor words, and insertion of auxiliary verbs and subject pronouns. Choudhury et al. (2007) used hidden Markov model to perform word-level normalization. Kobus et al. (2008) combined MT and automatic speech recognition (ASR) to better normalize French SMS message. Cook and Stevenson (2009) used an unsupervised noisy channel model considering different word formation processes. Han and Baldwin (2011) normalized informal words using morphophonemic similarity. Pennell and Liu (2011) only dealt with SMS abbreviations. Xue et al. (2011) normalized social media texts incorporating orthographic, phonetic, contextual, and acronym factors. Liu et al. (2012) designed a system combining different human perspectives to perform word-level normalization. Oliva et al. (2012) normalized Spanish SMS messages using a normalization and a phonetic dictionary. For normalization of Chinese social media text, Xia et al. (2005) investigated informal phrase detection, and Li and Yarowsky (2008) mined informal-formal phrase pairs from Web corpora. Wang and Kan (2013) performed Chinese word segmentation and informal word detection jointly using a dynamic conditional random fields (DCRF) model (Sutton et al., 2004), and Wang et al. (2013) normalized Chinese informal words with a two-stage selection-classification model.

All the above work focused on normalizing words. In contrast, our work also performs other normalization operations such as missing word recovery and punctuation correction, to further improve machine translation. Previously, Aw et al. (2006) adopted phrase-based MT to perform SMS normalization, and required a relatively large number

of manually normalized SMS messages. In contrast, our approach performs beam search at the sentence level, and does not require large training data.

In speech to speech translation (Paul, 2009; Nakov et al., 2009), the input texts contain wrongly transcribed words due to errors in automatic speech recognition, whereas social media texts contain abbreviations, new words, etc. Although the input texts in both cases deviate from normal texts, the exact deviations are different.

## 2.3 Social Media Text Translation

Statistical machine translation (SMT) (Brown et al., 1993; Lopez, 2008) treats machine translation (MT) as a machine learning problem. In SMT, we first need to collect large amounts of parallel corpus, and then we use a machine learning algorithm to learn statistical translation models from the parallel corpus. The learned model then can translate new sentences which can be unseen in the training parallel corpus. In only about two decades, SMT has been more and more popular in both the academic MT research field and the commercial MT market. That is why more and more MT researchers work on SMT. The advantage of SMT is that it needs no manual development of translation rules or dictionaries, but is trained on large parallel corpora. Its drawback is that it requires large parallel corpora which may not be available. However, assembling parallel corpora may be easier than developing translation rules, because every person who can use two languages is able to construct parallel corpora by manual translation, but only linguistic experts can develop grammars and linguistic rules for translation.

In this thesis, we use phrase-based SMT (Koehn, 2010) which is an approach to SMT. More precisely, we use the phrase-based SMT decoder in Moses (Koehn et al., 2007). Given a parallel training corpus, separate directed word alignments are first built using IBM model 4 (Brown et al., 1993) for both directions of the corpus. We then combine the word alignments using the intersect+grow heuristic (Och and Ney,

2003). Based on the combined word alignments, a phrase table containing phrase-level translation pairs and corresponding features is extracted using the alignment template approach (Och and Ney, 2004). A log-linear model is adopted to combine the features in the phrase table, a language model score, word penalty, and distortion costs. The weights of the log-linear model are tuned to optimize the BLEU score (Papineni et al., 2002) on the development set using minimum error rate training (MERT) (Och, 2003). The phrase-based SMT decoder of Moses is used to perform translation with the log-linear model.

We evaluate the success of social media text normalization in the context of machine translation, so research on machine translation of social media text is relevant to our work.

However, there is not much comparative evaluation of social media text translation other than the Haitian Creole to English SMS translation task in the 2011 Workshop on Statistical Machine Translation (WMT 2011) (Callison-Burch et al., 2011). The task assumes the availability of SMS training bi-texts and other general domain bi-texts including medical domain, newswire domain, glossary, wikipedia data, Bible, etc. The best reported system in WMT 2011 (Costa-jussà and Banchs, 2011) used a source context semantic feature to improve lexical selection for the raw SMS translation track. The CMU team (Hewavitharana et al., 2011) investigated word-level spelling normalization and attempted to augment the available training corpus using semantic role labeling rules as well as extracting parallel sentences from comparable documents. However, all their three proposed methods failed to improve the baseline system. The LIU system (Stymne, 2011) treated SMS normalization as an SMT task. Inspired by the spelling correction work of Brill and Moore (2000), they proposed an approach of finding spelling options for unknown words, and the options were encoded in a confusion network which was decoded by the SMT system. Eidelman et al. (2011) utilized two kinds of lattices to help SMS translation.

However, the setup of the WMT 2011 task is different from ours, in that the task provided parallel training data of SMS texts and their translations. As such, text normalization is not necessary in that task.

## 2.4 Source Language Adaptation for Resource-Poor Machine Translation

The second application of our beam-search text rewriting decoder is source language adaptation for resource-poor machine translation. More precisely, we use our text rewriting decoder to adapt bi-texts for a resource-rich language to another resource-poor language which is closely related to the resource-rich language, and the adapted bi-text is then used to improve machine translation of the resource-poor language.

One relevant line of research is on machine translation between closely related languages, which is arguably simpler than general SMT, and thus can be handled using word-for-word translation, manual language-specific rules that take care of the necessary morphological and syntactic transformations, or character-level translation/transliteration. This has been tried for a number of language pairs including Czech-Slovak (Hajič et al., 2000), Turkish-Crimean Tatar (Altintas and Cicekli, 2002), Irish-Scottish Gaelic (Scannell, 2006), and Bulgarian-Macedonian (Nakov and Tiedemann, 2012). In contrast, we have a different objective – we do not carry out full translation but rather adaptation since our ultimate goal is to translate into a third language $X$.

A special case of this same line of research is the translation between dialects of the same language, e.g., between Cantonese and Mandarin (Zhang, 1998), or between a dialect of a language and a standard version of that language, e.g., between some Arabic dialect (e.g., Egyptian) and Modern Standard Arabic (Bakr et al., 2008; Sawaf, 2010; Salloum and Habash, 2011). Here again, manual rules and/or language-specific tools are typically used. In the case of Arabic dialects, a further complication arises by the

informal status of the dialects, which are not standardized and not used in formal contexts but rather only in informal online communities[2] such as social networks, chats, Twitter and SMS messages. This causes further mismatch in domain and genre.

Thus, translating from Arabic dialects to Modern Standard Arabic requires, among other things, normalizing informal text to a formal form. In fact, this is a more general problem, which arises with informal sources like SMS messages and Tweets for just any language (Aw et al., 2006; Han and Baldwin, 2011). We have addressed this problem in Section 2.2.

A second relevant line of research is on language adaptation and normalization, when done specifically for improving SMT into another language. For example, Marujo et al. (2011) described a rule-based system for adapting Brazilian Portuguese (BP) to European Portuguese (EP), which they used to adapt BP-English bi-texts to EP-English. They report small improvements in BLEU for EP-English translation when training on the adapted "EP"-English bi-text compared to using the unadapted BP-English (38.55% vs. 38.29%), or when an EP-English bi-text is used in addition to the adapted/unadapted one (41.07% vs. 40.91% BLEU). Unlike their work, which heavily relied on language-specific rules, our approach is statistical, and largely language-independent. Moreover, our improvements are much more sizable.

A third relevant line of research is on reusing bi-texts between related languages without or with very little adaptation, which works well for very closely related languages. For example, the previous work of (Nakov and Ng, 2009; Nakov and Ng, 2012) experimented with various techniques for combining a small bi-text for a resource-poor language (Indonesian or Spanish[3]) with a much larger bi-text for a related resource-rich language (Malay or Portuguese); the target language of all bi-texts was English. However, the previous work did not attempt language adaptation, except for very simple transliteration for Portuguese-Spanish that ignored context entirely; since it could not

[2]The Egyptian Wikipedia is one notable exception.

[3]Pretending that Spanish is resource-poor.

substitute one word for a completely different word, it did not help much for Malay-Indonesian, which use unified spelling. Still, once we have language-adapted the large bi-text, it makes sense to try to combine it further with the small bi-text. We plan to directly compare and combine these two approaches in this thesis.

Another alternative, which we do not explore in this thesis, is to use cascaded translation using a pivot language (Utiyama and Isahara, 2007; Cohn and Lapata, 2007; Wu and Wang, 2009). Unfortunately, using the resource-rich language as a pivot (poor$\rightarrow$ rich$\rightarrow X$) would require an additional parallel poor-rich bi-text, which we do not have. Pivoting over the target $X$ (rich$\rightarrow X\rightarrow$poor) for the purpose of language adaptation, on the other hand, would miss the opportunity to exploit the relationship between the resource-poor and the resource-rich language; this would also be circular since the first step would ask an SMT system to translate its own training data (we only have one rich-$X$ bi-text).

## 2.5 Summary

This chapter reviews the related work of this thesis including beam-search decoders, social media text normalization and translation, and source language adaptation for resource-poor machine translation.

# Chapter 3

# A Beam-Search Decoder for Text Rewriting

In this chapter, we will present the general framework of our beam-search decoder for text rewriting. In the following chapters, the decoder will be applied to two applications: social media text normalization, and source-language adaptation for resource-poor machine translation.

The aim of the decoder will be first described, followed by its core beam-search algorithm. Then the details of the decoder will be discussed including its hypothesis producers, feature functions, and weight tuning. The comparison between the proposed decoder and traditional lattice decoding will be subsequently investigated, followed by the implementation details of the decoder. Lastly, we will conclude the chapter.

## 3.1 Goal

While designing our beam-search decoder for text rewriting, we aim for a general framework which can be applied to both social media text normalization and source language adaptation of bi-text, since the two applications are quite different in the sense that the

former normalizes informal text into formal text in the same language, while the latter adapts texts from one language to another related language. Furthermore, social media text normalization needs to perform different kinds of text rewriting operations, e.g., replacing informal words with their formal forms, inserting missing words like zero-pronouns, correcting non-standard punctuation marks, etc. Thus, our decoder should have the ability to effectively integrate different operations together to achieve better performance.

## 3.2 Beam-Search Algorithm for Text Rewriting

Given an input sentence, our text rewriting decoder searches for its best rewritten form (i.e., the best hypothesis), considering all the methods for rewriting the input sentence. To find the best hypothesis, our decoder iteratively performs two sub-tasks:

- producing new sentence-level hypotheses from the hypotheses in the current stack, which is carried out by the **hypothesis producers**;
- evaluating all the new hypotheses produced by the **hypothesis producers** to retain good ones in the next stack, which is carried out by the **feature functions**.

The beam-search algorithm is shown in Algorithm 1, in which we use the same pruning method of the phrase-based SMT decoder in Moses (Koehn et al., 2007). The pruning method is called lazy pruning: assuming the stack size is $K$, we only perform pruning to retain $K$-best hypotheses. In the algorithm, the stack index $i$ represents the total number of modifications made by all the hypothesis producers. The maximum number of iterations equals the number of tokens (including both words and punctuation marks) in the input sentence, i.e., we suppose each token needs at most one modification on average. Eventually, we choose the best hypothesis in all the hypothesis stacks as the best rewritten form for the input sentence. One example search tree of the algorithm is shown in Section 2.1.

**Algorithm 1** Beam-Search Text Rewriting

INPUT: an input **INPUT** whose length is **N**
RETURN: the best rewritten form for **INPUT**

```
1: initialize hypothesisStacks[0...N] and hypothesisProducers;
2: add the initial hypothesis INPUT to stack hypothesisStacks[0];
3: for i ← 0 to N-1 do
4:    for each hypo in hypothesisStacks[i] do
5:       for each producer in hypothesisProducers do
6:          for each newHypo produced by producer from hypo do
7:             add newHypo to hypothesisStacks[i+1];
8:             prune hypothesisStacks[i+1];
9: return the best hypothesis in hypothesisStacks[0...N];
```

## 3.3 Hypothesis Producers

Given a hypothesis, the duty of a specific hypothesis producer is to produce new hypotheses from the given one using the knowledge of the hypothesis producer. A new hypothesis has only one more modification than the given hypothesis.

For example, for social media text normalization, one simple hypothesis producer can utilize a pre-defined normalization dictionary which contains informal-formal phrase pairs. Given the hypothesis "*im waiting 4 u*", this hypothesis producer may examine each word of the hypothesis, and then produce the following new hypotheses:

- "***i 'm*** *waiting 4 u*",
- "*im waiting* ***for*** *u*", and
- "*im waiting 4* ***you***",

if the normalization dictionary contains phrase pairs: "*(im, i 'm)*", "*(4, for)*", and "*(u, you)*".

## 3.4 Feature Functions

The feature functions can be categorized into two kinds:

1. The first kind is called count feature functions, i.e., each hypothesis producer has a count feature function which is the count of modifications made by the hypothesis producer. The count feature functions can be used by the decoder to distinguish good hypothesis producers from bad ones. More precisely, if the decoder finds a specific hypothesis producer to be more useful than others, it can give the count feature of the hypothesis producer a higher weight to let the hypothesis producer perform more modifications, since hypotheses with more modifications made by the hypothesis producer have higher scores, they are more likely to survive pruning and be chosen as the best hypothesis.

2. The second kind is some general feature functions, e.g., language model scores, informal word penalty (i.e., the number of informal words), etc. Depending on the application, any feature function can be used inside the decoder.

All feature functions are combined using a linear model to obtain the score for a hypothesis $h$:

$$score(h) = \sum_i \lambda_i f_i(h), \tag{3.1}$$

where $f_i$ is the $i$-th feature function with weight $\lambda_i$. $score(h)$ is used by the decoder to discriminate good hypotheses from bad ones. More specifically, based on $score(h)$, the beam-search decoder can prune bad hypotheses and also select the best hypothesis with the highest $score(h)$ from all the stacks.

## 3.5 Weight Tuning

The weights of the feature functions are tuned using the pairwise ranking optimization (PRO) algorithm (Hopkins and May, 2011) on the development set.

PRO tunes the weights based on a pair-wise ranking approach. For each tuning instance in the development set, PRO first starts with sampling hypothesis pairs from the

$n$-best list of hypotheses output by the decoder for the tuning instance. The evaluation metric ranks the two hypotheses in every pair. PRO aims to find a weight vector to rank the hypothesis pair in the same order as the evaluation metric scores. More specifically, we can rewrite Equation 3.1 in the following form:

$$score_W(h) = W \cdot F(h), \tag{3.2}$$

where $W$ is a weight vector, i.e., a vector of $\lambda_i$, and $F$ is a feature function vector of $f_i$. Given one tuning instance and any of its hypothesis pair $(h_1, h_2)$, if the evaluation metric score of $h_1$ is higher than that of $h_2$, we wish that the hypothesis scores rank the hypothesis pair in the same order as the evaluation metric scores:

$$\begin{aligned} score_W(h_1) > score_W(h_2) &\Leftrightarrow W \cdot F(h_1) > W \cdot F(h_2) \\ &\Leftrightarrow W \cdot F(h_1) - W \cdot F(h_2) > 0 \\ &\Leftrightarrow W \cdot (F(h_1) - F(h_2)) > 0 \end{aligned} \tag{3.3}$$

Weight tuning can thus be simplified to a binary classification problem.

In our work, PRO is used to optimize a sentence-level BLEU approximation (BLEU+1) (Liang et al., 2006) on the development set instead of document-level BLEU (Papineni et al., 2002), because document-level BLEU often can be zero for an individual sentence.

## 3.6 The Text Rewriting Decoder Versus Lattice Decoding

Another alternative way for text rewriting is through lattice decoding which was introduced in automatic speech recognition (Jelinek, 1997). In lattice decoding, each word/phrase of an input sentence was augmented with its rewritten forms in a lattice, and then the lattice is decoded using a language model to find a rewritten form of the

input sentence. In this section, we will compare our proposed text rewriting decoder to lattice decoding.

First of all, our text rewriting decoder is more flexible in the sense that it can utilize more feature functions than lattice decoding in which only two feature functions are usually used: (1) the scores on the edges of the input lattice; and (2) language model score. For example, our social media text normalization decoder to be presented in Chapter 4 uses informal word penalty as a feature function, i.e., the count of informal words. Moreover, our decoder works at the sentence level, while lattice decoding works at the word or phrase level. As a result, our decoder can use sentence-level features during the search process, e.g., the language model score of the whole sentence.

Another advantage of our decoder is that lattice decoding is based on a static search graph while our text rewriting decoder uses hypothesis producers to expand the search paths dynamically. For example, our decoder can make multiple changes to one word/phrase, so it can normalize the informal word "*thx.whr*" in "*thx.whr r u*" which needs three changes for proper normalization: from "*thx.whr*" to "*thx . whr*", from "*thx . whr*" to "*thanks . whr*", and then from "*thanks . whr*" to "*thanks . where*". This is very difficult for lattice decoding, since the lattice is generated in advance before decoding the lattice using a language model, and it is not clear how to set the scores on the edges of the lattice. In contrast, our text rewriting decoder can handle these kinds of multiple changes very well, if we have appropriately designed the hypothesis producers.

## 3.7 Implementation Details

### 3.7.1 Programming Details

The proposed decoder is implemented using the Java programming language. Although Java applications are less efficient than C++ ones, Java has fewer dependencies on the operating systems. As a result, the decoder can work on different platforms, e.g., Mi-

crosoft Windows, Linux, Unix, etc. Moreover, the decoder also uses multi-threading to improve the decoding speed, which is also strongly supported by Java.

For the language model feature function, we use the Berkeley language model (Pauls and Klein, 2011), since it is also implemented in Java.

The conditional random fields (CRF) (Lafferty et al., 2001) and dynamic conditional random fields (DCRF) models (Sutton et al., 2004) are also used in our decoder to build hypothesis producers. We use the CRF and DCRF models in GRMM (GRaphical Models in Mallet) (Sutton, 2006) which is implemented using Java. In GRMM, we use the tree-based reparameterization (TRP) schedule (Wainwright et al., 2001) for approximate inference.

### 3.7.2 Decoder Parameters

The stack size of the decoder is set to 20. The maximum number of iterations equals the number of tokens in the input sentence, i.e., we assume that each token needs at most one modification on average. In previous experiments, we found that larger stack sizes had little effect on the results.

### 3.7.3 Weight Tuning Settings

The iterations of PRO weight tuning can be summarized as follows:

1. Run the decoder with the weights tuned in the previous iteration to generate an $n$-best ($n = 100$) list for each sentence of the development set;

2. Sample hypothesis pairs from the $n$-best lists;

3. Run a binary classifier to get the tuned weight for each feature function;

4. If the maximum number of iterations is reached, select the tuned weights with the best performance on the development set; otherwise, go to Step 1.

**Hypothesis sampling.** For weight tuning, the PRO parameters proposed by Hopkins and May (2011) are used. More precisely, for every input sentence in the development set, 5,000 hypothesis pairs are sampled from the 100-best list for the input sentence, and we only keep the top 50 sample pairs with the highest difference in BLEU+1 scores. For each sampled hypothesis pair, two training instances are created: one example with the original hypothesis pair, and the other example with the swapped hypothesis pair. All the training instances are used as a training file for a binary classifier which returns the tuned weights for the feature functions.

**Binary classification.** As shown in Section 3.5, by using the PRO tuning algorithm, the task of weight tuning can be simplified to a binary classification problem. We use the MegaM (Daumé III, 2004) classifier to solve the binary classification problem. MegaM solves binary classification problems using conjugate gradient ascent (Hestenes and Stiefel, 1952).

The initial weights of the feature functions are set to 1.0, and the maximum number of PRO loop iterations is set to 10. The tuned feature weights output by MegaM are normalized to a unit interval. In previous experiments, we found that larger maximum number of PRO loop iterations had little effect on the results.

## 3.8 Summary

This chapter presents the general framework of our beam-search text rewriting decoder, including its goal, beam-search algorithm, hypothesis producers, feature functions, and tuning algorithm. The proposed decoder is then compared to traditional lattice decoding, followed by its implementation details. In the following chapters, we will apply this decoder to two different tasks: (1) social media text normalization; and (2) source language adaptation for resource-poor machine translation.

So far the main framework of our text rewriting decoder has been presented, and

the remaining work to apply the decoder in each application is to design its **hypothesis producers** and **feature functions** according to the characteristics of the application.

# Chapter 4

# Normalization of Social Media Text with Application to Machine Translation

In this chapter, we will apply our text rewriting decoder presented in Chapter 3 to social media text normalization to help social media text translation. The work of this chapter has been published in the NAACL 2013 conference (Wang and Ng, 2013).

We will first analyze the challenges in social media text normalization, with a view towards application to machine translation. We then present a text normalization decoder based on our text rewriting decoder. Next, we introduce various text normalization operations including punctuation correction and missing word recovery, which will be used as hypothesis producers in the text normalization decoder. Subsequently, we will give the details of our text normalization decoders for Chinese and English, followed by experiments on social media text normalization and translation. Finally, we will summarize this chapter.

## 4.1 Challenges in Normalization of Social Media Text

To better understand the informal characteristics of social media texts, we first analyzed a small sample of such texts in Chinese and English.

| Category | Frequency | Example |
|---|---|---|
| Punctuation | 81 | 你好*[hi]* ～(你好 。*[hi .]*); |
| Pronunciation | 47 | 表*[watch]*(不要*[don't]*); 酱紫(这样子*[this]*); |
| New word | 43 | 萌*[bud]*(可爱*[cute]*); |
| Interjection | 27 | 好的*[ok]* 哦*[oh]*(好的*[ok]*); |
| Pronoun | 23 | 想要*[want]*(我*[i]* 想要*[want]*); |
| Segmentation | 14 | 表酱紫(不要*[don't]* 这样子*[this]*); |

Table 4.1: Occurrence frequency of various informal characteristics in 200 Chinese social media messages from Weibo. The manually normalized form is shown in round brackets, and the English gloss is shown in square brackets.

We crawled 200 Chinese messages from Weibo, a Chinese version of Twitter. The informal characteristics of these messages are shown in Table 4.1. The manually normalized form is shown in round brackets, and the English gloss is shown in square brackets. Omitted, extraneous, and misused punctuation symbols occur frequently, which presents a problem for the subsequent machine translation (MT) step, as MT systems are often trained on formal text with correct punctuation. On average, each Chinese message contains only less than one informal word, and many informal words are either new words (e.g., "酱紫(这样子*[this]*)") or existing words with new meaning (e.g., "表*[watch]*(不要*[don't]*)"). The messages also contain redundant interjections, e.g., the interjection "哦*[oh]*" in the message "好的*[ok]* 哦*[oh]*", which often hinder machine translation systems. Pronouns are often omitted in Chinese messages, especially the pronoun "我*[I]*". For example, "喜欢*[like]*" is often used in Chinese social media text instead of "我*[I]* 喜欢*[like]*", which also causes problems for machine translation systems, since current machine translation systems always translate phrase by phrase and cannot recover missing words. Chinese informal words can be wrongly segmented due to a lack of word seg-

mentation training data containing informal words, and these wrongly segmented words are often treated as unknown words by machine translation systems.

| Category | Frequency | Example |
|---|---|---|
| Pronunciation | 288 | *4*(*for*); *oredi*(*already*); |
| Abbreviation | 98 | *slp*(*sleep*); *whr*(*where*); |
| Prefix | 74 | *lect*(*lecture*); *doin*(*doing*); |
| Punctuation | 69 | *where r u*(*where r u ?*); |
| Interjection | 68 | *ok lor .*(*ok .*); |
| Quotation | 24 | *im sure*(*i 'm sure*); *dont go*(*don 't go*); |
| Be | 24 | *i coming*; *you free?*; |
| Tokenization | 19 | *ok.why ?*(*ok . why ?*); |
| Time | 2 | *end at 730*(*end at 7:30*); *1130 am*(*11:30 am*); |

Table 4.2: Occurrence frequency of various informal characteristics in 200 English social media messages from the NUS SMS corpus. The manually normalized form is shown in round brackets.

Similarly, 200 English SMS messages were randomly selected from the NUS SMS corpus (How and Kan, 2005). The informal characteristics of these messages are shown in Table 4.2. We found that our English messages contain more informal words than Chinese messages. We usually have no way to shorten Chinese words, while English words can be shortened in three ways: (1) using a shorter word form with similar pronunciation, e.g., "*oredi*(*already*)"; (2) abbreviating a formal word by removing vowel letters, e.g., "*slp*(*sleep*)"; and (3) using only a prefix of a formal word, e.g., " *doin*(*doing*)". These shortened words are often treated as unknown words by machine translation systems, so they cannot be translated. Other informal characteristics in the English messages include: (1) informal punctuation conventions including omitted and misused punctuation; (2) redundant interjections, e.g., the interjection word "*lor*" in "*ok lor .*", which often cause problems for machine translation systems; (3) quotation-related problems due to the simple reason that it is hard to type quotation marks using mobile phones, e.g., omitted quotation marks; (4) "*be*" omission, e.g., "*he going*"; (5) tokeniza-

tion problems which always pose difficulties for machine translation systems, since the incorrectly tokenized words cannot be translated; and (6) informally written time expressions which will be translated wrongly by machine translation systems.

## 4.2 Methods

As can be seen in Section 4.1, social media texts of different languages exhibit different informal characteristics. For example, English messages have more informal words than Chinese messages, while punctuation problems are more prevalent for Chinese messages. Also, fixing different types of informal characteristics often depends on each other. For example, to be able to correct punctuation, it helps that the surrounding words are already correctly normalized. On the other hand, with punctuation already corrected, it will be easier to normalize the surrounding words.

In this section, we first present a novel beam-search decoder for normalization of social media text. The decoder can effectively integrate different normalization operations, including statistical and rule-based normalization. Then we will introduce our punctuation correction method based on a dynamic conditional random fields (DCRF) model (Sutton et al., 2004), and missing word recovery method based on a conditional random fields (CRF) model (Lafferty et al., 2001). The two methods will be used as hypothesis producers in the text normalization decoder. Finally, other hypothesis producers for Chinese and English text normalization are presented.

### 4.2.1 A Decoder for Text Normalization

When designing our text normalization system, we aim for a general framework that can be applied to text normalization across different languages with minimal effort, based on the text rewriting decoder proposed in Section 3.2. This is a challenging task, since social media texts in different languages exhibit different informal characteristics, as illustrated

in Section 4.1.

Given an input message, the normalization decoder searches for its best normalization, i.e., the best hypothesis, by iteratively performing two subtasks:

1. Producing new sentence-level hypotheses from hypotheses in the current stack, carried out by *hypothesis producers*;

2. Evaluating the new hypotheses to retain good ones, carried out by *feature functions*.

Each hypothesis is the result of applying successive normalization operations on the initial input message, where each normalization operation is carried out by one hypothesis producer that deals with one aspect of the informal characteristics of social media text. The hypotheses are grouped into stacks, where stack $i$ stores all hypotheses obtained by applying $i$ hypothesis producers on the input message.

**Algorithm 2** Beam-Search Text Normalization

INPUT: a raw message **M** whose length is **N**
RETURN: the best normalization for **M**

```
1:  initialize hypothesisStacks[0...N] and hypothesisProducers;
2:  add the initial hypothesis M to stack hypothesisStacks[0];
3:  for i ← 0 to N-1 do
4:     for each hypo in hypothesisStacks[i] do
5:        for each producer in hypothesisProducers do
6:           for each newHypo produced by producer from hypo do
7:              detect informal words in newHypo;
8:              add newHypo to hypothesisStacks[i+1];
9:              prune hypothesisStacks[i+1];
10: return the best hypothesis in hypothesisStacks[0...N];
```

Considering the informal characteristics of social media texts discussed in Section 4.1, we have added one more informal word detection step to the decoder algorithm of Section 3.2. The new algorithm is shown in Algorithm 2. A number of the hypothesis producers detect and deal with informal words $w$ present in a hypothesis by relying

on bigram counts of $w$ in a large corpus of formal texts. Specifically, a word $w$ in a hypothesis $\ldots w_{-1}ww_1\ldots$ is considered an informal word if both bigrams $w_{-1}w$ and $ww_1$ occur infrequently ($\leq 5$) in the formal corpus. We will give the details of the hypothesis producers for Chinese and English social media texts in Section 4.2.4 and 4.2.5 respectively.

Given a hypothesis message $h$, the feature functions include a language model score (the normalized sentence probability of $h$), an informal word count penalty (the number of informal words detected in $h$), and count feature functions. Each count feature function gives the count of the modifications made by a hypothesis producer. The feature functions are used by the decoder to distinguish good hypotheses from bad ones. As shown in Section 3.4, the feature functions are combined in a linear model, and the weights of the feature functions are tuned using a pairwise ranking optimization algorithm (Hopkins and May, 2011) on the development set.

Figure 4.1 shows an example search tree of our Chinese text normalization decoder (Section 4.2.4) when normalizing the text "想*[want]* 买*[buy]* 。 神马*[magical horse]* 时候*[time]* 买*[buy]*". Figure 4.2 shows an example search tree of our English text normalization decoder (Section 4.2.5) when normalizing the text "*whr u*".

### 4.2.2 Punctuation Correction

In normalization of social media text, punctuation correction is also important besides word normalization, as the subsequent NLP applications are typically trained on formal texts with correct punctuation. We define punctuation correction as correcting punctuation in sentences which may have no or unreliable punctuation. The task performs three punctuation operations: insertion, deletion, and substitution. In our beam-search decoders for Chinese and English text normalization, the punctuation correction method will be used as a hypothesis producer which corrects punctuation in the current hypothesis.

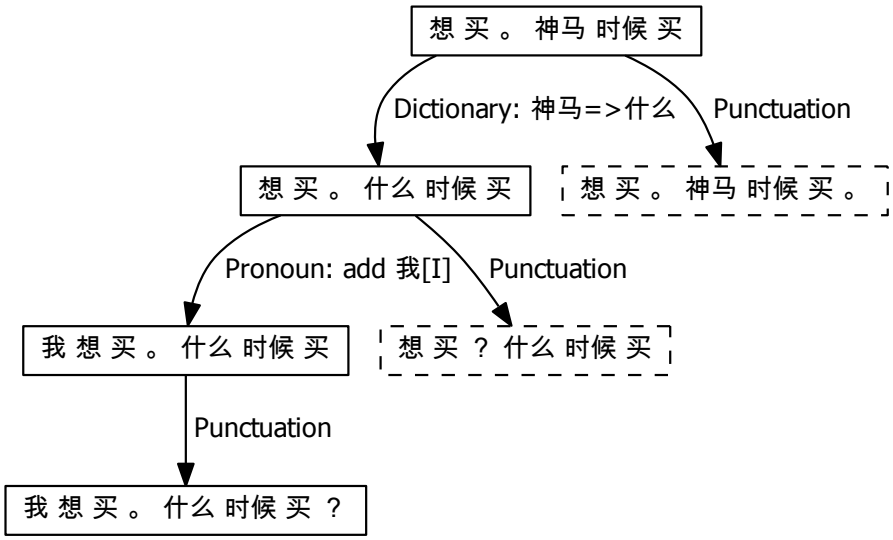


Figure 4.1: **An example search tree of our Chinese text normalization decoder.** The solid (dashed) boxes represent good (bad) hypotheses. The hypothesis producers are indicated on the edges.

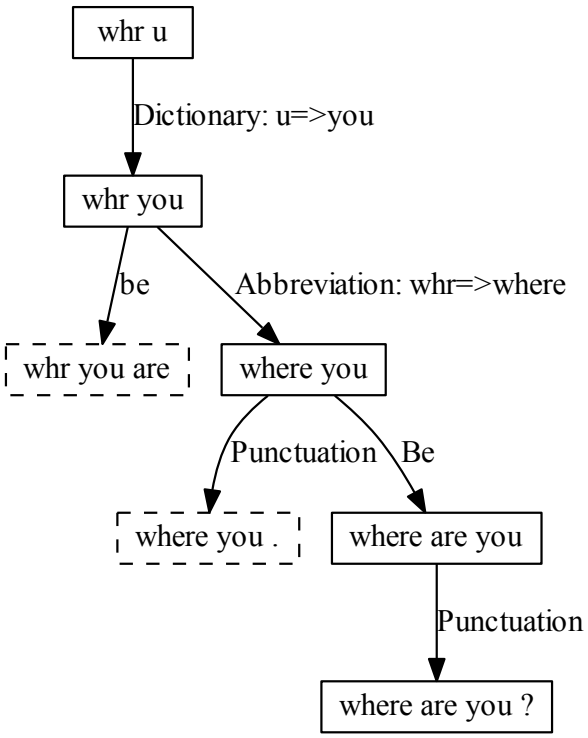


Figure 4.2: **An example search tree of our English text normalization decoder.** The solid (dashed) boxes represent good (bad) hypotheses. The hypothesis producers are indicated on the edges.

#### 4.2.2.1 Punctuation Correction Model

To our knowledge, no previous work has been done on punctuation correction for normalization of social media text. In automatic speech recognition (ASR), punctuation prediction only inserts punctuation symbols into ASR output that has no punctuation (Kim and Woodland, 2001; Huang and Zweig, 2002; Wang et al., 2012b), but without punctuation deletion or substitution. Lu and Ng (2010) argued that punctuation prediction should be jointly performed with sentence boundary detection, so they modeled punctuation prediction using a two-layer DCRF model (Sutton et al., 2004).

Given an observation sequence, the linear-chain CRF model can be used to label one layer of tags, i.e., each observation is assigned one tag, while the DCRF model is able to simultaneously label multiple layers of tags for the same observation sequence, i.e., each observation can have multiple tags. Formally, the DCRF model can be defined as follows:

$$\begin{aligned} p_\lambda(y|x) &= \frac{1}{Z(x)} \Pi_{t=1}^{T-1} \Pi_{l=1}^{L} \{exp(\sum_k \lambda_k f_k(y_{(l,t)}, y_{(l,t+1)}, x, t))\} \\ &\quad \Pi_{t=1}^{T} \Pi_{l=1}^{L-1} \{exp(\sum_k \lambda_k f_k(y_{(l,t)}, y_{(l+1,t)}, x, t))\}, \end{aligned} \tag{4.1}$$

where $p_\lambda(y|x)$ is the conditional probability of a sequence of tag vectors $y$ given the observation vector $x$ with parameter vector $\lambda$. $Z(x)$ is a normalization factor which guarantees a well-defined probability distribution. $t$ is a time index from 1 to $T$, and $l$ is a layer index from 1 to $L$. $f_k$ is the $k$-th feature function with weight $\lambda_k$. $y_{(l,t)}$ is the variable in layer $l$ at time $t$.

We also believe that punctuation correction is closely related to sentence boundary detection. Thus, we propose a two-layer DCRF model for punctuation correction. The tag sets for the two layers are shown in Table 4.3. Layer 1 gives the actual punctuation tags, while Layer 2 gives the sentence boundary tags indicating whether the current word is at the beginning of (or inside) a declarative, question, or exclamatory sentence.

For example, for the training sentence "*where ? i can not see you !*", its tags are

| Layer | Tag Set |
|---|---|
| Layer 1 | None, Comma, Period, Question-Mark, Exclamatory-Mark |
| Layer 2 | Declarative-Begin, Declarative-In, Question-Begin, Question-In, Exclamatory-Begin, Exclamatory-In |

Table 4.3: The tag sets used in the two-layer DCRF model for punctuation correction.

shown in Table 4.4.

| **Words** | where | i | can | not | see | you |
|---|---|---|---|---|---|---|
| **Layer 1** | Question-Mark | None | None | None | None | Ex-Mark |
| **Layer 2** | Question-Begin | Ex-Begin | Ex-In | Ex-In | Ex-In | Ex-In |

Table 4.4: An example of tags of the training sentence "*where ? i can not see you !*", in the two-layer DCRF model for punctuation correction. *Ex* stands for Exclamatory.

#### 4.2.2.2 Features for Punctuation Correction

We use word $n$-grams ($n = 1, 2, 3$) and punctuation symbols within 5 words before and after the current word as binary features in the DCRF model (special sentence start and end symbols are used to denote the sentence boundaries). Although the punctuation symbols in the input text are unreliable, some of them are still correct. Thus, the punctuation symbols are used as additional features for the words.

For example, Table 4.5 shows the tags and features for the word "*where*" in the message "*where*| .|*?* *i*| *can*| *not*| *see*| *you*| *!*|*!*" (hereafter, the punctuation symbols after the vertical bars are the corrected symbols). In the table, "*<s>@-1*" is a unigram feature meaning that a unigram "*<s>*" is located at one position to the left of the current word "*where*", and "*i+can+not@1*" is a trigram feature which indicates that a trigram "*i+can+not*" is located at one position to the right of the current word. Table 4.6 presents the tags and features used in our Chinese punctuation correction model for the word "旅游" in the message "神马| 时候| ,| 去| 北京| 旅游|?".

| **Tag Name** | **Content** |
|---|---|
| Layer 1 tag | *Question-Mark* |
| Layer 2 tag | *Question-Begin* |
| **Feature Type** | **Content** |
| unigram features | <s>@-1 where@0 i@1 can@2 not@3 see@4 you@5 |
| bigram features | <s>+where@-1 where+i@0 i+can@1 can+not@2 not+see@3 see+you@4 you+</s>@5 |
| trigram features | <s>+where+i@-1 where+i+can@0 i+can+not@1 can+not+see@2 not+see+you@3 see+you+</s>@4 |
| punctuation features | .@0 !@5 |

Table 4.5: An example of tags and features used in our English punctuation correction model.

#### 4.2.2.3 Training Data Construction for Punctuation Correction

Due to the lack of informal training texts with corrected punctuation, we train our punctuation correction model on formal texts with synthetically created punctuation errors.

We randomly add, delete, and substitute punctuation symbols in formal texts with equal probabilities. Specifically, for $s \in \{, .?!\}$, $P(none|s) = P(,|s) = P(.|s) = P(?|s) = P(!|s) = 0.2$ denotes the probability of replacing a punctuation symbol $s$ (replacing $s$ by $none$ denotes deletion); and for a real word (not a punctuation symbol) $w$, $P(none|w) = P(,|w) = P(.|w) = P(?|w) = P(!|w) = 0.2$ denotes the probability of inserting a punctuation symbol after $w$ (inserting $none$ after $w$ denotes no insertion).

After randomly adding, deleting, or replacing punctuation symbols in a formal text, we obtain a text with punctuation problems as well as its gold-standard correction (i.e., in the original formal text). For example, a training instance may be "*where*| .|*?* *i*| *can*| *not*| ,| *see*| *you*|*!*", where ".|*?*" means substituting "." with the correct "*?*"; ",|" means deletion of ","; and "*you*|*!*" means an insertion of "*!*" after the word "*you*".

| **Tag Name** | **Content** |
|---|---|
| Layer 1 tag | *Question-Mark* |
| Layer 2 tag | *Question-In* |
| **Feature Type** | **Content** |
| unigram features | <s>@-5 神马@-4 时候@-3 去@-2 北京@-1 旅游@0 </s>@1 |
| bigram features | <s>+神马@-5 神马+时候@-4 时候+ 去@-3 去+ 北京@-2 北京+旅游@-1 旅游+</s>@0 |
| trigram features | <s>+神马+时候@-5 神马+时候+ 去@-4 时候+去+北京@-3 去+北京+旅游@-2 北京+旅游+</s>@-1 |
| punctuation features | ,@-3 |

Table 4.6: An example of tags and features used in our Chinese punctuation correction model.

### 4.2.3 Missing Word Recovery

As shown in Section 4.1, some words are often omitted in social media texts, e.g., the pronoun "我*[I]*" in Chinese and "*be*" in English. To fix this problem, we propose a CRF model to recover such missing words, which will be used as a hypothesis producer in our beam-search decoder for text normalization. The hypothesis producer can insert missing words in the current hypothesis.

To recover a missing "*be*" in English social media text, the CRF model has five tags: *None*, *BE*, *IS*, *ARE*, and *AM*. In an input sentence, every token (including words, punctuation symbols, and a special start-of-sentence placeholder) will be assigned a tag, denoting the insertion of a form of "*be*" after the token. For example, the tags for the training sentence "*i going , where are you ?*" are presented in Table 4.7, and after applying the tags, we get a new sentence "*i am going , where are you ?*"

We use the same $n$-gram features as our punctuation correction model shown in Table 4.5, but exclude the punctuation features. The model is trained on synthetically created training texts in which "*be*" has been randomly deleted with probability 0.5. For example, a training instance can be "*i*|*AM going*| ,| *where*| *are*| *you*| *?*|", where "*i*|*AM*"

means an insertion of "*am*" after the word "*i*".

| **Words** | <s> | i | going | , | where | are | you | ? |
|---|---|---|---|---|---|---|---|---|
| **Tags** | None | AM | None | None | None | None | None | None |

Table 4.7: An example of tags of the training sentence "'*i going , where are you ?*", in the CRF model for missing word recovery. "<s>" is a special start-of-sentence placeholder.

In order to recover the Chinese pronoun "我*[I]*" in Chinese social media text, a similar CRF-based method is applied with two tags: *None* and 我.

### 4.2.4 Hypothesis Producers for Chinese Text Normalization

Taking into account the informal characteristics of Chinese social media text in Section 4.1, we design the following hypothesis producers for Chinese text normalization:

- **Dictionary**: We have manually assembled a dictionary of 703 informal-formal phrase pairs from the Internet. The pairs are used to produce new hypotheses. For example, given a hypothesis "神马*[magical horse]* 时候*[time]*", if the dictionary contains the pair "(神马,什么*[what]*)", the Dictionary hypothesis producer generates a new hypothesis "什么 时候".

- **Punctuation**: As shown in Section 4.2.2, a punctuation correction model is adopted to correct punctuation in the current hypothesis, e.g., it may normalize "什么*[what]* 时候*[time]*" into "什么 时候 ？".

- **Pronunciation**: We use Chinese Pinyin to model the pronunciation similarity of words. To accomplish this, we pair some Pinyin initials that sound similar into a group, because in some Chinese dialects, people do not distinguish the Pinyin initials in a group, which is one of the most important sources of informal Chinese words. The groups of paired Pinyin initials are *(c, ch)*, *(s, sh)*, and *(z, zh)*.

For example, given the hypothesis "北京*[Beijing]* 筒子*[tube]* 来了*[come]*", the Pinyin of the informal word "筒子" is "*t ong z i*". The Pinyin of the formal word "同志*[comrade]*" is "*t ong zh i*". Since the similar sounding Pinyin initials *z* and *zh* are paired in a group, a new hypothesis "北京 同志 来了" can be produced. In practice, this hypothesis producer can propose many spurious candidates $w'$ for an informal word $w$. As such, after we replace $w$ by $w'$ in the hypothesis, we require that some 4-gram containing $w'$ and its surrounding words in the hypothesis appears in a formal corpus. We call this filtering process *contextual filtering*. For example, given the informal word "筒子*[tube]*" in the hypothesis "北京*[Beijing]* 筒子*[tube]* 来了*[come]*", the Pronunciation hypothesis producer proposes formal candidates "同志*[comrade]*, 铜质*[copper]*, ...". If the 4-gram "<s> 北京 同志 来了" exists in a large formal corpus, a new hypothesis "北京 同志 来了" can be successfully produced. If the 4-gram "<s> 北京 铜质 来了" or "北京 铜质 来了 </s>" never appears in a large formal corpus, we discard the candidate "铜质".

- **Pronoun**: With the method of Section 4.2.3, a CRF model is trained to recover the missing pronoun "我*[I]*". For example, this hypothesis producer may normalize " 喜欢*[like]* 这个*[this]*" into "我 喜欢 这个".

- **Interjection**: If a word $w$ in a pre-defined list[1] of frequent redundant interjections appears at the end of a sentence, we produce a new hypothesis by removing $w$, e.g., from "好的*[ok]* 哦*[oh]*" to " 好的"; "哦*[oh]* 知道了*[know it]*" will not be normalized, since the interjection "哦" is not at the end of the sentence.

- **Resegmentation**: This hypothesis producer fixes word segmentation problems. If an informal word is a concatenation of two constituent informal words $w_1$ and $w_2$ in our normalization dictionary, the informal word will be segmented into two words $w_1$ and $w_2$. As a result, the Dictionary hypothesis producer can subsequent-

[1]咯, 哦, 呐, 呗, 哇, 哈, 哟, 吖, 乃

ly normalize $w_1$ and $w_2$. For example, given "表酱紫 好吗*[ok]*", if we have "(表*[watch]*, 不要*[don't]*)" and "(酱紫, 这样子*[this]*)" in the normalization dictionary, a new hypothesis "表 酱紫 好吗" will be produced. Thus, the Dictionary hypothesis producer can subsequently normalize the new hypothesis into "不要 这样子 好吗".

Other hypothesis producers may also be useful for social media text normalization, e.g., emoticon normalization. We have not done emoticon normalization, because our data have very few emoticons and also our goal is to propose a general framework which can then be adapted to fit different kinds of social media texts by adding new hypothesis producers.

### 4.2.5 Hypothesis Producers for English Text Normalization

Considering the informal characteristics of English social media text as presented in Section 4.1, we design the following hypothesis producers for English text normalization:

- **Dictionary**: Similar to Chinese text normalization, we have manually assembled a dictionary of 4,705 informal-formal phrase pairs from the Internet. The pairs are used to produce new hypotheses. For example, given a hypothesis "*r you there*", if the dictionary contains the pair "*(r, are)*", the Dictionary hypothesis producer generates a new hypothesis "*are you there*".

- **Punctuation**: A punctuation correction model (see Section 4.2.2) is adopted to correct punctuation in the current hypothesis, e.g., it may normalize "*are you there*" into "*are you there ?*".

- **Interjection**: If a word $w$ in a pre-defined list[2] of frequent redundant interjections appears at the end of a sentence, we produce a new hypothesis by removing $w$, e.g., from "*ok lor*" to "*ok*".

[2] *ah, ba, hah, hor, huh, k, la, lah, lao, lar, le, leh, lei, liao, lie, lo, loh, lor, ma, mah, meh, wat, yah*

- **Pronunciation**: This hypothesis producer uses pronunciation similarity to find formal candidates for a given informal word. It considers a word as a sequence of letters and converts it into a sequence of phones using phrase-based SMT trained on the CMU pronouncing dictionary (Weide, 1998). Similar sounding phones are paired together in a group: *(ah, ao)*, *(ow, uw)*, and *(s, z)*. To illustrate, in the hypothesis "*wat is it*", the informal word "*wat*" maps to the phone sequence "*w ao t*". Since the formal word "*what*" maps to the phone sequence "*w ah t*" and the phones *ah* and *ao* are paired in a group, the new hypothesis "*what is it*" is generated.

- **Be**: We train a CRF model to recover missing words *be*, as described in Section 4.2.3. For example, the producer can normalize "*i going home*" to "*i am going home*".

- **Retokenization**: This hypothesis producer fixes tokenization problems. More precisely, given an informal word which is not a URL or email address and contains a period, it splits the informal word at the period. For example, "*how r u.where r u*" is normalized to "*how r u . where r u*".

- **Prefix**: This hypothesis producer generates a formal word $w'$ for an informal word $w$ if $w$ is a prefix of $w'$. To avoid spurious candidates, we only generate $w'$ if $|w| \geq 3$ and $|w'| - |w| \leq 4$. For example, given "*i am goin now*", a new hypothesis "*i am going now*" can be produced.

- **Quotation**: If an informal word ends with a letter in *(m, s, t)* and if the word produced by inserting a quotation mark before the letter is a formal word, a new hypothesis with the quotation mark inserted is produced. This hypothesis producer thus generates "*i 'm*" from "*im*", "*she 's*" from "*shes*", "*isn 't*" from "*isnt*", etc. For example, given the hypothesis "*im here now*", a new hypothesis "*i 'm here now*" can be produced.

- **Abbreviation**: Letters denoting the vowels in a formal word are often deleted to form an informal word. This hypothesis producer generates a formal word $w'$ from an informal word $w$ if $w'$ can be obtained from $w$ by adding missing vowels. To avoid spurious candidates, we only consider $w$ where $|w| \geq 2$. For example, this hypothesis producer can normalize "*gd morning , everyone*" to "*good morning , everyone*".

- **Time**: If a number can be a potential time expression and appears after "*at*" or before "*am*" or "*pm*", a new hypothesis is produced by changing the number into a time expression, e.g., "*1130 am*" is normalized to "*11 : 30 am*".

Since the Pronunciation, Prefix, and Abbreviation hypothesis producers can propose spurious candidates for an informal word, we also use contextual filtering (See Section 4.2.4) to further filter the candidates for these hypothesis producers.

## 4.3 Experiments

In this section, we will first introduce the evaluation corpora used in our Chinese-English and English-Chinese normalization and translation experiments, followed by a description of the machine translation systems used for evaluating the success of text normalization. The baselines will then be introduced. Subsequently, the experimental results of our Chinese-English and English-Chinese experiments will be presented and discussed. Finally, some further analyses will be described.

### 4.3.1 Evaluation Corpora

As previous work (Choudhury et al., 2007; Han and Baldwin, 2011; Liu et al., 2012) mostly focused on word normalization, no data is available with corrected punctuation and recovered missing words. We thus create the following two corpora for social media text normalization and translation:

- **Chinese-English corpus:** We crawled 1,000 messages from Weibo which were first normalized into formal Chinese and then translated into formal English. The first half of the corpus serves as our development set to tune our text normalization decoder for Chinese, while the second half serves as the test set to evaluate text normalization for Chinese and Chinese-English machine translation. The statistics of the corpus are shown in Table 4.8.

- **English-Chinese corpus:** From the NUS English SMS corpus (How and Kan, 2005), we randomly selected 2,000 messages. The messages were first normalized into formal English and then translated into formal Chinese. Similar to the Chinese-English corpus, the first half of the corpus serves as our development set while the second half serves as the test set. The statistics of the corpus are shown in Table 4.9.

| **Corpus** | **# messages** | **# tokens (EN/CN/NCN)** |
|---|---|---|
| *CN2EN-dev* | 500 | 6.95K/5.45K/5.70K |
| *CN2EN-test* | 500 | 7.14K/5.64K/5.82K |

Table 4.8: Statistics of the corpus used in Chinese-English social media text normalization and translation experiments. *CN2EN-dev*/*CN2EN-test* is the development/test set in our Chinese-English experiments. *NCN* denotes manually normalized Chinese texts.

| **Corpus** | **# messages** | **# tokens (EN/CN/NEN)** |
|---|---|---|
| *EN2CN-dev* | 1,000 | 16.63K/18.14K/18.21K |
| *EN2CN-test* | 1,000 | 16.14K/17.69K/17.76K |

Table 4.9: Statistics of the corpus used in English-Chinese social media text normalization and translation experiments. *EN2CN-dev*/*EN2CN-test* is the development/test set in our English-Chinese experiments. *NEN* denotes manually normalized English texts.

In Section 4.2, a formal corpus is used to: (1) detect informal words; (2) train the punctuation correction and missing word recovery models; and (3) perform contextual

filtering. The formal corpus is the concatenation of two Chinese-English spoken parallel corpora: the IWSLT 2009 corpus (Paul, 2009) and another spoken text corpus collected at the Harbin Institute of Technology[3]. The language model used for Chinese (English) text normalization is the Chinese (English) side of the formal corpus and the LDC Chinese (English) Gigaword corpus.

Following (Aw et al., 2006; Oliva et al., 2012), we use BLEU scores (Papineni et al., 2002) to evaluate text normalization. We also use BLEU scores to evaluate machine translation quality. We use the sign test to determine statistical significance, for both text normalization and translation.

### 4.3.2 Machine Translation Systems

To evaluate the effect of text normalization on machine translation, we build phrase-based machine translation systems using Moses (Koehn et al., 2007) with formal parallel corpora.

We first build separate directed word alignments using IBM model 4 (Brown et al., 1993) for both directions of the training parallel text, and then combine the word alignments using the intersect+grow heuristic (Och and Ney, 2003). From the combined word alignments, a phrase table containing phrase-level translation pairs whose length is up to seven is extracted using the alignment template approach (Och and Ney, 2004). In the phrase table, each phrase pair has five features (Koehn, 2013): forward and reverse translation probabilities, forward and reverse lexical weighting, and a (fixed) phrase penalty. A log-linear model is adopted to combine the five features in the phrase table, a 5-gram language model score, word penalty, distance-based reordering cost, and six features for the lexical reordering model (*msd-bidirectional-fe* in Moses) (Koehn, 2013). The weights of the log-linear model are tuned to optimize the BLEU score (Papineni et al., 2002) on the manually normalized messages of our development sets using minimum er-

[3] `http://mitlab.hit.edu.cn/`

ror rate training (MERT) (Och, 2003). The phrase-based SMT decoder of Moses is used to perform translation with the log-linear model. The language model is trained with the SRILM toolkit (Stolcke, 2002) and modified Kneser-Ney smoothing (Kneser and Ney, 1995).

The training parallel corpora include the formal corpus described in Section 4.3.1 and some LDC[4] parallel corpora as shown in Table 4.10. The language model training data of the Chinese-English (English-Chinese) machine translation system is the English (Chinese) half of the FBIS corpus and the English (Chinese) Gigaword corpus.

| **Corpus** | **LDC catalog #** | **Size (EN/CN)** |
|---|---|---|
| *IWSLT09* | - | 765/630 |
| *HIT* | - | 524/486 |
| *Hong Kong News Parallel Text* | LDC2000T46 | 16,863/15,127 |
| *Xinhua* | LDC2002E18 | 4,071/3,934 |
| *FBIS* | LDC2003E14 | 10,097/7,767 |
| *United Nations* | LDC2004E12 | 167,892/150,611 |
| *Chinese News Translation Text Part 1* | LDC2005T06 | 321/283 |
| *Chinese English News Magazine* | LDC2005T10 | 5,570/6,442 |
| *GALE Phase 1 Blog* | LDC2008T06 | 191/169 |
| *GALE Phase 1 Broadcast News - Part 1* | LDC2007T23 | 271/239 |
| *GALE Phase 1 Broadcast News - Part 2* | LDC2008T08 | 255/223 |
| *GALE Phase 1 Broadcast News - Part 3* | LDC2008T18 | 176/149 |
| *GALE Phase 1 Broadcast Conversation - Part 1* | LDC2009T02 | 230/207 |
| *GALE Phase 1 Broadcast Conversation - Part 2* | LDC2009T06 | 255/233 |
| *GALE Phase 1 Newsgroup - Part 1* | LDC2009T15 | 153/133 |
| *GALE Phase 1 Newsgroup - Part 2* | LDC2010T03 | 145/125 |

Table 4.10: Statistics of the parallel corpora used to train our SMT systems. Sizes are in thousands of words.

[4] `http://www.ldc.upenn.edu/Catalog/`

### 4.3.3 Baselines

We compare our text normalization decoder against three baseline methods for performing text normalization. We then send the respective normalized texts to the same machine translation system to evaluate the effect of text normalization on machine translation.

The simplest baseline for text normalization is one that does no text normalization. The raw text (un-normalized) is simply passed on to the machine translation system for translation. We call this baseline ORIGINAL.

The second baseline, LATTICE, is to use a lattice to normalize text. For each input message, a lattice is generated in which each informal word is augmented with its formal candidates taken from the same normalization dictionary (downloaded from Internet) used in our text normalization decoder. The lattice is then decoded by the same language model used in our text normalization decoder to generate the normalized text (Stolcke, 2002). Another possible way of using lattice is to directly feed the lattice to the machine translation system (Eidelman et al., 2011), but since in our work, we assume that the machine translation system can only translate plain text, we leave this as future work.

The third baseline, PBMT, is a competitive baseline that performs text normalization via phrase-based machine translation (PBMT), as proposed by Aw et al. (2006). Moses (Koehn et al., 2007) is used to perform text normalization, by "translating" un-normalized text to normalized text. The training data used is the same development set used in our text normalization decoder. The normalized text is then sent to our machine translation system for translation. This method was also used in the SMS translation task of WMT 2011 by Stymne (2011).

In the tables showing experimental results, normalization and translation BLEU scores that are significantly higher than ($p < 0.01$) the LATTICE or PBMT baseline are **in bold** or underlined, respectively.

### 4.3.4 Chinese-English Experimental Results

The Chinese-English normalization and translation results are shown in Table 4.11. The first group of experiments is the three baselines, and the second group is an oracle experiment using manually normalized messages as the output of text normalization which indicates the theoretical upper bounds of perfect normalization. In the normalization experiments, the ORIGINAL baseline gets a BLEU score of 61.01%, and the LATTICE baseline greatly improves the ORIGINAL baseline by 13.51%, which shows that the dictionary collected from the Internet is highly effective in text normalization. The PBMT baseline further improves the BLEU score by 2.25%. In the corresponding machine translation experiments, as the normalization BLEU scores increase, the translation BLEU scores also increase.

| | **BLEU scores (%)** | |
|---|---|---|
| **System** | **Normalization** | **Translation** |
| ORIGINAL baseline | 61.01 | 9.06 |
| LATTICE baseline | 74.52 | 11.50 |
| PBMT baseline | 76.77 | 12.65 |
| ORACLE | 100.00 | 15.04 |
| Dictionary | **77.80** | 12.35 |
| Punctuation | 65.95 | 9.63 |
| Pronunciation | 61.30 | 9.13 |
| Pronoun | 61.11 | 9.01 |
| Interjection | 61.05 | 9.14 |
| Resegmentation | 60.98 | 9.03 |
| Dictionary | **77.80** | 12.35 |
| +Punctuation | **84.69** | **13.37** |
| +Pronunciation | **84.69** | **13.40** |
| +Pronoun | **84.96** | **13.50** |
| +Interjection | **85.33** | **13.68** |
| +Resegmentation | **86.75** | **14.03** |

Table 4.11: Chinese-English experimental results of social media text normalization and translation. Normalization and translation scores that are significantly higher than ($p < 0.01$) the LATTICE or PBMT baseline are **in bold** or underlined, respectively.

The third group is the isolated experiments, i.e., each experiment only uses one hypothesis producer. As expected, the individual hypothesis producers alone do not work well except the Dictionary hypothesis producer, which shows the importance of normalization dictionaries in social media text normalization. One interesting discovery is that the Dictionary hypothesis producer outperforms the LATTICE baseline, which shows that our normalization decoder can utilize the dictionary more effectively, probably because of the additional features used in our normalization decoder such as the informal word penalty. The Resegmentation hypothesis producer alone worsens the BLEU scores, since it can only split informal words, and is designed to work together with other hypothesis producers to normalize words.

The last group is the combined experiments. We add each hypothesis producer in the order of its normalization effectiveness in the isolated experiments. Adding the Punctuation hypothesis producer greatly improves the BLEU scores of both normalization and translation, which confirms the importance of punctuation correction. The Pronoun and Interjection hypothesis producers also contribute some improvements. Finally, Resegmentation significantly improves the normalization/translation BLEU scores by 1.42%/0.35%. Compared with the isolated experiments, the combined experiments show that our normalization decoder can effectively integrate different hypothesis producers to achieve better performance for both text normalization and translation.

Overall, in the Chinese text normalization experiments, our normalization decoder outperforms the best baseline PBMT by 9.98% in BLEU score. In the Chinese-English machine translation experiments, the normalized texts output by our normalization decoder lead to improved translation quality compared to normalization by the PBMT baseline, by 1.38% in BLEU score.

### 4.3.5 English-Chinese Experimental Results

The English-Chinese normalization and translation results are shown in Table 4.12, with the same experimental setup as in the Chinese-English experiments.

| | BLEU scores (%) | |
|---|---|---|
| **System** | **Normalization** | **Translation** |
| ORIGINAL baseline | 37.38 | 13.63 |
| LATTICE baseline | 56.98 | 20.56 |
| PBMT baseline | 59.19 | 21.46 |
| ORACLE | 100.00 | 28.48 |
| Dictionary | **59.90** | **20.84** |
| Retokenization | 38.79 | 14.06 |
| Prefix | 38.68 | 13.90 |
| Interjection | 38.37 | 13.92 |
| Quotation | 38.04 | 13.65 |
| Abbreviation | 37.94 | 13.74 |
| Time | 37.65 | 13.66 |
| Pronunciation | 37.62 | 13.80 |
| Punctuation | 37.62 | 13.79 |
| Be | 37.47 | 13.59 |
| Dictionary | **59.90** | **20.84** |
| +Retokenization | **62.27** | **21.70** |
| +Prefix | **63.22** | **21.88** |
| +Interjection | **64.85** | **22.30** |
| +Quotation | **65.24** | **22.31** |
| +Abbreviation | **65.35** | **22.34** |
| +Time | **65.59** | **22.38** |
| +Pronunciation | **65.64** | **22.38** |
| +Punctuation | **66.38** | **22.74** |
| +Be | **66.54** | **22.81** |

Table 4.12: English-Chinese experimental results of social media text normalization and translation. Normalization and translation scores that are significantly higher than ($p < 0.01$) the LATTICE or PBMT baseline are **in bold** or underlined, respectively.

The text normalization BLEU score of the ORIGINAL baseline is much lower in English compared to Chinese, since the English texts contain more informal words.

Again, in the isolated experiments, the individual hypothesis producers alone do not work well, except the Dictionary hypothesis producer.

In the combined experiments, the Retokenization hypothesis producer greatly improves the normalization/translation BLEU scores by 2.37%/0.86%. The Punctuation hypothesis producer helps less for English compared to Chinese, suggesting that our Chinese texts contain noisier punctuation.

Overall, we achieved similar improvements in English text normalization and English-Chinese translation, and the improvements in BLEU scores are 7.35% and 1.35% respectively.

### 4.3.6 Further Analysis

**The effect of contextual filtering.** To measure the effect of contextual filtering proposed in Section 4.2.4, we ran our normalization decoder without contextual filtering. We obtained BLEU scores of 65.05%/22.38% in the English-Chinese experiments, which were lower than 66.54%/22.81% obtained with contextual filtering. This shows the beneficial effect of contextual filtering.

**Decoding speed.** On a machine with a 2.27 GHz Intel Xeon CPU and 32 GB memory, the decoding speed of our Chinese text normalization decoder was 0.22 seconds per message on our test sets; the LATTICE baseline used 0.88 seconds per message; and the PBMT baseline used 1.00 seconds per message. On the same machine, the speed of our English text normalization decoder was 0.68 seconds per message on our test sets; the LATTICE baseline used 0.73 seconds per message; and the PBMT baseline used 0.90 seconds per message.

**The effect of text normalization decoder on machine translation.** We manually analyzed the effect of our text normalization decoder on machine translation. For example, in English-Chinese social media text normalization/translation, given the unnormalized English test message "*yeah must sign up , im in lt25*" , our English-Chinese

machine translation system translated it into "对*[yeah]* 必须*[must]* 签署*[sign up]* ， im 在*[in]* lt25" On the other hand, our English text normalization decoder normalized it into "*yeah must sign up , i 'm in lt25 .*" which was then translated into "对 必须 签署 , 我 在 lt25 。" by our machine translation system. This example shows that our English text normalization decoder uses word normalization and punctuation correction to improve machine translation. For a Chinese-English example, given the un-normalized Chinese test message "灰常 感谢*[thank]* 爱老虎油", our Chinese-English machine translation system translated it into "灰常 thanked 爱老虎油". On the other hand, our Chinese text normalization decoder normalized it into "非常*[very]* 感谢*[thank]* 我*[i]* 爱*[love]* 你*[you]* 。", which was then translated into "*thank you very much , i love you .*" by our translation system. This example shows that our Chinese text normalization decoder is also able to use word normalization and punctuation correction to improve machine translation quality.

**The difference between the text normalization decoder and phrase-based SMT decoder.** As shown in Table 4.11 and 4.12, the PBMT baseline is quite competitive, which may be due to the reason that its training data and test data are quite similar. Still, our text normalization decoder achieved statistically significant improvements over the PBMT baseline in both social media text normalization and translation. The important differences between the text normalization decoder and the PBMT baseline are as follows: (1) PBMT needs a relatively large amount of parallel corpora containing raw and manually normalized messages, while the text normalization decoder requires no such kind of training data; (2) PBMT is limited by its training data, which prevents it from normalizing new informal words which are frequent in social media texts. In contrast, the text normalization decoder has the ability to normalize new informal words, e.g., by using the Pronunciation hypothesis producer, and it can be easily extended to normalize new informal words, e.g., by adding new informal-formal phrase pairs to the Dictionary hypothesis producer; (3) the text normalization decoder can use more types of feature

functions, because of its general framework and the fact that the text normalization decoder performs beam search at the sentence level, and not the phrase level.

## 4.4 Summary

This chapter presents our social media text normalization work for machine translation using the text rewriting decoder proposed in Chapter 3. Previous work on normalization of social media text mostly focused on normalizing words by substituting an informal word with its formal form. To further improve machine translation, we argue that other normalization operations should also be performed, e.g., punctuation correction and missing word recovery. We propose to use our text rewriting decoder which can effectively integrate different normalization operations. To show the applicability of our approach, we experiment with two languages, Chinese and English. In our experiments, we achieved statistically significant improvements over two strong baselines: an improvement of 9.98%/7.35% in BLEU scores for normalization of Chinese/English social media text, and an improvement of 1.38%/1.35% in BLEU scores for translation of Chinese/English social media text.

As far as we know, our work is the first to perform missing word recovery and punctuation correction for normalization of social media text, and also the first to perform message-level normalization of Chinese social media text. We investigate the effects on translating social media text after addressing various characteristics of informal social media text through normalization. We also created two corpora: a Chinese corpus containing 1,000 Weibo messages with their normalizations and English translations; and another similar English corpus containing 2,000 SMS messages from the NUS SMS corpus (How and Kan, 2005). As far as we know, our corpora are the first publicly available Chinese/English corpora for normalization and translation of social media text[5].

[5]Available at `www.comp.nus.edu.sg/~nlp/corpora.html`

Future work can investigate how to more tightly integrate our beam-search decoder for text normalization with a standard machine translation decoder. For example, one possible direction is to get an n-best list as the normalization output for each input message and then translate each output in the n-best list using the machine translation system, and finally select the best translation output generated by the translation system. Another potential direction is to generate a lattice as the normalization output from the text normalization decoder, and then translate the lattice using the translation system (Dyer, 2007).

# Chapter 5

# Source Language Adaptation for Resource-Poor Machine Translation

In this chapter, we will apply our text rewriting decoder presented in Chapter 3 to source language adaptation for resource-poor machine translation. More precisely, assuming that we have a large bi-text for a resource-rich language and another small bi-text for a related resource-poor language, we use the text rewriting decoder to adapt the resource-rich bi-text to get closer to the resource-poor language. Eventually, the adapted bi-text is used to help machine translation of the resource-poor language.

We compare the text rewriting decoder approach with two approaches proposed in our previous work (Wang et al., 2012a): (1) word-level paraphrasing approach using confusion networks; and (2) phrase-level paraphrasing approach using pivoted phrase tables.

In the remainder of this chapter, we will first discuss the closely related language pair (Malay and Indonesian) that we will focus on in our experiments. Then the text rewriting decoder for source language adaptation will be presented, followed by the other two approaches in our previous work. Next, we will introduce the bi-text combination methods. Then we will present the experiments and discussions, as well as some further

analysis. Lastly we will summarize the whole chapter.

## 5.1 Malay and Indonesian

Malay and Indonesian are closely related, mutually intelligible Austronesian languages with 180 million speakers combined. They have a unified spelling, with occasional differences, e.g., *kerana* vs. *karena* ("because"), *Inggeris* vs. *Inggris* ("English"), and *wang* vs. *uang* ("money").

They differ more substantially in vocabulary, mostly because of loan words, where Malay typically follows the English pronunciation, while Indonesian tends to follow Dutch, e.g., *televisyen* vs. *televisi*, *Julai* vs. *Juli*, and *Jordan* vs. *Yordania*.

While there are many cognates between the two languages, there are also a lot of false friends, e.g., *polisi* means *policy* in Malay but *police* in Indonesian. There are also many partial cognates, e.g., *nanti* means both *will* (future tense marker) and *later* in Malay but only *later* in Indonesian.

Thus, fluent Malay and fluent Indonesian can differ substantially. Consider, for example, Article 1 of the *Universal Declaration of Human Rights*[1]:

- *Semua manusia dilahirkan bebas dan samarata dari segi kemuliaan dan hak-hak. Mereka mempunyai pemikiran dan perasaan hati dan hendaklah bertindak di antara satu sama lain dengan semangat persaudaraan.* ***(Malay)***

- *Semua orang dilahirkan merdeka dan mempunyai martabat dan hak-hak yang sama. Mereka dikaruniai akal dan hati nurani dan hendaknya bergaul satu sama lain dalam semangat persaudaraan.* ***(Indonesian)***

- *All human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood.* ***(English)***

[1] `http://www.un.org/en/documents/udhr/index.shtml`

There is only 50% overlap at the word level, but the actual vocabulary overlap is much higher, e.g., there is only one word in the Malay text that does not exist in Indonesian: *samarata* ("equal"). Other differences are due to the use of different morphological forms, e.g., *hendaklah* vs. *hendaknya* ("conscience"), derivational variants of *hendak* ("want").

Of course, word choice in translation is often a matter of taste. Thus, we asked a native speaker of Indonesian to adapt the Malay version to Indonesian while preserving as many words as possible:

- *Semua manusia dilahirkan bebas dan mempunyai martabat dan hak-hak yang sama. Mereka mempunyai pemikiran dan perasaan dan hendaklah bergaul satu sama lain dalam semangat persaudaraan.* (**Indonesian**)

Obtaining this latter version from the original Malay text requires three word-level operations: (1) deletion of *dari*, *segi*, (2) insertion of *yang*, *sama*, and (3) substitution of *samarata* with *mempunyai*.

Unfortunately, we do not have parallel Malay-Indonesian text, which complicates the process of learning when to apply these operations. Thus, below we restrict our attention to the simplest and most common operation of word/phrase substitution only, leaving the other two[2] operations for future work.

Note that word substitution is enough in many cases, e.g., it is all that is needed for the following Malay-Indonesian sentence pair:

- *KDNK Malaysia dijangka cecah 8 peratus pada tahun 2010. (**Malay**)*
- *PDB Malaysia akan mencapai 8 persen pada tahun 2010. (**Indonesian**)*
- *Malaysia's GDP is expected to reach 8 percent in 2010. (**English**)*

[2]There are other potentially useful operations, e.g., a correct translation for the Malay *samarata* can be obtained by splitting it into the Indonesian sequence *sama rata*.

## 5.2 Methods

Assuming a resource-rich bi-text (Malay-English) and a resource-poor bi-text (Indonesian-English), we improve machine translation from the resource-poor language (Indonesian) to English by using our text rewriting decoder to *adapt* the bi-text for the related resource-rich language (Malay) and English to the resource-poor language (Indonesian) and English. After adaptation, we combine the adapted bi-text with the original resource-poor bi-text using three bi-text combination methods.

More specifically, given a Malay sentence in the resource-rich Malay-English bi-text, we use an adaptation method to adapt the Malay sentence to a ranked list of $n$ corresponding adapted "Indonesian" sentences. The adaptation method can be the text rewriting decoder, word-level paraphrasing approach, or phrase-level paraphrasing approach. Then, we pair each such adapted "Indonesian" sentence with the English counter-part in the bi-text for the Malay sentence it was derived from, thus obtaining a synthetic "Indonesian"-English bi-text. Finally, we combine this synthetic bi-text with the resource-poor Indonesian-English bi-text to train the final Indonesian-English SMT system.

In this section, we will first present a text rewriting decoder for source language adaptation of bi-text. In order to compare the decoder with other approaches, the other two approaches for source language adaptation proposed in our previous work will then be described. Lastly, the three bi-text combination methods will be described.

### 5.2.1 A Text Rewriting Decoder for Source Language Adaptation

Based on the text rewriting decoder presented in Chapter 3, we propose a source language adaptation decoder to adapt the Malay-English bi-text into "Indonesian"-English to help Indonesian-English translation.

The beam search algorithm for source language adaptation is shown in Algorithm

3. The algorithm returns a ranked list of best adapted "Indonesian" sentences for a given Malay sentence, while previously Algorithm 1 and 2 only output the 1-best for each given input.

**Algorithm 3** Beam-Search Source-Language Adaptation

INPUT: an input **INPUT** whose length is **N**
RETURN: a ranked list of best rewritten forms for **INPUT**

```
1:  initialize hypothesisStacks[0...N] and hypothesisProducers;
2:  add the initial hypothesis INPUT to stack hypothesisStacks[0];
3:  for i ← 0 to N-1 do
4:    for each hypo in hypothesisStacks[i] do
5:      for each producer in hypothesisProducers do
6:        for each newHypo produced by producer from hypo do
7:          detect Malay words in newHypo;
8:          add newHypo to hypothesisStacks[i+1];
9:          prune hypothesisStacks[i+1];
10: return a ranked list of best hypotheses in all stacks hypothesisStacks[0...N];
```

In the following subsections, we will first introduce how we generate three resources which will be used by the text rewriting decoder, i.e., a word-level pivoted Malay-Indonesian dictionary, a pivoted Malay-Indonesian phrase table, and a cross-lingual morphological variant dictionary. We will then present the hypothesis producers and feature functions proposed for source language adaptation.

#### 5.2.1.1 Inducing Word-Level Paraphrases

We use pivoting over English to induce potential Indonesian word translations for a given Malay word.

First, for the Malay-English bi-text and the Indonesian-English bi-text, we build separate directed word alignments using IBM model 4 (Brown et al., 1993), and then combine the directed word alignments using the intersect+grow heuristic (Och and Ney, 2003). We then induce Indonesian-Malay word translation pairs assuming that if an Indonesian word $i$ and a Malay word $m$ are aligned to the same English word $e$, they could

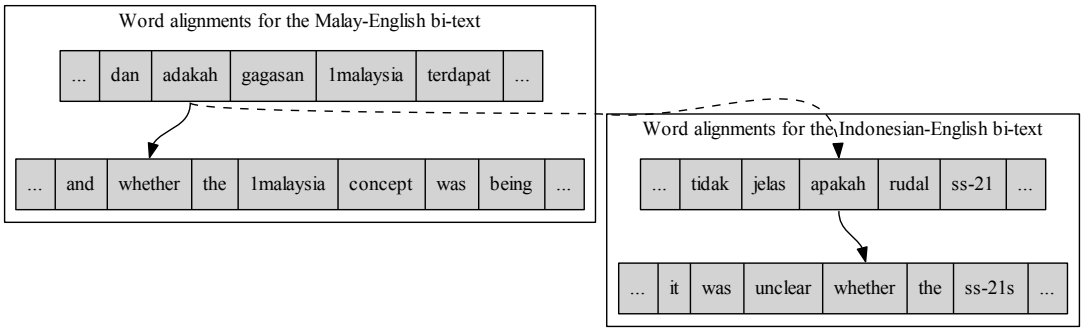


Figure 5.1: **An example of word-level paraphrase induction by pivoting over English.** The Malay word *adakah* is aligned to the English word *whether* in the Malay-English bi-text (solid arcs). The Indonesian word *apakah* is aligned to the same English word *whether* in the Indonesian-English bi-text. We consider *apakah* as a potential translation option of *adakah* (the dashed arc). Other word alignments are not shown.

be mutual translations. Each translation pair is associated with a conditional probability, estimated by pivoting over English:

$$\Pr(i|m) = \sum_{e} \Pr(i|e)\Pr(e|m) \tag{5.1}$$

$\Pr(i|e)$ and $\Pr(e|m)$ are estimated using maximum likelihood from the word alignments. Following (Callison-Burch et al., 2006), we further assume that $i$ is conditionally independent of $m$ given $e$.

For example, Figure 5.1 shows an example which induces an Indonesian word *apakah* as a translation option for the Malay word *adakah*, since the two words are both aligned to the same English word *whether* in the word alignments for the Indonesian-English bi-text and the Malay-English bi-text, respectively.

#### 5.2.1.2 Inducing Phrase-Level Paraphrases

We use standard phrase-based SMT techniques (Koehn et al., 2007) to build separate phrase tables for the Indonesian-English and the Malay-English bi-texts. More specifically, based on the combined word alignments for the two bi-texts in Section 5.2.1.1, two phrase tables are extracted using the alignment template approach (Och and Ney, 2004), respectively. In the phrase tables, we have four phrase translation scores: forward/reverse phrase translation probability, and forward/reverse lexical weighting. We pivot over English phrases to generate Indonesian-Malay phrase pairs, whose scores are derived from the corresponding ones in the two phrase tables using Equation 5.1.

Following Koehn (2010), if we are translating a foreign language $f$ to English $e$, the forward ($\phi(\bar{e}|\bar{f})$) and reverse ($\phi(\bar{f}|\bar{e})$) phrase translation probabilities are defined as follows:

$$\phi(\bar{e}|\bar{f}) = \frac{count(\bar{e},\bar{f})}{\sum_{\bar{e}'} count(\bar{e}',\bar{f})} \tag{5.2}$$

$$\phi(\bar{f}|\bar{e}) = \frac{count(\bar{e},\bar{f})}{\sum_{\bar{f}'} count(\bar{e},\bar{f}')} \tag{5.3}$$

$\bar{f}$ and $\bar{e}$ are phrases in the two languages, and $count(\bar{e},\bar{f})$ is the count of sentence pairs from which a particular phrase pair $(\bar{e},\bar{f})$ is extracted.

The forward ($lex(\bar{e}|\bar{f})$) and reverse ($lex(\bar{f}|\bar{e})$) lexical weighting scores are defined as follows:

$$lex(\bar{e}|\bar{f},a) = \prod_{i=1}^{length(\bar{e})} \frac{1}{|\{j|(i,j)\in a\}|} \sum_{j\in\{j|(i,j)\in a\}} \omega(e_i|f_j) \tag{5.4}$$

$$lex(\bar{f}|\bar{e},a) = \prod_{j=1}^{length(\bar{f})} \frac{1}{|\{i|(i,j)\in a\}|} \sum_{i\in\{i|(i,j)\in a\}} \omega(f_j|e_i) \tag{5.5}$$

$f_j$ and $e_i$ are words in the phrases $\bar{f}$ and $\bar{e}$, respectively. $a$ is the word alignments between $\bar{f}$ and $\bar{e}$. $\omega(e_i|f_j)$ is the word translation probability estimated from the word alignments.

#### 5.2.1.3 Inducing Cross-Lingual Morphological Variants

Assuming a large monolingual Indonesian text, we first build a lexicon of the words in the text. Then, we lemmatize these words using two different lemmatizers: the Malay lemmatizer of Baldwin and Awab (2006), and a similar Indonesian lemmatizer. Since these two analyzers have different strengths and weaknesses, we combine their outputs to increase recall. Next, we group all Indonesian words that share the same lemma, e.g., for *minum*, we obtain {*diminum, diminumkan, diminumnya, makan-minum, makanan-minuman, meminum, meminumkan, meminumnya, meminum-minuman, minum, minum-minum, minum-minuman, minuman, minumanku, minumannya, peminum, peminumnya, perminum, terminum*}. Since Malay and Indonesian are subject to the same morphological processes and share many lemmata, we use such groups to propose Indonesian translation options for a Malay word. We first lemmatize the target Malay word, and then we find all groups of Indonesian words the Malay lemma belongs to. The union of these groups is the set of morphological variants for the Malay word.

While the different morphological forms typically have different meanings, e.g., *minum* ("drink") vs. *peminum* ("drinker"), in some cases the forms could have the same translation in English, e.g., *minum* ("drink", verb) vs. *minuman* ("drink", noun). This is our motivation for trying morphological variants, even though they are almost exclusively derivational, and thus generally somewhat risky as translational variants.

For example, given *seperminuman* ("drinking") in the Malay input, we first find its stem *minum*, and then we get the above example set of Indonesian words, which contains some reasonable substitutes such as *minuman* ("drink").

We give each Malay-Indonesian morphological variant pair a score $\mathrm{Score}(i, m)$ which is one minus the minimum edit distance ratio (Ristad and Yianilos, 1998) between the Malay word $m$ and the Indonesian word $i$:

$$\mathrm{Score}(i, m) \quad = \quad 1 - \frac{\mathrm{EditDistance}(i, m)}{\max(len(i), len(m))} \tag{5.6}$$

$\mathrm{EditDistance}(i, m)$ is the Levenshtein edit distance between the Indonesian word

$i$ and the Malay word $m$. $len(w)$ is the length of a word $w$ (i.e., the number of characters in $w$).

#### 5.2.1.4 Hypothesis Producers

We design the following hypothesis producers in our source language adaptation decoder:

- Word-level mapping: This hypothesis producer uses the word-level pivoted Malay-Indonesian dictionary described in Section 5.2.1.1. For example, given the hypothesis "*KDNK Malaysia dijangka cecah 8.1 peratus pada tahun 2010.*", if the dictionary has the translation pair "(*peratus*, *persen*)", this hypothesis producer will produce a new hypothesis "*KDNK Malaysia dijangka cecah 8.1 persen pada tahun 2010.*"

- Phrase-level mapping: This hypothesis producer utilizes the pivoted phrase table described in Section 5.2.1.2. For example, if the pivoted phrase table contains the phrase pair "(*dijangka cecah*, *akan mencapai*)", given the hypothesis "*KDNK Malaysia dijangka cecah 8.1 peratus pada tahun 2010.*", a new hypothesis "*KDNK Malaysia akan mencapai 8.1 peratus pada tahun 2010.*" will be produced by this hypothesis producer.

- Cross-lingual morphological mapping: This hypothesis producer uses the cross-lingual morphological variant dictionary from a Malay word to its Indonesian morphological variants described in Section 5.2.1.3. For example, given the hypothesis "*dan untuk meringkaskan pengalamannya ?*", if the dictionary has the morphological variant pair "(*meringkaskan*, *meringkas*)", this hypothesis producer will produce a new hypothesis "*dan untuk meringkas pengalamannya ?*"

The hypothesis producers presented above are all based on statistical methods. We may also design some rule-based hypothesis producers to adapt Malay to Indone-

sian. As an example, the number format of Malay is different from that of Indonesian. Malay numbers are written in accordance with British convention, i.e., "." denotes the decimal point and "," denotes digit grouping. Indonesian numbers are the opposite. This difference allows us to build a rule-based hypothesis producer to convert Malay numbers to Indonesian ones, e.g., converting "*KDNK Malaysia dijangka cecah 8.1 peratus pada tahun 2010.*" to "*KDNK Malaysia dijangka cecah 8,1 peratus pada tahun 2010.*" However, these rule-based hypothesis producers are language-specific. Since we want to make our source language adaptation decoder language-independent, only statistical hypothesis producers are designed for this work. As a result, our decoder can be applied to different closely related language pairs, which we will show in Section 5.5.4.

#### 5.2.1.5 Feature Functions

In our source language adaptation decoder, the feature functions can be categorized into two kinds. The first kind is the count feature functions described in Section 3.4. The second kind includes some general feature functions:

- An Indonesian language model;
- A word penalty, i.e., the number of tokens in the hypothesis; (As the language model prefers shorter hypotheses, the word penalty is used to guard against hypotheses which are too short.)
- A Malay word penalty, i.e., the count of Malay words identified by bigram counts from the Indonesian language model; a word $w$ in a hypothesis $\ldots w_{-1}ww_1\ldots$ is considered a Malay word if both bigrams $w_{-1}w$ and $ww_1$ have no occurrences in the Indonesian language model;
- Word-level mapping hypothesis producer: We have a feature function which is the summation of the logarithms of all the conditional probabilities (see Equation 5.1) used so far;

- Phrase-level mapping hypothesis producer: We have four feature functions, each of which is the summation of the logarithms of one of the four scores in the pivoted phrase table, i.e., forward/reverse phrase translation probability and forward/reverse lexical weighting (see Section 5.2.1.2);

- Cross-lingual morphological mapping hypothesis producer: We have a feature function which is the summation of the logarithms of all the morphological variant mapping scores (see Equation 5.6) used so far.

As discussed in Section 3.4, the feature functions are combined in a linear model, and the weights of the feature functions are tuned using pairwise ranking optimization (Hopkins and May, 2011) on the development set.

### 5.2.2 Word-Level Paraphrasing Approach

Given a Malay sentence, we generate a confusion network containing multiple Indonesian word-level paraphrase options for each Malay word. Each such Indonesian option is associated with a corresponding weight in the network, which is defined as the probability of this option being a translation of the original Malay word (see Equation 5.1). We decode this confusion network using a large Indonesian language model, thus generating a ranked list of $n$ corresponding adapted "Indonesian" sentences. Below we explain how we build, decode, and improve the confusion network.

#### 5.2.2.1 Confusion Network Construction

Given a Malay sentence, we construct an Indonesian confusion network, where each Malay word is augmented with a set of network transitions, which are the possible Indonesian word translations. The weight of each transition is the conditional Indonesian-Malay translation probability as calculated by Equation 5.1. The original Malay word is assigned a weight of 1.

Note that we paraphrase *each* word in the input Malay sentence as opposed to only those Malay words that we believe do not to exist in Indonesian, e.g., because they do not appear in our Indonesian monolingual text. This is necessary because of the large number of false friends and partial cognates between Malay and Indonesian (see Section 5.1).

Finally, we decode the confusion network for a Malay sentence using a large Indonesian language model, and we extract an $n$-best list[3] containing the $n$-best adapted "Indonesian" sentences for the Malay sentence. For example, Table 5.1 shows the 10-best adapted "Indonesian" sentences that we generated for the confusion network in Figure 5.2. According to a native Indonesian speaker, options 1 and 3 in Table 5.1 are perfect adaptations, options 2 and 5 have a wrong word order, and the rest are grammatical though not perfect.

| **Rank** | **"Indonesian" Sentence** | | | |
|---|---|---|---|---|
| 1 | pdb | malaysia | akan | mencapai 8 persen pada tahun 2010 . |
| 2 | pdb | malaysia | untuk | mencapai 8 persen pada tahun 2010 . |
| 3 | pdb | malaysia | diperkirakan | mencapai 8 persen pada tahun 2010 . |
| 4 | maka | malaysia | akan | mencapai 8 persen pada tahun 2010 . |
| 5 | maka | malaysia | untuk | mencapai 8 persen pada tahun 2010 . |
| 6 | pdb | malaysia | dapat | mencapai 8 persen pada tahun 2010 . |
| 7 | maka | malaysia | diperkirakan | mencapai 8 persen pada tahun 2010 . |
| 8 | sebesar | malaysia | akan | mencapai 8 persen pada tahun 2010 . |
| 9 | pdb | malaysia | diharapkan | mencapai 8 persen pada tahun 2010 . |
| 10 | pdb | malaysia | ini | mencapai 8 persen pada tahun 2010 . |

Table 5.1: The 10-best "Indonesian" sentences extracted from the confusion network in Figure 5.2.

[3]For balance, in case of less than $n$ adaptations for a Malay sentence, we randomly repeat some of the available ones.

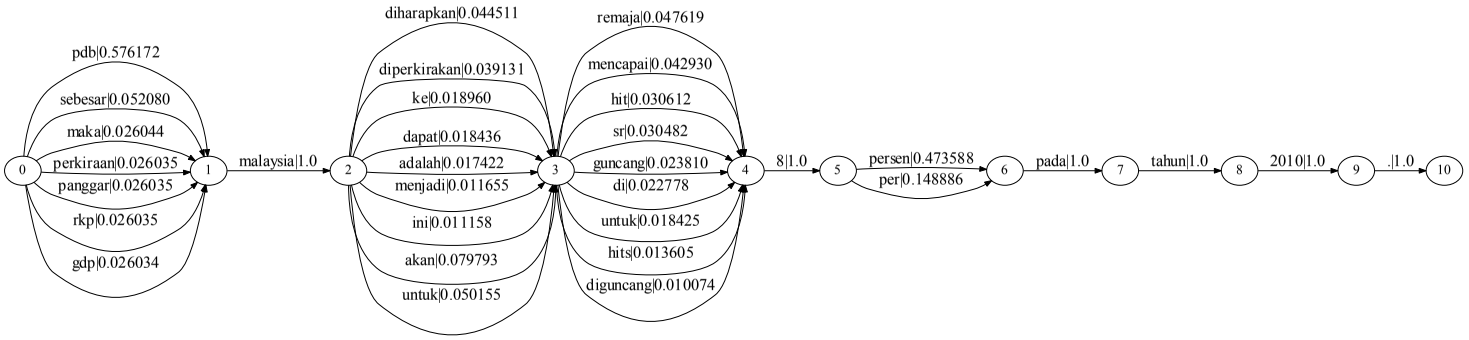


Figure 5.2: Indonesian confusion network for the Malay sentence "*KDNK Malaysia dijangka cecah 8 peratus pada tahun 2010.*" Arcs with scores below 0.01 are omitted, and words that exist in Indonesian are not paraphrased (for better readability).

#### 5.2.2.2 Further Refinements

Since the Indonesian-Malay paraphrases are obtained from pivoting over English, many of the paraphrases are bad: some have very low probabilities, while others involve rare words for which the probability estimates are unreliable.

Moreover, the Indonesian paraphrases that we propose for a Malay word are inherently restricted to the small Indonesian vocabulary of the Indonesian-English bi-text. Below we describe how we address these issues:

- **Score-based filtering:** We filter out translation pairs whose probabilities (Equation 5.1) are lower than some threshold which is tuned on the development set, e.g., 0.01 or 0.001.

- **Improved estimations for $\Pr(i|e)$:** We concatenate $k$ copies of the small Indonesian-English bi-text and one copy of the larger Malay-English bi-text, where the value of $k$ is selected so that we have roughly the same number of Indonesian and Malay sentences. Then, we generate word-level alignments for the resulting bi-text. Finally, we truncate these alignments keeping them for one copy of the original Indonesian-English bi-text only. Thus, we end up with improved word alignments for the Indonesian-English bi-text, and ultimately with better estimations for Equation 5.1. Since Malay and Indonesian share many cognates, this improves word alignments for Indonesian words that occur rarely in the small Indonesian-English bi-text but are relatively frequent in the larger Malay-English one; it also helps for some frequent words.

- **Cross-lingual morphological variants:** We increase the Indonesian options for a Malay word using morphology. Since the set of Indonesian options for a Malay word in pivoting is restricted to the Indonesian vocabulary of the small Indonesian-English bi-text, this is a severe limitation of pivoting. Thus, we use the same method of Section 5.2.1.3 to generate the Indonesian morphological variants for

each Malay word, and then add the morphological variants to the confusion network as additional options for the Malay word. In the confusion network, the weight of the original Malay word is set to 1, while the weight of a morphological variant is the morphological variant mapping score between the variant and the Malay word based on Equation 5.6, multiplied by the highest probability for all pivoting variants for the Malay word, i.e., we trust pivoting more.

### 5.2.3 Phrase-Level Paraphrasing Approach

*Word-level* paraphrasing ignores context when generating Indonesian variants, relying on the Indonesian language model to make the right contextual choice. We also try to model context more directly by generating adaptation options at the *phrase level*.

We use the same method described in Section 5.2.1.2 to induce the phrase-level Indonesian translation options for Malay phrases, i.e., using the pivoted phrase table. The phrase-based SMT decoder of Moses (Koehn et al., 2007) is used to "translate" the Malay side of the Malay-English bi-text to get closer to Indonesian without word reordering. The decoder is tuned on a development set using minimum error rate training (MERT) (Och, 2003).

#### 5.2.3.1 Cross-Lingual Morphological Variants

While phrase-level paraphrasing models context better, it remains limited in the size of its Indonesian vocabulary by the small Indonesian-English bi-text, just like what word-level paraphrasing was. We address this by transforming the Indonesian sentences in the *development* and the *test* Indonesian-English bi-texts into confusion networks (Dyer, 2007; Du et al., 2010), where we add Malay morphological variants for the Indonesian words, weighting them based on Equation 5.6. Note that we do not alter the training bi-text.

### 5.2.4 Combining Bi-Texts

We combine the Indonesian-English and the synthetic "Indonesian"-English bi-texts as follows:

- **Simple concatenation:** Assuming the two bi-texts are of comparable quality, we simply train an SMT system on their concatenation.

- **Balanced concatenation with repetitions:** However, the two bi-texts are not directly comparable. For one thing, the adapted "Indonesian"-English bi-text is obtained from $n$-best lists, i.e., it has exactly $n$ very similar variants for each Malay sentence. Moreover, the original Malay-English bi-text is much larger in size than the Indonesian-English one and now it has further expanded $n$ times to become "Indonesian"-English, which means that it will dominate the concatenation due to its size. To counter-balance this, we repeat the smaller Indonesian-English bi-text enough times to make its number of sentences roughly the same as the "Indonesian"-English bi-text; then we concatenate them and train an SMT system on the resulting bi-text.

- **Sophisticated phrase table combination:** Finally, we experiment with a method for combining phrase tables proposed in (Nakov and Ng, 2009; Nakov and Ng, 2012). The first phrase table is extracted from word alignments for the balanced concatenation with repetitions, which are then truncated so that they are kept for only one copy of the Indonesian-English bi-text. The second table is built from the simple concatenation. The two tables are then merged as follows: all phrase pairs from the first one are retained, and to them are added those phrase pairs from the second one that are not present in the first one. Each phrase pair retains its original scores, which are further augmented with 1-3 extra feature scores indicating its origin: the first/second/third feature is 1 if the pair came from the first/second/both table(s), and 0 otherwise. We experiment using all three, the first two, or the first

feature only; we also try setting the features to 0.5 instead of 0. This makes the following six combinations (0, 00, 000, .5, .5.5, .5.5.5); on testing, we use the one that achieves the highest BLEU score on the development set.

Other possibilities for combining the phrase tables include using alternative decoding paths (Birch et al., 2007), simple linear interpolation, and direct phrase table merging with extra features (Callison-Burch et al., 2006). However, they were previously found to be inferior to the last two approaches above (Nakov and Ng, 2009; Nakov and Ng, 2012).

## 5.3 Experiments

With a small Indonesian-English bi-text and a larger Malay-English bi-text, we use three approaches for source language adaptation to adapt the Malay side of the Malay-English bi-text to look like Indonesian, thus obtaining a synthetic "Indonesian"-English bi-text. With the synthetic bi-text, we run two kinds of experiments:

- *isolated*, where we train an SMT system on the synthetic "Indonesian"-English bi-text only;
- *combined*, where we combine the synthetic bi-text with the original Indonesian-English bi-text.

All the experiments are tuned on the same Indonesian-English development set and tested on the same Indonesian-English test set.

### 5.3.1 Datasets

In our experiments, we use the following datasets, normally required for Indonesian-English SMT:

- **Indonesian-English training bi-text (*IN2EN*):** 28,383 sentence pairs; 915,192 English tokens; 796,787 Indonesian tokens;

- **Indonesian-English dev bi-text (*IN2EN-dev*):** 2,000 sentence pairs; 37,101 English tokens; 35,509 Indonesian tokens;

- **Indonesian-English test bi-text (*IN2EN-test*):** 2,018 sentence pairs; 36,584 English tokens; 35,708 Indonesian tokens;

- **Monolingual English text (*EN-LM*):** 174,443 sentences; 5,071,988 English tokens.

We also use a Malay-English set (to be adapted into "Indonesian"-English), and monolingual Indonesian text (for decoding the confusion network):

- **Malay-English training bi-text (*ML2EN*):** 290,000 sentence pairs; 8,638,780 English tokens; 8,061,729 Malay tokens;

- **Monolingual Indonesian text (*IN-LM*):** 1,132,082 sentences; 20,452,064 Indonesian tokens.

We use two bi-texts (*IN2EN* and *ML2EN*) to induce word-level and phrase-level paraphrases as described in Section 5.2.1.1 and 5.2.1.2, respectively. In Section 5.2.1.3, to induce the Indonesian morphological variants for a Malay word, we use a large monolingual Indonesian corpus which is *IN-LM*.

All the above datasets were built from texts which were crawled from the Internet.

Another Malay-Indonesian development set is needed to tune our source language adaptation decoder of Section 5.2.1 and the phrase-based SMT decoder in the phrase-level paraphrasing approach of Section 5.2.3. Since we have no such bi-text, we create a synthetic bi-text by translating the English side of the *IN2EN-dev* into Malay using

Google Translate[4], and then pair the translated Malay texts with the Indonesian side of *IN2EN-dev*:

- **Synthetic Malay-Indonesian dev bi-text (*ML2IN-dev*):** 2,000 sentence pairs; 34,261 Malay tokens; 35,509 Indonesian tokens.

### 5.3.2 Baseline Systems

We build five baseline systems – two using a single bi-text, *ML2EN* or *IN2EN*, and three combining *ML2EN* and *IN2EN*, using simple concatenation, balanced concatenation, and sophisticated phrase table combination. The last combination is a very strong baseline and the most relevant one that we need to improve upon.

In the experiments, we build each SMT system as follows. Given a training bi-text, its separate directed word alignments are built using IBM model 4 (Brown et al., 1993) for both directions of the bi-text. The word alignments of the two directions are then combined using the intersect+grow heuristic (Och and Ney, 2003). Based on the combined word alignments, phrase translation pairs of length up to seven are extracted using the alignment template approach (Och and Ney, 2004). A phrase table containing the phrase pairs is generated. In the phrase table, each phrase pair has five features (Koehn, 2013): forward and reverse translation probabilities, forward and reverse lexical weighting probabilities, and a phrase penalty. A log-linear model is adopted to combine the features: (1) the five features in the phrase table; (2) a language model score; (3) a word penalty, i.e., the number of words in the output translation; (4) distance-based reordering cost. The weights of the log-linear model are tuned to optimize the BLEU score (Papineni et al., 2002) on the development set *IN2EN-dev* using MERT (Och, 2003). The phrase-based SMT decoder of Moses is used to perform translation with the log-linear model. A 5-gram language model is trained with the SRILM toolkit (Stolcke, 2002) and

[4] `http://translate.google.com/`

modified Kneser-Ney smoothing (Kneser and Ney, 1995). All the experiments are tested on the same test set *IN2EN-test*.

### 5.3.3 Isolated Experiments

The isolated experiments only use the adapted "Indonesian"-English bi-text as the training bi-text, which allows for a direct comparison to using *ML2EN* or *IN2EN* only.

#### 5.3.3.1 Word-Level Paraphrasing

In our word-level paraphrasing experiments, we adapt Malay to Indonesian using three kinds of confusion networks (CN) (see Section 5.2.2.2 for details):

- ***CN:word*** – using word-level pivoting only;
- ***CN:word′*** – using word-level pivoting, with probabilities from word alignments for *IN2EN* that were improved using *ML2EN*;
- ***CN:word′+morph*** – *CN:word′* augmented with cross-lingual morphological variants.

There are two parameter values to be tuned on *IN2EN-dev* for the above confusion networks: (1) the minimum pivoting probability threshold for the Malay-Indonesian word-level paraphrases, and (2) the number of $n$-best Indonesian-adapted sentences that are to be generated for each input Malay sentence. We try $\{0.001, 0.005, 0.01, 0.05\}$ for the threshold and $\{1, 5, 10\}$ for $n$.

#### 5.3.3.2 Phrase-Level Paraphrasing

In our phrase-level paraphrasing experiments, we use pivoted phrase tables (PPT) with the following features for each phrase table entry (in addition to the phrase penalty; see Section 5.2.3 for more details):

- ***PPT:phrase1*** – only using the forward phrase translation probability;
- ***PPT:phrase4*** – using all four scores;
- ***PPT:phrase4::CN:morph*** – *PPT:phrase4* but used with a cross-lingual morphological confusion network for the dev/test Indonesian sentences.

Here we tune one parameter only: the number of $n$-best Indonesian-adapted sentences to be generated for each input Malay sentence; we try $\{1, 5, 10\}$. The phrase-level paraphrasing systems are tuned on the development set *ML2IN-dev*.

#### 5.3.3.3 Source Language Adaptation Decoder

Using our source language adaptation decoder (DD) based on the proposed text rewriting decoder, we conduct four experiments with different hypothesis producers (see Section 5.2.1.4 for more details):

- ***DD:word′*** – only using one hypothesis producer, word-level mapping, whose dictionary contains word-level pivoting with probabilities from word alignments for *IN2EN* that were improved using *ML2EN*;
- ***DD:word′+morph*** – *DD:word′* added one more hypothesis producer, cross-lingual morphological mapping, which utilizes a dictionary of cross-lingual morphological variants;
- ***DD:phrase4*** – only using one phrase-level mapping hypothesis producer which utilizes the same pivoted phrase table as *PPT:phrase4*;
- ***DD:phrase4+morph*** – *DD:phrase4* but used with another cross-lingual morphological mapping hypothesis producer as *DD:word′+morph*.

The source language adaptation decoders used to generate the adapted "Indonesian"-English training bi-text are tuned on the development set *ML2IN-dev*. There are two

parameter values to be tuned on *IN2EN-dev* for the first two experiments: (1) the minimum pivoting probability threshold for the Malay-Indonesian word-level paraphrases, and (2) the number of $n$-best Indonesian-adapted sentences that are to be generated for each input Malay sentence. For the last two experiments, we only need to tune (2). We try $\{0.001, 0.005, 0.01, 0.05\}$ for (1) and $\{1, 5, 10\}$ for (2).

We have also tried to use the word-level mapping and phrase-level mapping hypothesis producers in a decoder, which performs about the same as the phrase-level mapping hypothesis producer alone. The reason may be due to the fact that both mappings are extracted from the word alignments of the same Malay-English and Indonesian-English bi-texts by pivoting. The phrase-level mapping should contain more knowledge than the word-level mapping, i.e., the context knowledge. As a result, when using them together in one decoder, we only get similar results as using phrase-level mapping alone.

### 5.3.4 Combined Experiments

These experiments assess the impact of our adaptation approach when combined with the original Indonesian-English bi-text *IN2EN* as opposed to combining *ML2EN* with *IN2EN* (as was in the last three baselines). We experiment with the same three combinations: simple concatenation, balanced concatenation, and sophisticated phrase table combination. We tune the parameters as before; for the last combination, we further tune the six extra feature combinations (see Section 5.2.4 for details).

## 5.4 Results and Discussion

For all tables, statistically significant improvements ($p < 0.01$), according to Collins et al. (2005)'s sign test, over the baseline are in **bold**; in case of two baselines, <u>underline</u> is used for the second baseline.

### 5.4.1 Baseline Experiments

The results for the baseline systems are shown in Table 5.2. We can see that training on *ML2EN* instead of *IN2EN* yields over 4 points absolute drop in BLEU (Papineni et al., 2002) score, even though *ML2EN* is about 10 times larger than *IN2EN* and both bi-texts are from the same domain. This confirms the existence of important differences between Malay and Indonesian. While simple concatenation does not help, balanced concatenation with repetitions improves by 1.12% BLEU points over *IN2EN*, which shows the importance of giving *IN2EN* a proper weight in the combined bi-text. This is further reconfirmed by the sophisticated phrase table combination, which yields an additional absolute gain of 0.31% BLEU points.

| **System** | **BLEU (%)** |
|---|---|
| *ML2EN* | 14.50 |
| *IN2EN* | 18.67 |
| Simple concatenation | **18.49** |
| Balanced concatenation | **19.79** |
| Sophisticated phrase table combination | **20.10**$_{(.5.5)}$ |

Table 5.2: **The five baselines.** The subscript indicates the parameters found on *IN2EN-dev* and used for *IN2EN-test*. The scores that are statistically significantly better than *ML2EN* and *IN2EN* ($p < 0.01$, Collins' sign test) are shown in **bold** and are underlined, respectively.

### 5.4.2 Isolated Experiments

Table 5.3 shows the results for the isolated experiments. We can see that word-level paraphrasing improves by up to 5.56% and 1.39% BLEU scores over the two baselines (both statistically significant). Compared to *ML2EN*, *CN:word* yields an absolute improvement of 4.41% BLEU scores, *CN:word′* adds another 0.59%, and *CN:word′+morph* adds 0.56% more. The scores for TER (v. 0.7.25) (Snover et al., 2006) and METEOR (v. 1.3)

(Banerjee and Lavie, 2005) are on par with those for BLEU (NIST v. 13).

| | System | $n$-gram precision 1-gr. | 2-gr. | 3-gr. | 4-gr. | BLEU (%) | TER | METEOR |
|---|---|---|---|---|---|---|---|---|
| | *ML2EN* (baseline) | 48.34 | 19.22 | 9.54 | 4.98 | 14.50 | 67.14 | 43.28 |
| | *IN2EN* (baseline) | 55.04 | 23.90 | 12.87 | 7.18 | 18.67 | 61.99 | 54.34 |
| | *CN:word* | 54.50 | 24.41 | 13.09 | 7.35 | **18.91**$^{(+4.41,+0.24)}_{(0.005,10best)}$ | 61.94 | 51.07 |
| | *CN:word′* | 55.05 | 25.09 | 13.60 | 7.69 | **19.50**$^{(+5.00,+0.83)}_{(0.001,10best)}$ | 61.25 | 51.97 |
| (i) | *CN:word′+morph* | 55.97 | 25.73 | 14.06 | 7.99 | **20.06**$^{(+5.56,+1.39)}_{(0.005,10best)}$ | 60.31 | 55.65 |
| | *PPT:phrase1* | 55.11 | 25.04 | 13.66 | 7.80 | **19.58**$^{(+5.08,+0.91)}_{(10best)}$ | 60.92 | 51.93 |
| | *PPT:phrase4* | 56.64 | 26.20 | 14.53 | 8.40 | **20.63**$^{(+6.13,+1.96)}_{(10best)}$ | 59.33 | 54.23 |
| (ii) | *PPT:phrase4::CN:morph* | 56.91 | 26.53 | 14.76 | 8.55 | **20.89**$^{(+6.39,+2.22)}_{(10best)}$ | 59.30 | 57.19 |
| | *DD:word′* | 56.57 | 26.15 | 14.39 | 8.18 | **20.39**$^{(+5.89,+1.72)}_{(0.01,10best)}$ | 59.33 | 56.66 |
| | *DD:word′+morph* | 56.74 | 26.22 | 14.41 | 8.18 | **20.46**$^{(+5.96,+1.79)}_{(0.005,10best)}$ | 59.50 | 56.89 |
| | *DD:phrase4* | 57.14 | 26.49 | 14.72 | 8.49 | **20.85**$^{(+6.35,+2.18)}_{(10best)}$ | 58.79 | 57.33 |
| (iii) | *DD:phrase4+morph* | 57.35 | 26.71 | 14.92 | 8.63 | **21.07**$^{(+6.57,+2.40)}_{(10best)}$ | 58.55 | 57.53 |
| | System combination: (i)+(ii)+(iii) | 58.46 | 27.64 | 15.46 | 9.07 | **21.76**$^{(+7.26,+3.09)}$ | 57.26 | 58.04 |

Table 5.3: **Isolated experiments.** The subscript indicates the parameters found on *IN2EN-dev* and used for *IN2EN-test*. The superscript shows the absolute test improvement over the *ML2EN* and the *IN2EN* baselines. The scores that are statistically significantly better than *ML2EN* and *IN2EN* ($p < 0.01$, Collins' sign test) are shown in **bold** and are underlined, respectively. The last line shows system combination results using MEMT.

Table 5.3 further shows that the optimal parameters for the word-level systems (*CN:**) involve a very low probability cutoff, and a high number of $n$-best sentences. This shows they are robust to noise, probably because bad source-side phrases are unlikely to match the test-time input. Note also the effect of repetitions: good word choices are shared by many $n$-best sentences, and thus have higher probability.

The gap between *ML2EN* and *IN2EN* for unigram precision could be explained by vocabulary differences between Malay and Indonesian. Compared to *IN2EN*, all *CN:** models have higher 2/3/4-gram precision. However, *CN:word* has lower unigram precision, which could be due to bad word alignments, as the results for *CN:word′* show.

When morphological variants are further added, the unigram precision improves by almost 1% absolute over *CN:word′*. This shows the importance of morphology for

overcoming the limitations of the small Indonesian vocabulary of the *IN2EN* bi-text.

The second part of Table 5.3 shows that phrase-level paraphrasing approach (*PPT:**) performs a bit better. This confirms the importance of modeling context for closely-related languages like Malay and Indonesian, which are rich in false friends and partial cognates. We further see that using more scores in the pivoted phrase table is better. Extending the Indonesian vocabulary with cross-lingual morphological variants is still helpful, though not as much as at the word-level.

The third part of Table 5.3 shows that text rewriting decoder approach (*DD:**) performs better than the first two approaches. The decoder approach further increases the improvements up to 6.57% and 2.40% BLEU scores over the two baselines (statistically significant).

Finally, the combination of the output of the best *PPT*, *CN* and *DD* systems using MEMT (Heafield and Lavie, 2010) improves even further, which shows that the three approaches are complementary. The best BLEU score for our isolated experiments is 21.76%, which is already better than all five baselines in Table 5.2, including the three bi-text combination baselines, which only achieve up to 20.10%.

### 5.4.3 Combined Experiments

Table 5.4 shows the performance of the three bi-text combination strategies (see Section 5.2.4 for details) when applied to combine *IN2EN* with (1) the original *ML2EN* and (2) various adapted versions of it.

We can see that for the word-level paraphrasing experiments (*CN:**), all combinations except *CN:word* perform significantly better than their corresponding baselines, but the improvements are most sizeable for simple concatenation. Note that while there is a difference of 0.31% BLEU scores between the balanced concatenation and the sophisticated combination for the original *ML2EN*, they differ little for the adapted versions. This is probably due to the sophisticated combination assuming that the second bi-text is

| | | Combining *IN2EN* with an adapted version of *ML2EN* | | |
|---|---|---|---|---|
| | **Combination with** | **Simple Concatenation** | **Balanced Concatenation** | **Sophisticated Combination** |
| (i) | + *ML2EN* (unadapted; baseline) | 18.49 | 19.79 | $20.10_{(.5.5)}$ |
| | + *CN:word* | $\mathbf{19.99}^{(+1.50)}_{(0.001,1best)}$ | $20.16^{(+0.37)}_{(0.001,10best)}$ | $20.32^{(+0.22)}_{(0.01,10best,.5.5)}$ |
| | + *CN:word′* | $\mathbf{20.03}^{(+1.54)}_{(0.05,1best)}$ | $\mathbf{20.80}^{(+1.01)}_{(0.05,10best)}$ | $\mathbf{20.55}^{(+0.45)}_{(0.05,10best,.5.5)}$ |
| (ii) | + *CN:word′+morph* | $\mathbf{20.60}^{(+2.11)}_{(0.01,10best)}$ | $\mathbf{21.15}^{(+1.36)}_{(0.01,10best)}$ | $\mathbf{21.05}^{(+0.95)}_{(0.01,5best,00)}$ |
| | + *PPT:phrase1* | $\mathbf{20.61}^{(+2.12)}_{(1best)}$ | $\mathbf{20.71}^{(+0.92)}_{(10best)}$ | $20.32^{(+0.22)}_{(1best,000)}$ |
| | + *PPT:phrase4* | $\mathbf{20.75}^{(+2.26)}_{(1best)}$ | $\mathbf{21.08}^{(+1.29)}_{(5best)}$ | $\mathbf{20.76}^{(+0.66)}_{(10best,.5.5.5)}$ |
| (iii) | + *PPT:phrase4::CN:morph* | $\mathbf{21.01}^{(+2.52)}_{(1best)}$ | $\mathbf{21.31}^{(+1.52)}_{(5best)}$ | $\mathbf{20.98}^{(+0.88)}_{(10best,.5)}$ |
| | + *DD:word′* | $\mathbf{20.67}^{(+2.18)}_{(0.01,5best)}$ | $\mathbf{20.75}^{(+0.96)}_{(0.001,10best)}$ | $\mathbf{21.16}^{(+1.06)}_{(0.01,10best,.5.5.5)}$ |
| | + *DD:word′+morph* | $\mathbf{20.78}^{(+2.29)}_{(0.01,1best)}$ | $\mathbf{21.25}^{(+1.46)}_{(0.01,5best)}$ | $\mathbf{21.41}^{(+1.31)}_{(0.005,10best,.5.5)}$ |
| | + *DD:phrase4* | $\mathbf{20.91}^{(+2.42)}_{(5best)}$ | $\mathbf{21.20}^{(+1.41)}_{(5best)}$ | $\mathbf{20.99}^{(+0.89)}_{(10best,111)}$ |
| (iv) | + *DD:phrase4+morph* | $\mathbf{21.33}^{(+2.84)}_{(5best)}$ | $\mathbf{21.42}^{(+1.63)}_{(5best)}$ | $\mathbf{21.08}^{(+0.98)}_{(10best,111)}$ |
| | System combination: (i)+(ii)+(iii)+(iv) | $\mathbf{21.74}^{(+3.25)}$ | $\mathbf{21.81}^{(+2.02)}$ | $\mathbf{22.03}^{(+1.93)}$ |

Table 5.4: **Combined experiments: BLEU (%).** The subscript indicates the parameters found on *IN2EN-dev* and used for *IN2EN-test*. The absolute test improvement over the corresponding baseline (on top of each column) is in superscript. The scores that are statistically significantly better than *ML2EN* ($p < 0.01$, Collins' sign test) are shown in **bold**. The last line shows system combination results using MEMT.

worse than the first one, which is not really the case for the adapted versions: as Table 5.3 shows, they all outperform *IN2EN*.

Overall, phrase-level paraphrasing (*PPT:**) performs a bit better than word-level paraphrasing, and the text rewriting decoder approach (*DD:**) further increases the improvements. At last, system combination with MEMT improves even further. This is consistent with the isolated experiments.

### 5.4.4 Summary of Experiments

To summarize all the experiments, Table 5.5 shows the overall improvements that we have obtained in our experiments over the baselines. The first two experiments are the best isolated baseline (*IN2EN* in Table 5.2) and the best combined baseline (*Sophisticated phrase table combination* in Table 5.2), respectively. The last two experiments are the best systems that we have built: the best isolated system (the last row of Table 5.3) and

the best combined system (*Sophisticated Combination* in the last row of Table 5.4). As we can see that both of the last two systems perform statistically significantly better than the two baselines, which shows the potential of our source language adaptation idea.

| System | BLEU (%) |
|---|---|
| Best isolated baseline | 18.67 |
| Best combined baseline | 20.10 |
| Best isolated system | **21.76** |
| Best combined system | **22.03** |

Table 5.5: **Overall improvements.** The scores that are statistically significantly better than the best isolated baseline and the best combined baseline ($p < 0.01$, Collins' sign test) are shown in **bold** and are underlined, respectively.

## 5.5 Further Analysis

Below we perform more analysis and experiments.

### 5.5.1 Paraphrasing only Non-Indonesian Words

In *CN:** above, we paraphrased *each* word in the Malay input, because of false friends like *polisi* and partial cognates like *nanti*. This risks proposing worse alternatives, e.g., changing *beliau* ("he", respectful) to *ia* ("he", casual), which the weights on the confusion network edges and the language model would not always handle. Thus, we tried paraphrasing non-Indonesian words only, i.e., those not in *IN-LM*. Since *IN-LM* occasionally contains some Malay-specific words, we also tried paraphrasing words that occur at most $t$ times in *IN-LM*. Table 5.6 shows that this hurts by up to 1% BLEU scores for $t = 0; 10$, and a bit less for $t = 20; 40$.

| System | BLEU (%) |
|---|---|
| *CN:word*, $t = 0$ | $17.88_{(0.01,5best)}$ |
| *CN:word*, $t = 10$ | $17.88_{(0.05,10best)}$ |
| *CN:word*, $t = 20$ | $18.14_{(0.01,5best)}$ |
| *CN:word*, $t = 40$ | $18.34_{(0.01,5best)}$ |
| *CN:word* (i.e., paraphrase all) | $18.91_{(0.005,10best)}$ |

Table 5.6: **Paraphrasing non-Indonesian words only:** those appearing at most $t$ times in *IN-LM*. The subscript indicates the parameters found on *IN2EN-dev* and used for *IN2EN-test*.

| System | Better | Equal | Worse |
|---|---|---|---|
| *CN:word*, $t = 0_{(Rank1)}$ | 53% | 31% | 16% |
| *CN:word′+morph*$_{(Rank1)}$ | 38% | 8% | 54% |
| *CN:word′+morph*$_{(Rank2)}$ | 41% | 9% | 50% |
| *CN:word′+morph*$_{(Rank3)}$ | 32% | 11% | 57% |
| *CN:word′+morph*$_{(Ranks:1-3)}$ | 45% | 12% | 43% |

Table 5.7: **Human judgments: Malay versus adapted "Indonesian".** A subscript shows the ranking of the sentences, and the parameter values are those from Tables 5.3 and 5.6.

### 5.5.2 Manual Evaluation

We asked a native Indonesian speaker who does not speak Malay to judge whether our "Indonesian" adaptations are more understandable to him than the original Malay input for 100 random sentences. We used two extremes: the conservative *CN:word,t*=0 vs. *CN:word′+morph*. Since the latter is noisy, the top 3 choices were judged for it. Table 5.7 shows that *CN:word,t*=0 is better/equal to the original 53%/31% of the time. Thus, it is a very good step in the direction of turning Malay into Indonesian. In contrast, *CN:word′+morph* is typically worse than the original; moreover, those at rank 2 are a bit better than those at rank 1; even compared to the best in top 3, the better:worse ratio is 45%:43%. Still, this latter model works better, which means that phrase-based SMT systems are robust to noise and prefer more variety. Note also that the judgments were at

the sentence level, while phrases are sub-sentential, i.e., there can be many good phrases in a "bad" sentence.

### 5.5.3 Reversed Adaptation

In all experiments above, we were adapting the Malay sentences to look like Indonesian. Here we try to reverse the direction of adaptation, i.e., to adapt Indonesian to Malay. We have tried three approaches to this idea:

- ***lattice*:** Build an Indonesian-to-Malay confusion network for each dev/test Indonesian sentence using a pivoted word-level Indonesian-Malay dictionary which is induced by reversing the direction of the method in Section 5.2.1.1. Use the confusion networks directly as input to a Malay-English SMT system trained on the *ML2EN* dataset, i.e., tune a log-linear model using confusion networks for the source side of the *IN2EN-dev* dataset, and then evaluate the tuned system using confusion networks for the source side of the *IN2EN-test* dataset.

- ***1-best*:** Based on the Indonesian-to-Malay confusion networks generated in *lattice*, decode the networks for the source side of the *IN2EN-dev* and the *IN2EN-test* with a Malay language model to get the 1-best outputs. Then pair each 1-best output with the corresponding English sentence. Finally, get an adapted "Malay"-English development set and an adapted "Malay"-English test set, and use them to tune and evaluate the *ML2EN* SMT system.

- ***decoder*:** Use our text rewriting decoder to adapt the source side of the *IN2EN-dev* and the *IN2EN-test* to get 1-best outputs. Since the first two approaches only take the advantage from the pivoted word-level Indonesian-Malay dictionary, we only use a word-level mapping hypothesis producer in the text rewriting decoder which uses the same dictionary as the first two approaches. Then pair each 1-best output with the corresponding English sentence, obtaining an adapted "Malay"-English

development set and an adapted "Malay"-English test set. Use them to tune and evaluate the *ML2EN* SMT system.

Table 5.8 shows that all of the three approaches perform worse than *CN:word*. We believe this is because *lattice* encodes many options, but does not use a Malay language model, while *1-best* uses a Malay language model, but has to commit to 1-best. *decoder* uses a Malay language model, but is also limited to 1-best. In contrast, *CN:word* uses both $n$-best outputs and an Indonesian language model. Designing a similar setup for reversed adaptation is a research direction that we would like to pursue in future work.

| **System** | **BLEU (%)** |
|---|---|
| *CN:word* (Malay→Indonesian) | $18.91_{(0.005,10best)}$ |
| *CN:word* (Indonesian→Malay) – lattice | $17.22_{(0.05)}$ |
| *CN:word* (Indonesian→Malay) – 1-best | $17.77_{(0.001)}$ |
| *DD:word* (Indonesian→Malay) – decoder | $18.29_{(0.001)}$ |

Table 5.8: **Reversed adaptation: Indonesian to Malay.** The subscript indicates the parameters found on *IN2EN-dev* and used for *IN2EN-test*.

### 5.5.4 Adapting Bulgarian to Macedonian to Help Macedonian-English Translation

In order to show the applicability of our approaches, we experimented with another pair of closely-related languages, Macedonian (*MK*) and Bulgarian (*BG*), using data from a different, non-newswire domain: the OPUS corpus of movie subtitles (Tiedemann, 2009). We used datasets of sizes that are comparable to those in the previous Malay-Indonesian experiments: 160K *MK2EN* and 1.5M *BG2EN* sentence pairs (1.2M and 11.5M English words). Since the sentences of movie subtitles were short, we used 10K *MK2EN* sentence pairs for tuning and testing (77K and 72K English words). For language modeling, we used 9.2M Macedonian and 433M English words.

| System | BLEU (%) | TER | METEOR |
|---|---|---|---|
| *BG2EN* (baseline) | 24.57 | 57.64 | 41.60 |
| *MK2EN* (baseline) | 26.46 | 54.55 | 46.15 |
| **Balanced concatenation of *MK2EN* with an adapted *BG2EN*** | | | |
| + *BG2EN* (unadapted) | **27.33** | 54.61 | 48.16 |
| + *CN:word′+morph* | **27.97** | 54.08 | 49.65 |
| + *PPT:phrase4::CN:morph* | **28.38** | 53.35 | 48.21 |
| + *DD:phrase4+morph* | **28.44** | 53.51 | 50.95 |
| Combining last four | **29.35** | 51.83 | 51.63 |

Table 5.9: **Improving Macedonian-English SMT by adapting Bulgarian to Macedonian.** The scores that are significantly better ($p < 0.01$) than *BG2EN* and *MK2EN* are in **bold** and underlined, respectively. The last line shows system combination results using MEMT.

Table 5.9 shows that all the three approaches (*CN:**, *PPT:** and *DD:**) outperforms the balanced concatenation with unadapted *BG2EN*. Moreover, system combination with MEMT improves even further. This indicates that our approach can work for other pairs of closely related languages and even for other domains.

We should note that the improvements here are less sizeable than those for Malay-Indonesian adaptation. This may be due to the fact that our monolingual Macedonian dataset is much smaller that the monolingual Indonesian data set (10M Macedonian vs. 20M Indonesian words). Also, our monolingual Macedonian dataset is too noisy, since it contains many OCR errors, typos, concatenated words, and even some Bulgarian text. Moreover, Macedonian and Bulgarian are arguably somewhat more dissimilar than Malay and Indonesian. Our source language adaptation approaches assume the two languages are closely related and share some words and phrases. Thus, the more different the two languages are, the worse performance we can get.

### 5.5.5 Differences between the Source Language Adaptation Decoder and the Phrase-Level Paraphrasing Approach

In our previous work (Wang et al., 2012a), the phrase-level paraphrasing approach performed better than the word-level paraphrasing approach. Essentially, the phrase-level paraphrasing approach uses the standard phrase-based SMT decoder to perform source language adaptation with a pivoted phrase table.

In the current work, we have shown that the proposed source language adaptation decoder outperforms the phrase-level paraphrasing approach. The main differences between the two approaches can be summarized as follows:

- The standard phrase-based SMT decoder works at the phrase level, while the proposed decoder works at the sentence level. As a result, the proposed decoder can utilize sentence-level features, e.g., the language model score of the whole sentence. Even though in the standard SMT decoder, we also use a language model score as a feature function, the score is actually an estimation, since the target sentence is incomplete before the final iteration.

- Due to the general framework of the text rewriting decoder presented in Chapter 3, the proposed source language adaptation decoder can use more types of feature functions, e.g., the Malay word penalty, while the traditional SMT decoder often utilizes limited types of feature functions.

- It is more straightforward to add the cross-lingual morphological variants to the proposed decoder, i.e., as a hypothesis producer. In contrast, in the phrase-level paraphrasing approach, we have to transform the sentences in the *development* and the *test* sets into confusion networks, which contains the additional morphological variants.

- The proposed decoder can also use some rule-based hypothesis producers (e.g., the

number adaptation discussed in Section 5.2.1.4), while it is not easy to add such kind of methods to a standard SMT decoder.

## 5.6 Summary

In this chapter, we have proposed to apply the text rewriting decoder of Chapter 3 to source language adaptation for resource-poor machine translation, and compared the decoder approach with two other approaches proposed in our previous work (Wang et al., 2012a): (1) word-level paraphrasing approach using confusion networks; and (2) phrase-level paraphrasing approach using pivoted phrase tables.

We have achieved very significant improvements over several baselines (7.26% BLEU scores over an unadapted version of *ML2EN*, 3.09% BLEU scores over *IN2EN*, and 1.93-3.25% BLEU scores over three bi-text combinations of *ML2EN* and *IN2EN*), thus proving the potential of the idea, source-language adaptation for resource-poor machine translation. We have further demonstrated the applicability of the general approach to other languages and domains.

# Chapter 6

# Conclusion and Future Work

The primary objective of this thesis is to devise a beam-search text rewriting decoder, and then apply it to two applications: normalization of social media text and source language adaptation. We investigate two issues: (1) performing social media text normalization for machine translation using the proposed text rewriting decoder; (2) adapting the source side of bi-texts for resource-rich languages to help the translation of a related resource-poor language.

## 6.1 Conclusion

### 6.1.1 Normalization of Social Media Text with Application to Machine Translation

To better translate social media texts without social media training bi-texts, we propose to apply our text rewriting decoder to social media text normalization, with a view towards applying it to machine translation. Although word substitutions have been investigated in previous work, we argue that some other normalization operations are also useful, e.g., missing word recovery and punctuation correction.

To show the applicability of our approach, we experiment with two languages, Chinese and English. In the experiments, we have achieved statistically significant improvements over two strong baselines: an improvement of 9.98%/7.35% in BLEU scores for normalization of Chinese/English social media text, and an improvement of 1.38%/1.35% in BLEU scores for translation of Chinese/English social media text.

As far as we know, our work is the first to perform punctuation correction and missing word recovery for normalization of social media text. These two operations proved effective for machine translation in the experiments. We have also created two corpora: a Chinese corpus containing 1,000 Weibo messages with their normalizations and English translations; another similar English corpus including 2,000 messages from the NUS SMS corpus (How and Kan, 2005). As far as we know, these two corpora are the first publicly available Chinese and English corpora for normalization and translation of social media text.

### 6.1.2 Source Language Adaptation for Resource-Poor Machine Translation

As most of the world languages still remain resource-poor for machine translation and many resource-poor languages are actually related to some resource-rich languages, to help machine translation of a resource-poor language, we apply the text rewriting decoder to source language adaptation for resource-poor machine translation. Moreover, we compare the decoder with two approaches from our previous work (Wang et al., 2012a): (1) word-level paraphrasing using confusion networks; and (2) phrase-level paraphrasing using pivoted phrase tables.

More precisely, assuming a large $RICH$-$TGT$ bi-text for a resource-rich language and a small $POOR$-$TGT$ bi-text for a related resource-poor language, we use our text rewriting decoder to adapt the $RICH$ side of the $RICH$-$TGT$ bi-text to get closer to $POOR$, thus obtaining a synthetic "$POOR$"-$TGT$ bi-text which is combined with

the original $POOR$-$TGT$ bi-text to improve the translation from $POOR$ to $TGT$.

Using a resource-rich Malay-English bi-text and a resource-poor Indonesian-English bi-text, we have achieved very significant improvements over several baselines: (1) 7.26% BLEU scores over an unadapted version of the Malay-English bi-text; (2) 3.09% BLEU scores over the Indonesian-English bi-text; and (3) 1.93-3.25% BLEU scores over three bi-text combinations of the Malay-English and Indonesian-English bi-texts. We thus prove the potential of the idea, source-language adaptation of a resource-rich bi-text to improve machine translation for a related resource-poor language. We have further demonstrated the applicability of the general approach to other languages and domains.

Our work is of importance for resource-poor machine translation since it can provide a useful guideline for people building machine translation systems of resource-poor languages. They can adapt bi-texts for related resource-rich languages to the resource-poor language, and subsequently improve the resource-poor language translation using the adapted bi-texts.

## 6.2 Future Work

### 6.2.1 Normalization of Social Media Text with Application to Machine Translation

Future study may investigate how to tightly integrate our beam-search decoder for text normalization with a standard SMT system, since in the current study, only the 1-best output for each input message is used to generate the translation. To accomplish this, there are three potential directions as follows:

- **n-best list:** One possible direction is to get an n-best list as the normalization output for each input message, and then translate each output in the n-best list using the SMT system individually. We eventually choose the best translation

output generated by the SMT system as the final translation for the input message, according to some metric, e.g., the language model score of the target language.

- **lattice:** Another potential direction is through source lattice translation of SMT systems (Dyer, 2007; Du et al., 2010). Given an input message, the text normalization decoder generates a lattice as the normalization output. Then we use the SMT system to directly translate the lattice. Using a lattice, we can pass more varieties of normalization output from the normalization decoder to the SMT system, compared to the previous direction.

- **a combined decoder:** Another way is to integrate the normalization decoder with the SMT decoder together. As a result, we can jointly perform text normalization and translation. In this way, we will have no loss of normalization information.

### 6.2.2 Source Language Adaptation for Resource-Poor Machine Translation

In order to further improve our work on source language adaptation for resource-poor machine translation, future studies could attempt the following directions:

- One direction is to add more word editing operations, e.g., word deletion, insertion, splitting, and concatenation, because we mainly focused on word substitution in this study.

- Another direction is to add word reordering. In the current work, we assume no word reordering is needed, but there actually exist some word reordering differences between closely related languages.

- One more direction is to utilize the relationships between the source and target sides of the input resource-rich bi-text to perform language adaptation, since only the source side was used in our current work. For example, in our Malay-

Indonesian adaptation work, we may adapt a Malay word considering the English words which the Malay word is aligned to in the word alignments for the Malay-English bi-text.

- Another direction is to experiment with other closely related language pairs, e.g. the language pairs proposed in Section 1.3.

- Further work may apply the language adaptation idea to other linguistic problems, e.g., we may adapt the Malay training data for part-of-speech (POS) tagging to "Indonesian" in order to help Indonesian POS tagging.

# References

Kemal Altintas and Ilyas Cicekli. 2002. A machine translation system between a pair of closely related languages. In *Proceedings of the 17th International Symposium on Computer and Information Sciences*, ISCIS '02, pages 192–196.

AiTi Aw, Min Zhang, PohKhim Yeo, ZhenZhen Fan, and Jian Su. 2005. Input normalization for an English-to-Chinese SMS translation system. In *Proceedings of the Tenth Machine Translation Summit*, MT Summit X.

AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for SMS text normalization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, ACL-COLING '06, pages 33–40.

Hitham Abo Bakr, Khaled Shaalan, and Ibrahim Ziedan. 2008. A hybrid approach for converting written Egyptian colloquial dialect into diacritized Arabic. In *Proceedings of the 6th International Conference on Informatics and Systems*, ICSAI '08, pages 27–33.

Timothy Baldwin and Su'ad Awab. 2006. Open source corpus analysis tools for Malay. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, LREC '06, pages 2212–2215.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.

Richard Beaufort, Sophie Roekhaut, Louise-Amélie Cougnon, and Cédrick Fairon. 2010. A hybrid rule/model-based finite-state framework for normalizing SMS messages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 770–779.

Alexandra Birch, Miles Osborne, and Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, WMT '07, pages 9–16.

Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 286–293.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 17–24.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 22–64.

Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal on Document Analysis and Recognition*, 10(3-4):157–174.

Trevor Cohn and Mirella Lapata. 2007. Machine translation by triangulation: Making effective use of multi-parallel corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL '07, pages 728–735.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL '05, pages 531–540.

Paul Cook and Suzanne Stevenson. 2009. An unsupervised model for text message

normalization. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 71–78.

Marta R. Costa-jussà and Rafael E. Banchs. 2011. The BM-I2R Haitian-Créole-to-English translation system description for the WMT 2011 evaluation campaign. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 452–456.

Daniel Dahlmeier and Hwee Tou Ng. 2012. A beam-search decoder for grammatical error correction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 568–578.

Hal Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at `http://pub.hal3.name#daume04cg-bfgs`, implementation available at `http://hal3.name/megam/`.

Jinhua Du, Jie Jiang, and Andy Way. 2010. Facilitating translation using source language paraphrase lattices. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 420–429.

Chris Dyer. 2007. The University of Maryland translation system for IWSLT 2007. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT '07, pages 180–185.

Vladimir Eidelman, Kristy Hollingshead, and Philip Resnik. 2011. Noisy SMS machine translation in low-density languages. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 344–350.

Jennifer Foster, Özlem Çetinoglu, Joachim Wagner, Joseph Le Roux, Stephen Hogan, Joakim Nivre, Deirdre Hogan, and Josef Van Genabith. 2011. # hardtoparse: POS tagging and parsing the twitterverse. In *Proceedings of the Workshop on Analyzing Microtext (AAAI 2011)*, pages 20–25.

Stephan Gouws, Dirk Hovy, and Donald Metzler. 2011. Unsupervised mining of lexical

variants from noisy text. In *Proceedings of the First Workshop on Unsupervised Learning in NLP*, pages 82–90.

Jan Hajič, Jan Hric, and Vladislav Kuboň. 2000. Machine translation of very close languages. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLP '00, pages 7–12.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL-HLT '11, pages 368–378.

Kenneth Heafield and Alon Lavie. 2010. Combining machine translation output with open source: The Carnegie Mellon multi-engine machine translation scheme. *The Prague Bulletin of Mathematical Linguistics*, 93(1):27–36.

Magnus Rudolph Hestenes and Eduard Stiefel. 1952. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436.

Sanjika Hewavitharana, Nguyen Bach, Qin Gao, Vamshi Ambati, and Stephan Vogel. 2011. CMU Haitian Creole-English translation system for WMT 2011. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 386–392.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1352–1362.

Yijue How and Min-Yen Kan. 2005. Optimizing predictive text entry for short message service on mobile phones. In *Proceedings of Human Computer Interfaces International*, HCII '05.

Jing Huang and Geoffrey Zweig. 2002. Maximum entropy model for punctuation an-

notation from speech. In *Proceedings of International Conference on Spoken Language Processing*, ICSLP '02, pages 917–920.

Frederick Jelinek. 1997. *Statistical methods for speech recognition*. MIT press.

Ji Hwan Kim and P. C. Woodland. 2001. The use of prosody in a combined system for punctuation generation and speech recognition. In *Proceedings of Eurospeech*, Eurospeech '01.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, ICASSP '95, pages 181–184.

Catherine Kobus, François Yvon, and Géraldine Damnati. 2008. Normalizing SMS: are two metaphors better than one? In *Proceedings of the 22nd International Conference on Computational Linguistics*, COLING '08, pages 441–448.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, ACL '07, pages 177–180.

Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press.

Philipp Koehn. 2013. Moses user manual and code guide. Paper available at `http://www.statmt.org/moses/manual/manual.pdf`.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, ICML '01, pages 282–289.

Zhifei Li and David Yarowsky. 2008. Mining and modeling relations between formal and informal Chinese phrases from web corpora. In *Proceedings of the 2008*

*Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1031–1040.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, COLING-ACL '06, pages 761–768.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL-HLT '11, pages 359–367.

Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, ACL '12, pages 1035–1044.

Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys*, 40(3).

Wei Lu and Hwee Tou Ng. 2010. Better punctuation prediction with dynamic conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 177–186.

Yuval Marton, Chris Callison-Burch, and Philip Resnik. 2009. Improved statistical machine translation using monolingually-derived paraphrases. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP '09, pages 381–390.

Luís Marujo, Nuno Grazina, Tiago Luís, Wang Ling, Luísa Coheur, and Isabel Trancoso. 2011. BP2EP - adaptation of Brazilian Portuguese texts to European Portuguese. In *Proceedings of the 15th Conference of the European Association for Machine Translation*, EAMT '11, pages 129–136.

Preslav Nakov and Hwee Tou Ng. 2009. Improved statistical machine translation for

resource-poor languages using related resource-rich languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP '09, pages 1358–1367.

Preslav Nakov and Hwee Tou Ng. 2012. Improving statistical machine translation for a resource-poor language using related resource-rich languages. *Journal of Artificial Intelligence Research*, 44:179–222.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, ACL '12, pages 301–305.

Preslav Nakov, Chang Liu, Wei Lu, and Hwee Tou Ng. 2009. The NUS statistical machine translation system for IWSLT 2009. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT '09, pages 91–98.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, ACL '03, pages 160–167.

J. Oliva, J. I. Serrano, M. D. Del Castillo, and Á. Igesias. 2012. A SMS normalization system integrating multiple grammatical resources. *Natural Language Engineering*, 1(1):1–21.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, ACL '02, pages 311–318.

Michael Paul. 2009. Overview of the IWSLT 2009 evaluation campaign. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT '09.

Adam Pauls and Dan Klein. 2011. Faster and smaller n-gram language models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL-HLT '11, pages 258–267.

Deana L. Pennell and Yang Liu. 2011. A character-level machine translation approach for normalization of SMS abbreviations. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, IJCNLP '11, pages 974–982.

Eric Ristad and Peter Yianilos. 1998. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1524–1534.

Stuart Russell and Peter Norvig. 2010. *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Wael Salloum and Nizar Habash. 2011. Dialectal to Standard Arabic paraphrasing to improve Arabic-English statistical machine translation. In *Proceedings of the Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, pages 10–21.

Hassan Sawaf. 2010. Arabic dialect handling in hybrid machine translation. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas*, AMTA '10.

Kevin P. Scannell. 2006. Machine translation for closely related language pairs. In *Proceedings of the LREC 2006 Workshop on Strategies for Developing Machine Translation for Minority Languages*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, AMTA '06, pages 223–231.

Andreas Stolcke. 2002. SRILM–an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, ICSLP '02, pages 901–904.

Sara Stymne. 2011. Spell checking techniques for replacement of unknown words and data cleaning for Haitian Creole SMS translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 470–477.

Charles Sutton, Khashayar Rohanimanesh, and Andrew McCallum. 2004. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04.

Charles Sutton. 2006. GRMM: GRaphical Models in Mallet. Implementation available at `http://mallet.cs.umass.edu/grmm/`.

Jörg Tiedemann. 2009. News from OPUS - a collection of multilingual parallel corpora with tools and interfaces. In *Proceedings of the Recent Advances in Natural Language Processing*, RANLP '09, pages 237–248.

Masao Utiyama and Hitoshi Isahara. 2007. A comparison of pivot methods for phrase-based statistical machine translation. In *Proceedings of the Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT '07, pages 484–491.

Martin J. Wainwright, Tommi Jaakkola, and Alan S. Willsky. 2001. Tree-based reparameterization for approximate inference on loopy graphs. In *Proceedings of the Advances in Neural Information Processing Systems*, NIPS '01, pages 1001–1008.

Aobo Wang and Min-Yen Kan. 2013. Mining informal language from chinese microtext: Joint word recognition and segmentation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, pages 731–741.

Pidong Wang and Hwee Tou Ng. 2013. A beam-search decoder for normalization of social media text with application to machine translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '11, pages 471–481.

Pidong Wang, Preslav Nakov, and Hwee Tou Ng. 2012a. Source language adaptation for resource-poor machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 286–296.

Xuancong Wang, Hwee Tou Ng, and Khe Chai Sim. 2012b. Dynamic conditional random fields for joint sentence boundary and punctuation prediction. In *Proceedings of the 13th Annual Conference of the International Speech Communication Association*, Interspeech '12.

Aobo Wang, Min-Yen Kan, Daniel Andrade, Takashi Onishi, and Kai Ishikawa. 2013. Chinese informal word normalization: an experimental study. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, IJCNLP '13, pages 127–135.

Robert L. Weide. 1998. The CMU pronouncing dictionary. *URL: http://www.speech.cs.cmu.edu/cgi-bin/cmudict*.

Hua Wu and Haifeng Wang. 2009. Revisiting pivot language approach for machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, ACL-IJCNLP '09, pages 154–162.

Yunqing Xia, Kam-Fai Wong, and Wei Gao. 2005. NIL is not nothing: recognition of Chinese network informal language expressions. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing at IJCNLP*, pages 95–102.

Zhenzhen Xue, Dawei Yin, and Brian D. Davison. 2011. Normalizing microtext. In *Proceedings of the Workshop on Analyzing Microtext (AAAI 2011)*, pages 74–79.

Steve Young, Gunnar Evermann, Dan Kershaw, Gareth Moore, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland. 2002. The HTK book. *Cambridge University Engineering Department*, 3.

Xiaoheng Zhang. 1998. Dialect MT: a case study between Cantonese and Mandarin. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, ACL-COLING '98, pages 1460–1464.

Conghui Zhu, Jie Tang, Hang Li, Hwee Tou Ng, and Tie-Jun Zhao. 2007. A unified tagging approach to text normalization. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL '07, pages 688–695.