博士論文

Translation and Description Methods for Multilingual Text Understanding

（多言語テキスト理解のための
翻訳および語義説明手法）

48-167401

石渡　祥之佑

指導教員　喜連川　優

大学院　情報理工学系研究科　電子情報学専攻

東京大学

This dissertation is submitted for the degree of
Doctor of Philosophy

December 2018

Acknowledgements

First and foremost, my deepest gratitude is to my advisor, Prof. Masaru Kitsuregawa, for his continuous support of my Ph.D. study. During the five years I have spent in Kitsuregawa-Toyoda-Nemoto-Yoshinaga lab, he always provided me with the best environment to focus on my research and the greatest opportunities to learn from the top-level researchers in the world. I would also like to express my gratitude to the members of my dissertation committee: Prof. Sadao Kurohashi (Kyoto University), Prof. Akiko Aizawa, and Prof. Yoshimasa Tsuruoka. Their serious review and helpful comments on this work made my thesis more complete and solid.

I would like to express my great appreciation to my co-advisor Prof. Naoki Yoshinaga for the long discussions, not only for research but also for all the things in my life. He is not only an outstanding researcher but also an excellent teacher who had been guiding me during the long journey to the Ph.D. As is often the case, there were many painful rejections during my Ph.D. study. Every time I got a rejection from top-conferences, scholarships, or graduate schools, Prof. Yoshinaga always supported me and encouraged me to prepare for the next opportunities.

I am deeply grateful to another co-advisor, Prof. Masashi Toyoda. Despite being extremely busy with the management of the huge research projects, he always helped me with my papers. I could not accomplish my Ph.D. study without his useful comments on my research and his help with the wonderful computing resources.

I am grateful to Dr. Nobuhiro Kaji and Dr. Danushka Bollegala, who were my advisors during the master course and undergraduate study. I got interested in the area of Natural Language Processing because of their excellent teaching.

I would like to express my gratitude to my mentors at Microsoft Research Asia, Dr. Shujie Liu, Dr. Mu Li, and Dr. Ming Zhou. During the half-year internship at MSRA, I learned a lot of techniques to do cutting-edge research in a competitive area. Life in Beijing was one of the most beautiful memories of my life.

I would also like to express my gratitude to my mentors at Carnegie Mellon University, Dr. Graham Neubig and Mr. Hiroaki Hayashi. The half-year stay at CMU taught me the crucial techniques to "understand machine learning models",

such as error analysis or visualization, and many tips for writing a good paper. Things that I learned in Pittsburgh in the cold winter have become essential skills through my professional carrier.

I would also like to thank all researchers, students, and secretaries of the Kitsuregawa-Toyoda-Nemoto-Yoshinaga lab. Especially, I thank Mr. Shoetsu Sato, who helped me a lot with my work as a co-author. Thanks to his help, I could publish a top-conference paper and finish this thesis. The funny conversations with Mr. Takashi Kawamoto, Mr. Ryo Koyajima, and Mr. Kohei Ohara in hot sauna always helped me to relax when I was tired. Thanks to them, my five-year life as a graduate student will be my lifetime treasure.

Finally, I would like to dedicate this thesis to my parents and sister, who have been supporting me during my whole life.

Abstract

The development of Web technologies has been rapidly accelerating human communication and sharing of knowledge. The massive amount of text on the new communication platforms or knowledge sources often consists of documents in several domains and multiple languages. In order to obtain fresh and diverse information from multilingual and diverse text sources such as Twitter, Wikipedia, or arXiv, we need to cope with language barrier while also paying attention to domain differences.

Let us move on to the general topic of natural language processing. Machine translation, as one of the most promising applications of natural language processing, has been playing an essential role in overcoming the language barrier. The recent development of machine learning techniques and huge annotated corpora have considerably improved the performance of machine translation. In the face of the increasing use of multilingual platforms and knowledge sources, can machine translation help us understand the *real* text data in various domains? Can machine translation be applied to languages pairs whose vocabulary and grammar are significantly different (such as English vs. Japanese)? More generally, can machine directly help humans understand text written in unfamiliar domains/languages? These are the central topics of this thesis.

To answer the above questions, we propose **an instant domain adaptation method**, **an accurate translation method for English-to-Japanese translation**, and **an automatic description method for unknown phrases**.

- **Instant domain adaptation for Statistical Machine Translation**: To translate text in various domains, the most basic method is domain adaptation. Most studies on domain adaptation require supervised in-domain resources such as parallel corpora or in-domain dictionaries. The necessity of supervised data has made such methods difficult to adapt to practical machine translation systems. In this thesis, we thus propose a method that adapts translation models without

in-domain parallel corpora. Our method improves out-of-domain translation from Japanese to English by 0.5-1.5 BLEU score.

- **Accurate translation method for English-to-Japanese translation**: English-to-Japanese translation is more difficult than other language pairs such as English-to-German or English-to-French translations. This is mainly because (1) Japanese sentence has much more words in a sentence compared to English, and (2) Japanese is a free-word-order language. To cope with these problems, we propose a chunk-based decoder for neural machine translation. Our method improves English-to-Japanese translation by 0.93 BLEU score and achieves state-of-the-art performance on the WAT '16 translation task.

- **Automatic description generation for unknown phrases**: Even if a text is translated perfectly, or written in our familiar languages, it is still common for humans to become stuck on unfamiliar words or phrases. To help humans understand unknown phrases which are not included in hand-crafted dictionaries, we undertake a task of describing a given phrase in natural language based on its contexts. In contrast to the existing methods, our model appropriately takes important clues from contexts and achieves state-of-the-art performance in four description generation datasets.

To help humans understand *real* multilingual text is a challenging task because (1) the target domain is unknown, (2) the source language may extremely differ from the users' languages, and (3) the users may be unfamiliar with the words/phrases in the text. Our proposed methods tackle these problems by (1) instant domain adaptation, (2) accurate English-to-Japanese translation, and (3) automatic description generation. We expect that this thesis will provide a promising future direction for research into multilingual text understanding.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Multilingual Text on the Web

The development of Web technologies has been rapidly accelerating human communication and sharing of knowledge. Statistics show that the daily active users on Facebook in 2018 reached the amount of 1.49 billion [1]. On the other hand, 141 thousand of academic papers are uploaded on arXiv [2], the world's largest academic paper platform, in 2018.

These platforms enable us to obtain the (1) latest information (e.g., news in various academic fields), and (2) regional information (e.g., terror incidents occur at a specific area) instantaneously. For we humans, however, it is not easy to understand all of those new and various information effectively because most of them are described in unfamiliar languages for readers (e.g., Facebook is used in more than 100 languages). Therefore, the natural language processing technologies, such as machine translation or cross-lingual information retrieval, are becoming more and more desirable to help human understanding.

1.2 Towards Understanding the Multilingual Text

How can we humans understand the information in the multilingual text efficiently? In this thesis, we define two requirements for the *understandable* text. Fist, the text needs to be translated into the target languages accurately. Since inaccurate translation usually avoids humans to understand the original meanings of the text,

[1] "Facebook Demographics & Usage", https://adespresso.com/blog/facebook-statistics/
[2] "arXiv monthly submission rates", https://arxiv.org/stats/monthly_submissions

we need an accurate translation methodology. Second, the text needs to be clear and straightforward. If the readers are not experts in the fields of the text, we need to simplify the technical terms or attach the descriptions for those terms. If we can generate a text that meets these requirements from the multilingual text source, we can obtain fresh and various information from the text.

1.3 Research Challenges

Machine translation, as one of the most promising applications of natural language processing, has been playing an essential role in overcoming the language barrier. The recent development of machine learning techniques and huge annotated corpora have considerably improved the performance of machine translation. However, current machine translation technology is not yet a perfect solution for human understanding of the multilingual text.

In the previous section, we defined two requirements of the readable text. We summarize the difficulties of the language and domain differences from the viewpoints of machine translation and human understanding. From the perspective of machine translation, (1) domain differences between training and test data deteriorate the translation performance. Also, (2) machine translation between the distant language pairs (e.g., English to Japanese) is still difficult because of their significant differences of vocabulary and grammar. On the other hand, from the viewpoint of human understanding, (3) we can be stuck when reading a text in unfamiliar domains, even if it is written in our native languages.

In the face of the increasing use of multilingual platforms and knowledge sources, can machine translation help us understand the *real* text data in various domains? Can machine translation be applied to languages pairs whose vocabulary and grammar are significantly different (such as English vs. Japanese)? More generally, can machine directly help humans understand text written in unfamiliar domains? These are the central topics of this thesis.

1.4 Contributions

In order to obtain fresh and diverse information from multilingual and diverse text sources such as Twitter, Wikipedia, or arXiv, we need to cope with language barriers while also paying attention to domain differences.

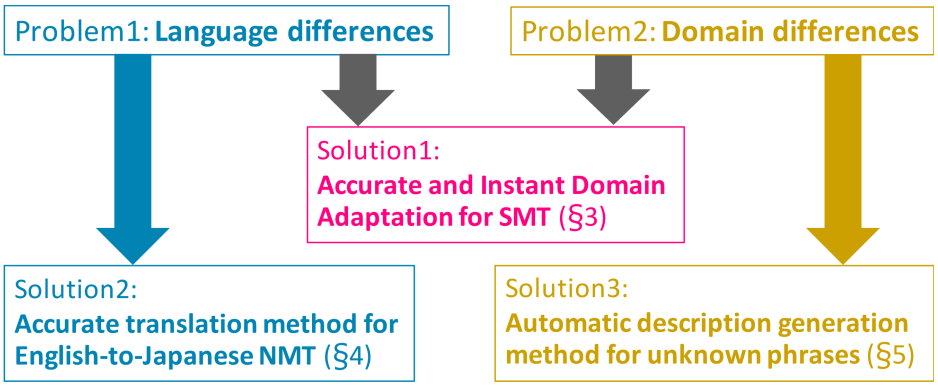


Figure 1.1: The problems for humans to understand multilingual text and the solutions presented in this thesis.

Figure 1.1 shows the two problems and the solutions presented in this thesis. To answer the questions in the previous section, we propose **an accurate and instant domain adaptation method for statistical machine translation**, **an accurate translation method for English-to-Japanese neural machine translation**, and **an automatic description generation method for unknown phrases**.

- **Accurate and instant domain adaptation for statistical machine translation**: To translate text in various domains, the most basic method is domain adaptation. Most studies on domain adaptation require supervised in-domain resources such as parallel corpora or in-domain dictionaries. The necessity of supervised data has made such methods difficult to adapt to practical machine translation systems. In this thesis, we thus propose a method that adapts translation models without in-domain parallel corpora. Our method improves out-of-domain translation from Japanese to English by 0.5-1.5 BLEU score.

- **Accurate translation method for English-to-Japanese neural machine translation**: English-to-Japanese translation is more difficult than other language pairs such as English-to-German or English-to-French translations. This is mainly because the differences in vocabulary and grammar between Japanese and English are bigger than other language pairs. To cope with this problem, we propose a chunk-based decoder for neural machine translation. Our method improves English-to-Japanese translation by 0.93 BLEU score and achieves state-of-the-art performance on the WAT '16 translation task.

- **Automatic description generation method for unknown phrases**: Even if a text is translated perfectly, or written in our familiar languages, it is still

common for humans to become stuck on unfamiliar words or phrases. To help humans understand unknown phrases which are not included in hand-crafted dictionaries, we undertake a task of describing a given phrase in natural language based on its contexts. In contrast to the existing methods, our model appropriately takes important clues from contexts and achieves state-of-the-art performance in four description generation datasets.

To help humans understand *real* multilingual text is a challenging task because (1) the target domain is unknown, (2) the source language may extremely differ from the users' languages, and (3) the users may be unfamiliar with the words/phrases in the text. Our proposed methods tackle these problems by (1) instant domain adaptation, (2) accurate English-to-Japanese translation, and (3) automatic description generation. We expect that this thesis will provide a promising future direction for research into multilingual text understanding.

1.5 Thesis Structure

The rest of this thesis is structured as follows. In Chapter 2, we introduce the related work from the viewpoints of the three challenges described above. This is followed by Chapter 3, in which we propose a method of accurate and instant domain adaptation for statistical machine translation (SMT). In Chapter 4, we present a new neural machine translation (NMT) decoder that provides accurate English-to-Japanese translation. In Chapter 5, we set up a task of describing phrases and propose a model to generate a description to help human understand unfamiliar expressions. Finally, in Chapter 6, we conclude the three proposed methods and address future work in the research area of multilingual text understanding.

Chapter 2

Related Work

In this chapter, we introduce the related work from the view points of the three challenges described in Chapter 1. Section 2.1 describes the motivation and related work of domain adaptation for SMT. This is followed by Section 2.2, in which we introduce the efforts to improve machine translation for distant language pairs by utilizing phrase or chunk structures, from the viewpoints of statistical machine translation and newly appeared neural machine translation. Finally, in Chapter 2.3, we present the related work of our description generation task, such as word sense disambiguation, paraphrasing, and definition generation tasks.

2.1 Out-of-vocabulary Word Translation and Domain Adaption for Statistical Machine Translation

To translate text in various domains, domain difference between train and test data causes a serious deterioration of translation performance. One of the major reasons for this phenomenon is out-of-vocabulary(oov) words. Since there exist several oov words if the domains of train and test data are different, we can improve the overall translation performance if we have an accurate oov word translation method.

We discuss the related studies of our translation model adaptation by categorizing them into two tasks: oov word translation, and more generally, the domain adaptation for SMT.

2.1.1 Out-of-vocabulary Word Translation

oov word translation has been inherently a difficult task since there is no perfect multilingual dictionay available in the world. New words and new usages of words are emerging everyday, and thus it is not possible for human to maintain a dictionary that covers all words in all languages. To cope with this problem, researchers have been working on the methods for translating oov words by using monolingual corpora. The basic idea of oov word translation is: (1) assign a word vector to the source oov word, and (2) map it to target space, then (3) find the translation candidates using nearest-neighbor search in the target space. This idea of vector-based oov translation was originally proposed by Fung [31] for the purpose of automatic extraction of bilingual dictionary. Their direct mapping method can be achieved without any machine learning methods because it simply maps a word vector by translating the context words, each of which is assigned to a specific dimension of the source vector. Since the translation of context words is based on a seed bilingual dictionary, the accuracy of vector mapping can be extremely low if the seed dictionary does not cover many of context words.

To cope with this problem, Mikolov et al.[69] reformulated the mapping of word vectors as a linear transformation between source and target vector spaces. By using the seed dictionary as training data, their method learns a linear mapping function (translation matrix) that translates a source word vector into the target space. While Fung's method directly maps vectors using hard constraints (i.e., the correspondence between dimensions of source and target vector spaces), Mikolov's translation matrix automatically learns soft correspondence between two spaces.

Mapping is not the only way to utilize word vectors in multi-languages. There also exist several methods that directly induce word vectors shared by different languages [12, 27, 36, 39, 46, 113]. Since these methods require a huge parallel corpus to train the multilingual word vectors, they cannot be applied to language pairs that have no parallel corpus. In addition, these approaches are unable to handle words not appearing in the training data, unlike the aforementioned mapping-based approaches [31, 70].

2.1.2 Domain Adaptation for Statistical Machine Translation

Most previous approaches to domain adaptation for SMT assume a scenario where a small or pseudo in-domain parallel corpus is available. In this section, we briefly

overview a method of domain adaptation for SMT in a setting where no in-domain parallel corpus is available.

Wu et al. [110] have proposed domain adaptation for SMT that exploits an in-domain bilingual dictionary. They generate a translation model from the bilingual dictionary and combine it with the translation model learned from out-of-domain parallel corpora. An issue here is how to learn a translation probability between words (or phrases) needed for the translation model, and they resort to probabilities of words in the target language in a monolingual corpus. Although building a bilingual dictionary for the target domain is more effective than developing a parallel corpus to cover rare oov words, it is still difficult to develop a bilingual dictionary for most MT users who cannot command the target language.

To cope with this problem, several researchers have recently exploited a bilingual lexicon automatically induced from in-domain corpora to generate a translation model for SMT [20, 41, 93]. These approaches induce a bilingual lexicon from in-domain comparable corpora prior to the translation and use it to obtain an in-domain translation model.

Marthur et al. [65] exploit parallel corpora in various domains to induce the translation model for the target domain. They used 11 sets of parallel corpora for domains including TED talks, news articles, and software manuals to train the translation model for each domain and then linearly interpolated these translation models to derive a translation model for the target domain. They successfully improved the quality of translation when no parallel corpus was available for the target domain. Yamamoto and Sumita [114] assume various language expressions in translating travel conversations and train several language and translation models from a set of parallel corpora that are split by unsupervised clustering of the entire parallel corpus for travel conversations. The language and translation models for translating a given sentence are chosen in accordance with the similarity between the given sentence and the sentences in each split of the parallel corpus. Although this method is not intended for domain adaptation, it can be used in our setting when we have a parallel corpus for the general domain (and the domain of the target sentence is included in the general domain). These studies, however, implicitly assume in-domain (or related domain) parallel corpora are available, while we assume those resources are unavailable to broaden the applicability of our method.

Among these studies, our method is most closely related to domain adaptation using bilingual lexicon induction [20, 41, 93] but is different from these approaches in that it does not need to build a sort of bilingual lexicon prior to the translation

to support the translation of oov words in a given sentence. We use a projection of semantic representations of source-language words to the target-language semantic space to dynamically find translation candidates of found oov words by computing the similarity of the obtained representations to semantic representations for words in the target language at the time of translation. Also, we empirically show that our approach could even benefit from general-domain non-comparable monolingual corpora instead of in-domain comparable monolingual corpora used in these studies on bilingual lexicon induction.

2.2 Utilizing Chunk Structures in Neural Machine Translation

Neural machine translation performs end-to-end translation based on a simple encoder-decoder model and has now overtaken the classical, complex statistical machine translation in terms of performance and simplicity. With the word-based neural decoder, however, there are two problems to be solved: the long-distance dependencies and free word-order in some languages. To cope with these problems, in Chapter 4, we propose a chunk-based decoder for NMT. The proposed decoder generates sentences in a chunk-by-chunk manner instead of word-by-word manner and improves the performance of English-to-Japanese translation. In this section, we introduce the related work that uses chunk (or phrase) structure to improve machine translation quality.

The most notable work involved phrase-based SMT [51], which has been the basis for a huge amount of work on SMT for more than ten years. Apart from this, Watanabe et al. [2003] proposed a chunk-based translation model that generates output sentences in a chunk-by-chunk manner. The chunk structure is effective not only for SMT but also for example-based machine translation (EBMT). Kim et al. [45] proposed a chunk-based EBMT and showed that using chunk structures can help with finding better word alignments. Our work is different from theirs in that our models are based on NMT, but not SMT or EBMT. The decoders in the above studies can model the chunk structure by storing chunk pairs in a large table. In contrast, we do that by individually training a chunk generation model and a word prediction model with two RNNS.

While most of the NMT models focus on the conversion between sequential data, some works have tried to incorporate non-sequential information into NMT [23, 102].

Eriguchi et al. [23] use a Tree-based LSTM [104] to encode the input sentence into context vectors. Given a syntactic tree of a source sentence, their tree-based encoder encodes words from the leaf nodes to the root nodes recursively. Su et al. [102] proposed a lattice-based encoder that considers multiple tokenization results while encoding the input sentence. To prevent the tokenization errors from propagating to the whole NMT system, their attice-based encoder can utilize multiple tokenization results. These works focus on the encoding process and propose better encoders that can exploit the structures of the source language. In contrast, our work focuses on the decoding process to capture the structure of the target language. The encoders described above and our proposed decoders are complementary so they can be combined into a single network.

Considering that our model can be seen as a hierarchical RNN, our work is also related to previous studies that utilize multi-layer RNNs to capture hierarchical structures in data. Hierarchical RNNs are used for various NLP tasks such as machine translation [60], document modeling [56, 57], dialog generation [95], image captioning [52], and video captioning [118]. In particular, Li et al. [56] and Luong and Manning [60] use hierarchical encoder-decoder models, but not for the purpose of learning syntactic structures of target sentences. Li et al. [56] build hierarchical models at the sentence-word level to obtain better document representations. Luong and Manning [60] build the word-character level to cope with the out-of-vocabulary problem. In contrast, we build a hierarchical models at the chunk-word level to explicitly capture the syntactic structure based on chunk segmentation.

In addition, the architecture of our proposed model is also related to stacked RNN, which has proven to be effective in improving the translation quality [61, 103]. Although these architectures look similar to each other, there is a fundamental difference between the directions of the connection between two layers. A stacked RNN consists of multiple RNN layers that are connected from the input side to the output side at every time step. In contrast, our model has a different connection at each time step. Before it generates a chunk, there is a feed-forward connection from the chunk-level decoder to the word-level decoder. However, after generating a chunk representation, the connection is to be reversed to feed back the information from the word-level decoder to the chunk-level decoder. By switching the connections between two layers, our model can capture the chunk structure explicitly. This is the first work that proposes decoders for NMT that can capture plausible linguistic structures such as chunk.

Finally, we noticed that [120] have also proposed a chunk-based decoder for NMT. Their good experimental result on Chinese to English translation task also indicates the effectiveness of "chunk-by-chunk" decoders. Although their architecture is similar to our model, there are several differences: (1) they adopt chunk-level attention instead of word-level attention; (2) their model predicts chunk tags (such as noun phrase), while ours only predicts chunk boundaries; and (3) they employ a boundary gate to decide the chunk boundaries, while we do that by simply having the model generate end-of-chunk tokens.

2.3 Identifying the Sense for Words and Phrases

When we read news text with emerging entities, text in unfamiliar domains, or text in foreign languages, we often encounter expressions (words or phrases) whose senses we are unsure of. To cope with this problem, in Chapter 5, we will address a task of describing a given phrase with its context. In this section, we explain existing tasks that are related to our work.

Our task of describing phrases is closely related to word sense disambiguation (WSD) [76], which identifies a pre-defined sense for the target word with its context. Although we can use it to solve our task by retrieving the definition sentence for the sense identified by WSD, it requires a substantial amount of training data to handle a different set of meanings of each word, and cannot handle words (or senses) which are not registered in the dictionary. Although some studies have attempted to detect novel senses of words for given contexts [25, 53], they do not provide definition sentences. Our task avoids these difficulties in WSD by directly generating descriptions for phrases or words. It also allows us to flexibly tailor a fine-grained definition for the specific context.

Paraphrasing [5, 63] (or text simplification [99]) can be used to rephrase words with unknown senses. However, the target of paraphrase acquisition are words/phrases with no specified context. Although a few studies [17, 66, 67] consider sub-sentential (context-sensitive) paraphrases, they do not intend to obtain a definition-like description as a paraphrase of a word.

Recently, Noraset et al. [85] introduced a task of generating a definition sentence of a word from its pre-trained embedding. Since their task does not take local contexts of words as inputs, their method cannot generate an appropriate definition for a polysemous word for a specific context. To cope with this problem, Gadetsky et al. [32] proposed a definition generation method that works with polysemous

words in dictionaries. They presented a model that utilizes local context to filter out the unrelated meanings from a pre-trained word embedding in a specific context. While their method uses local context only for disambiguating the meanings that are mixed up in word embeddings, the information from local contexts cannot be utilized if the pre-trained embeddings are unavailable or unreliable. On the other hand, our method can fully utilize the local context through an attentional mechanism, even if the reliable word embeddings are unavailable.

The most related work to ours is Ni and Wang [83]. Focusing on non-standard English phrases, they proposed a model to generate the explanations solely from local context. They followed the strict assumption that the target phrase was newly emerged and there was only a single local context available, which made the task of generating an accurate and coherent definition difficult. Our proposed task and model are more general and practical than Ni and Wang [83]; where (1) we use Wikipedia, which includes expressions from various domains, and (2) our model takes advantage of global contexts if available.

Our task of describing phrases with its context is a generalization of these three tasks [85, 83, 32], and the proposed method naturally utilizes both local and global contexts of an expression in question.

Chapter 3

Accurate and Instant Translation Model Adaptation for Statistical Machine Translation

3.1 Overview

In order to obtain fresh and diverse information from multi-lingual text such as Twitter, Wikipedia, or research papers on ArXiv, we need to cope with language barrier. Machine translation, as one of the most important applications of natural language processing, has been playing an important role in overcoming the language barrier. SMT has been successfully applied to the translation between various language pairs, particularly phrase-based SMT, which is the most common since it can learn a translation model from a sentence-aligned parallel corpus without any linguistic annotations.

Although we can improve the quality of machine translation by using a large language model that can be obtained from easily available monolingual corpora [10], language models capture only the fluency in languages so the quality of translation cannot be improved much if the translation model does not provide correct translation candidates for source-language words and phrases. The quality of translation in SMT is therefore bounded by the size of parallel corpus to train the translation model. Even if a large parallel corpus is available for the pair of languages in question, we often want to translate sentences in a domain that has a different vocabulary from the domain of available parallel corpora, and this inconsistency deteriorates the quality of translation [40, 18].

Researchers have tackled this problem and proposed methods of domain adaptation that exploits a larger out-of-domain parallel corpus. They have focused on a scenario in which a small or pseudo in-domain parallel corpus is available for training [64]. In actual scenarios when users want to exploit machine translation, the target domains can differ so the domain mismatches between the prepared SMT system and the target documents are likely to occur. Domain adaptation is thus expected to improve the quality of translation. However, it is unrealistic for most MT users who cannot command the target language to prepare in-domain parallel corpora by themselves. The use of crowdsourcing for preparing in-domain parallel corpora is allowed for a few users who have a large number of documents for translation and are willing to pay money for improving the quality of translation.

In this study, we assume domain adaptation for SMT in a scenario where no sentence-aligned parallel corpus is available for the target domain. To overcome the difference of domains in training and test data, we propose an accurate and instant domain adaptation method for SMT. Our method consists of two modules to adapt translation model accurately and instantly: (1) an accurate word translator that searches the translation candidates of unknown (out-of-vocabulary, oov) words; and (2) an instant back-off model that dynamically assigns appropriate translation probabilities to the translation candidates in SMT systems.

Based on the two proposed modules, our method adapts translation model in four steps. First, assuming that source- and target-language monolingual corpora are available, we train vector-based semantic representations of words in the source and target languages from those monolingual corpora. Second, we obtain an accurate projection function from semantic representations in the source language to those in the target language. The projection function can be automatically trained with a seed dictionary (in general domain) to learn a translation matrix. Then, we map the vectors of oov words into target vector space to find translation candidates with nearest-neighbor search in the target space. Finally, our instant back-off model approximates translation probabilities of the found candidates by using their cosine-similarity with the mapped oov word in the target language. The obtained back-off model can be naturally integrated into any SMT models.

To evaluate the effectiveness of our methods, we conducted evaluations in two tasks: oov word translation and out-of-domain translation. In the oov word translation experiment, our oov translation method outperformed existing methods in four languages, Japanese (ja), Chinese (zh), English (en), and Spanish (es) without any additional supervisions. In the out-of-domain translation experiment, we apply

our methods to a translation between English and Japanese in recipe documents. Since the translation model was trained with Kyoto-related Wikipedia articles, we applied our back-off model to cope with this serious domain difference. The experimental results confirmed that our back-off model improves BLEU score by 0.5-1.5 and 0.1-0.2 for ja-en and en-ja translations, respectively.

The remainder of this chapter is structured as follows. In Section 3.3, we describe our method of accurate translation of word semantic representations. This is followed by Section 3.4, in which we propose a method of adapting SMT to a new domain without a sentence-aligned parallel corpora. In the next two sections, we present the evaluations for the proposed methods. In Section 3.5 and Section 3.6, we describe the evaluations of our proposed methods on (1) oov word translation task, and (2) domain adaptation task for SMT, respectively. Finally, in Section 3.7, we conclude this study and address future work.

3.2 Preliminaries: Statistical Machine Translation

In this section, we briefly introduce the history and the architecture of statistical machine translation. The original idea of machine translation was first discussed by Warren Weaver in a letter to Nobert Wiener [84]. His idea of "using digital computers to translate documents between natural human languages" has been the basis of machine translation until today.

In 1950's, the early research on machine translation had been focusing on rule-based methods. In the rule-based machine translation systems, computers translate text based on the rules that are described by the linguists who were familiar with both the source and target languages. In late 1980's, statistical machine translation [11] (SMT) was newly proposed. Since SMT automatically extracts the translation rules from massive parallel corpora, it no longer requires human linguists to write down the language-dependent grammars for each language. This characteristic of SMT allows us to build translation systems for several language pairs using a universal architecture. In the following sections, we describe the basic mechanism of SMT and the method to evaluate the translation systems.

3.2.1 Noisy Channel Model

Weaver's idea of "using digital computers to translate documents between natural human languages" can be modeled as a noisy channel model [97], which was

originally proposed by Claude E. Shannon. A translation system that selects an appropriate translation $\boldsymbol{e}^\star$ (e.g., an English sentence) from all set of sentences $\bar{\boldsymbol{E}}$ given $\boldsymbol{f}$ (e.g., a French sentence) can be formulated as:

$$\boldsymbol{e}^\star = \underset{\boldsymbol{e}\in\boldsymbol{E}}{\operatorname{argmax}} P(\boldsymbol{e}|\boldsymbol{f}). \tag{3.1}$$

Using Bayes' theorem, this equation can be rewritten as:

$$\boldsymbol{e}^\star = \underset{\boldsymbol{e}\in\boldsymbol{E}}{\operatorname{argmax}} \frac{P(\boldsymbol{f}|\boldsymbol{e})P(\boldsymbol{e})}{P(\boldsymbol{f})} \tag{3.2}$$

$$= \underset{\boldsymbol{e}\in\boldsymbol{E}}{\operatorname{argmax}} P(\boldsymbol{f}|\boldsymbol{e})P(\boldsymbol{e}) \tag{3.3}$$

Here, since the denominator $P(\boldsymbol{f})$ in Eq. (3.2) is a constant, the problem of translation is reduced to the maximization problem of Eq. (3.3). In the context of machine translation, we call $P(\boldsymbol{f}|\boldsymbol{e})$ a translation model and $P(\boldsymbol{e})$ a language model. While the translation model represents the adequacy of translation, the language model captures the fluency of the sentence $\boldsymbol{e}$. SMT systems output correct and fluent translations by maximizing the product of the two models.

3.2.2 Language Model

As described in the previous section, the language model encourages SMT systems to output a fluent sentence $\boldsymbol{e}$. That is, the language model assigns higher scores to the sentences which (1) consist of appropriate words and (2) are grammatically correct. Three examples of translation candidates are shown below:

- $\boldsymbol{e_1}$: I drink coffee.
- $\boldsymbol{e_2}$: I coffee drink.
- $\boldsymbol{e_3}$: I am coffee.

Comparing the three sentences above, we can find that $\boldsymbol{e_1}$ consists of appropriate words and grammatically correct. While $\boldsymbol{e_2}$ is grammatically incorrect, $\boldsymbol{e_3}$ has inappropriate words. The role of the language model is to assign a higher translation probability to translation candidates such as $\boldsymbol{e_1}$, and help the system to output correct sentences.

An accurate language model can be obtained by training from a massive monolingual document set (hereafter *corpus*). For example, if a sentence "I drink coffee."

appears 200 times in a corpus consists of one million sentences, the generation probability can be computed as

$$P(\boldsymbol{e} = \text{I drink coffee.}) = 0.0002.$$

However, this naive model suffers from the data sparseness problem. Since long sentences are not likely to exist in training corpus, this probability model often outputs 0 given a long sentence. N-gram language model is widely used in the area of natural language processing to cope with this problem. An n-gram[1] is a sequence of n words that appears in the corpus. Instead of computing the probability of the whole sentence, the n-gram language model computes the probability of a sequence of words that consists of the sentence. For example, the sentence probability described above can be rewritten as

$$\begin{aligned} &P(\boldsymbol{e} = \text{I drink coffee.}) \\ =&P(\langle s\rangle, \text{I}, \text{drink}, \text{coffee}, \langle /s\rangle), \end{aligned} \tag{3.4}$$

where the $\langle s\rangle$ and the $\langle /s\rangle$ represent the beginning and the end of a sentence, respectively. By applying chain rule, this equation can be rewritten as a product of the conditional probabilities as

$$\begin{aligned} &P(\langle s\rangle, \text{I}, \text{drink}, \text{coffee}, \langle /s\rangle) \\ &= P(e_1 = \text{I} | e_0 = \langle s\rangle) \\ &\times P(e_2 = \text{drink} | e_0 = \langle s\rangle, e_1 = \text{I}) \\ &\times P(e_3 = \text{coffee} | e_0 = \langle s\rangle, e_1 = \text{I}, e_2 = \text{drink}) \\ &\times P(e_4 = \langle /s\rangle | e_0 = \langle s\rangle, e_1 = \text{I}, e_2 = \text{drink}, e_3 = \text{coffee}). \end{aligned} \tag{3.5}$$

This can be approximated by the n-gram language model by assuming that the generation of words is only conditioned by the previous $n-1$ words, instead of considering all the preceding words as:

$$P(e_n | e_1^{n-1}) \approx P(e_n | e_{n-N+1}^{n-1}) \tag{3.6}$$

[1]The terms unigram ($n = 1$), bigram ($n = 2$), trigram($n = 3$) are used when n is smaller than 4. For $n \geqq 4$, it is referred to as 4-gram, 5-gram, and so on.

For example, if N is set to 2, the conditional probability in Eq. 3.5 can be approximated as:

$$\begin{aligned}
&P(\langle s\rangle, \mathrm{I}, \mathrm{drink}, \mathrm{coffee}, \langle /s\rangle) \\
&= P(e_1 = \mathrm{I} | e_0 = \langle s\rangle) \\
&\times P(e_2 = \mathrm{drink} | e_1 = \mathrm{I}) \\
&\times P(e_3 = \mathrm{coffee} | e_2 = \mathrm{drink}) \\
&\times P(e_4 = \langle /s\rangle | e_3 = \mathrm{coffee}).
\end{aligned} \tag{3.7}$$

Here, note that the number of the $\langle s\rangle$ tokens k increases as N becomes larger ($k = N-1$). This approximation allows us to assign appropriate probabilities to longer sentences even if we cannot observe the same sentence in the training corpus.

3.2.3 Phrase-based Statistical Machine Translation

Most of the currently used SMT systems are based on the Phrase-based SMT [51] model. Phrase-based SMT consists of (1) a phrase translation model that translates the phrases in the source sentence into target language, and (2) a distortion model that reorders the translated phrases into a correct order. The translation is computed with the models as:

$$\begin{aligned}
\boldsymbol{e}^\star &= \operatorname*{argmax}_{\boldsymbol{e}\in\boldsymbol{E}} P(\boldsymbol{f}|\boldsymbol{e})P(\boldsymbol{e}) \\
&= \operatorname*{argmax}_{\boldsymbol{e}\in\boldsymbol{E}} \sum_{\boldsymbol{\phi},\boldsymbol{\alpha}} P(\boldsymbol{f},\boldsymbol{\phi},\boldsymbol{\alpha}|\boldsymbol{e})P(\boldsymbol{e}) \\
&\approx \operatorname*{argmax}_{\boldsymbol{e}\in\boldsymbol{E}} \sum_{\boldsymbol{\phi},\boldsymbol{\alpha}} P(\boldsymbol{f},\boldsymbol{\alpha}|\boldsymbol{\phi})P(\boldsymbol{\phi}|\boldsymbol{e})P(\boldsymbol{e})
\end{aligned} \tag{3.8}$$

Here, a latent variable $\boldsymbol{\alpha}$ is a vector that represents the order of the phrases. A probability distribution $\boldsymbol{\phi}$ models the phrase translation. The right term of Eq. 3.8 denotes the translation process as followings. First, a target sentence $\boldsymbol{e}$ is generated from the language model $P(\boldsymbol{e})$. Next, it is translated by the translation model $P(\boldsymbol{\phi}|\boldsymbol{e})$. Finally, the translated phrases are reordered by the distortion model $P(\boldsymbol{f},\boldsymbol{\alpha}|\boldsymbol{\phi})$ to obtain the source sentence $\boldsymbol{f}$.

Given a set of hand-made translation pair $\langle \boldsymbol{f},\boldsymbol{e}\rangle$, we can train the translation model and the distortion model by maximizing the likelihood of $P(\boldsymbol{f}|\boldsymbol{e})$. After training the models, the translation of a source sentence $\boldsymbol{f}$ can be reduced to a problem of

selecting the best $\boldsymbol{e}^{\star}$ from the set of all target sentences $\boldsymbol{E}$ by solving the Eq. 3.8. However, it is impractical to generate all target sentences in $\boldsymbol{E}$. In order to reduce the search space, the practical systems usually set constraints to limit the number of reordering patterns.

Several researchers have been working on phrase-based SMT and many proposed methods are implemented on Moses,[2] an open-sourced SMT project. All experiments on SMT in this thesis are also conducted with Moses toolkit.

3.2.4 Evaluation Methods for Machine Translation

To evaluate a machine translation system, subjective evaluation and automatic evaluation can be conducted. Since the purpose of machine translation is to help human understand foreign language, the subjective evaluation conducted by human is important and reasonable. However, human evaluation has problems such as: (1) evaluation results by different evaluators may vary, (2) evaluation procedure is difficult to be reproduced, and (3) it cost much and is time-consuming[108, pp.47–57]. Since the statistical models in SMT systems have lots of parameters to be tuned, an easy, fast, and cheap evaluation method had been required.

The Bilingual Evaluation Understudy (hereafter BLEU) [88] is the most commonly used automatic evaluation method for machine translation. BLEU is a precision based metric that measures the ratio of n-gram overlap between the system output and reference translations as:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right), \tag{3.9}$$

where p_n represent the precision of n-gram in the system outputs. Here, the BLEU has a problem that high p_n can be easily achieved by only generating the words with high confidence. To cope with this problem, the Brevity penalty (BP) is introduced to avoid the system output being extremely short:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r, \end{cases} \tag{3.10}$$

where c and r represent the number of words in system output and the reference, respectively.

[2] http://www.statmt.org/moses/

3.3 Proposed: Accurate Cross-lingual Projection of Word Semantic Representations

There exist substantial differences between distributions of words in documents when they are in different domains. The differences will make a sentence whose domain is different from that of training data tends to be difficult to be translated.

Our method exploits a projection of semantic representations of oov words in the source-language onto the target-language semantic space to look for translation candidates for the oov words. In this section, we first introduce semantic representations of words in a continuous vector space. And then, we propose a method that accurately learns a translation matrix for projecting vector-based representations of words across languages.

3.3.1 Learning Cross-lingual Projection between Vector Representations of Words

A vector-based semantic representation of a word, hereinafter *word vector*, represents the meaning of a word with a continuous vector. These representations are based on the distributional hypothesis [37, 29], which states that words that occur in the similar contexts tend to have similar meanings. The word vectors can be obtained from monolingual corpora in an unsupervised manner, such as a count-based approach [59] or prediction-based approaches [8, 68].

The words that have similar meanings tend to have similar vectors [106, 26]. By mapping words into a continuous vector space, we can use cosine similarity to compute the similarity of meanings between words. However, the similarity between word vectors across languages is difficult to compute, so these word vectors are difficult to utilize in cross-lingual applications such as machine translation or cross-lingual information retrieval.

To solve this problem, Mikolov et al. [70] proposed a method that learns a cross-lingual projection of word vectors from one language into another. By projecting a word vector into the target-language semantic space, we can compute the semantic similarity between words in different languages. Suppose that we have training data of n examples, $\{(\boldsymbol{x}_1, \boldsymbol{z}_1), (\boldsymbol{x}_2, \boldsymbol{z}_2), \ldots (\boldsymbol{x}_n, \boldsymbol{z}_n)\}$, where $\boldsymbol{x}_i$ is the vector representation of a word in the source language (*e.g.*, "*gato*"), and $\boldsymbol{z}_i$ is the word vector of its translation in the target language (*e.g.*, "*cat*"). Then the translation matrix, $\boldsymbol{W}$, such that $\boldsymbol{W}\boldsymbol{x}_i$

approximates $\boldsymbol{z}_i$, can be obtained by solving the following optimization problem:

$$\boldsymbol{W}^{\star} = \operatorname*{argmin}_{\boldsymbol{W}} \sum_{i=1}^{n} \|\boldsymbol{W}\boldsymbol{x}_i - \boldsymbol{z}_i\|^2.$$

3.3.2 Exploiting Translatable Context Pairs

Within the learning framework above, we propose exploiting the fact that dimensions of count-based word vectors are associated with context words, and some dimensions in the source language are translations of those in the target language.

For an illustration purpose, suppose count-based word vectors of Spanish and English. The Spanish word vectors would have dimensions associated with context words such as "*amigo*," "*comer*," "*importante*," while the dimensions of the English word vectors are associated with "*eat*," "*run*," "*small*" and "*importance*," and so on. Since, for example, "*friend*" is an English translation of "*amigo*," the Spanish dimension associated with "*amigo*" is likely to be mapped to the English dimension associated with "*friend*." Such knowledge about the cross-lingual correspondence between dimensions is considered beneficial for learning an accurate translation matrix.

We take two approaches to obtaining such correspondence. Firstly, since we have already assumed that a small amount of training data is available for training the translation matrix, it can also be used for finding the correspondence between dimensions (referred to as $\mathcal{D}_{train}$). Note that it is natural that some words in a language have many translations in another language. Thus, for example, $\mathcal{D}_{train}$ may include ("*amigo*", "*friend*"), ("*amigo*", "*fan*") and ("*amigo*", "*supporter*").

Secondly, since languages have evolved over the years while often deriving or borrowing words (or concepts) from those in other languages, those words have similar or even the same spelling. We take advantage of this to find the correspondence between dimensions. We specifically define function $\textsc{dist}(r,s)$ that measures the surface-level similarity, and regard all context word pairs (r,s) having smaller distance than a threshold[3] as translatable ones (referred to as $\mathcal{D}_{sim}$).

$$\textsc{dist}(r,s) = \frac{\mathit{Levenshtein}(r,s)}{\min(\mathit{len}(r), \mathit{len}(s))} \tag{3.11}$$

where function $\mathit{Levenshtein}(r,s)$ represents the Levenshtein distance between the two words, and $\mathit{len}(r)$ represents the length of the word.

[3]The threshold was fixed to 0.5.

3.3.3 Modified Objective Function

We incorporate the knowledge about the correspondence between the dimensions into the learning framework. Since the correspondence obtained by the methods presented above can be noisy, we want to treat it as a soft constraint. This consideration leads us to develop the following new objective function:

$$\boldsymbol{W}^{\star} = \operatorname*{argmin}_{\boldsymbol{W}} \sum_{i=1}^{n} \|\boldsymbol{W}\boldsymbol{x}_i - \boldsymbol{z}_i\|^2 + \frac{\lambda}{2}\|\boldsymbol{W}\|^2 - \beta_{train} \sum_{(j,k)\in\mathcal{D}_{train}} w_{jk} - \beta_{sim} \sum_{(j,k)\in\mathcal{D}_{sim}} w_{jk}.$$

The second term is the L_2 regularizer, while the third and fourth terms are meant to strengthen w_{jk} when k-th dimension in the source language corresponds to j-th dimension in the target language. $\mathcal{D}_{train}$ and $\mathcal{D}_{sim}$ are sets of translatable dimension pairs. $\mathcal{D}_{train}$ is obtained from the above training data, while $\mathcal{D}_{sim}$ is obtained by computing the surface-level similarity between the dimensions. λ, β_{train} and β_{sim} are corresponding hyperparameters to control the strength of the added terms.

3.3.4 Optimization

We use the Pegasos algorithm [96], an instance of the stochastic gradient descent [9], to optimize the new objective. Given the τ-th learning sample $(\boldsymbol{x}_\tau, \boldsymbol{z}_\tau)$, we update translation matrix $\boldsymbol{W}$ as follows:

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta_\tau \nabla E_\tau(\boldsymbol{W}) \tag{3.12}$$

where η_τ represents the learning rate and is set to $\eta_\tau = \frac{1}{\lambda\tau}$, and $\nabla E_\tau(\boldsymbol{W})$ is the gradient which is calculated from τ-th sample $(\boldsymbol{x}_\tau, \boldsymbol{z}_\tau)$:

$$2(\boldsymbol{W}\boldsymbol{x}_\tau - \boldsymbol{z}_\tau)\boldsymbol{x}_\tau^{\mathrm{T}} - \beta_{train}\boldsymbol{A} - \beta_{sim}\boldsymbol{B} + \lambda\boldsymbol{W}. \tag{3.13}$$

$\boldsymbol{A}$ and $\boldsymbol{B}$ are gradients corresponding to the two new terms. $\boldsymbol{A}$ is a matrix in which $a_{jk} = 1$ if $(j,k) \in \mathcal{D}_{train}$ otherwise 0. $\boldsymbol{B}$ is defined similarly.

3.4 Proposed: Instant Translation Model Adaptation for Statistical Machine Translation

In the previous section, we proposed an accurate cross-lingual projection method for word vectors. Since this method does not require any parallel corpora, it can be naturally applied to domain adaptation pipeline for SMT, where we cannot access any in-domain parallel corpora.

In this section, we propose an instant translation model for SMT using the projection of word vectors. Our method assumes that monolingual corpora are available for the source and target language (in the target domain, if any) and first induces word vectors from those corpora. It then learns a cross-lingual projection (translation matrix) using a seed dictionary in a general domain as described in Section 3.3. Note that a seed dictionary for common words is usually available for most pairs of languages or could be constructed assuming English as a pivot language [105].

Having a translation matrix to obtain projections of semantic representations of oov words in a given sentence, our method instantly constructs a back-off translation model used for enumerating translation candidates for the oov words in the following way:

Step 1: When the translation system accepts a sentence with an oov word, f_{oov}, it translates a semantic representation of the word, $\boldsymbol{x}_{\text{oov}}$ into a semantic representation in the target language $\boldsymbol{x}'_{\text{oov}}$ using the translation matrix obtained by the method described in Section 3.3.

Step 2: It then computes the cosine similarity between the obtained semantic representations with those in the target languages to enumerate k translation candidates[4] in accordance with the value of cosine similarity. The cosine similarity is also used to obtain $P_{vec}(e|f_{\text{oov}})$, the direct translation probabilities from the oov word in the source language, f_{oov}, to a candidate word in the target language, e, by normalizing them to sum up to 1. Although the obtained translation candidates could include wrong translations, the language model can choose one that is more appropriate in the contexts in the next step, unless the contexts are full of oov words.

Step 3: The decoder of phrase-based SMT uses the above translation probabilities as a back-off translation model to perform the translation. More formally, we

[4] k was set to 10 in the experiments.

add a new feature function h_{vec} to the log-linear model used in the decoder as follows:

$$\log P(\boldsymbol{e}|\boldsymbol{f}) = \sum_i \log(h_i(\boldsymbol{e},\boldsymbol{f}))\lambda_i + \log(h_{vec}(\boldsymbol{e},\boldsymbol{f}))\lambda_{vec} \tag{3.14}$$

The $h_{vec}(\boldsymbol{e},\boldsymbol{f})$ in Eq. (3.14) is computed with $P_{vec}(e|f_{\text{oov}})$, only for each oov word f_{oov} in source sentence $\boldsymbol{f}$. An issue here is how to set feature weight λ_{vec} since no in-domain training data are available for turning. We simply set λ_{vec} to the same value as the weight of direct phrase translation probability of the translation model.

3.5 Experiments: Cross-lingual Projection of Word Semantic Reprensentations

In the previous sections, we proposed two modules to realize an accurate and instant domain adaptation for SMT. In this section, we evaluate the first module: the accurate cross-lingual projection of word vectors. We perform vector translation experiments between four languages: English (En), Spanish (Es), Japanese (Ja) and Chinese (Zh) so that we can examine the impact of each type of translatable context pairs integrated into the learning objective.

3.5.1 Settings

First, we prepared source text in the four languages from Wikipedia[5] dumps following [7]. We extracted plain text from the XML dumps by using wp2txt.[6] Since words are concatenated in Japanese and Chinese, we used MeCab[7] and Stanford Word Segmenter[8] to tokenize the text. Since inflection occurs in English, Spanish, and Japanese, we used Stanford POS tagger,[9] Pattern,[10] and MeCab to lemmatize the text.

Next, we induced count-based word vectors from the obtained text. We considered context windows of five words to both sides of the target word. The function words are then excluded from the extracted context words. Since the count vectors are very

[5] http://dumps.wikimedia.org/
[6] https://github.com/yohasebe/wp2txt/
[7] http://taku910.github.io/mecab/
[8] http://nlp.stanford.edu/software/segmenter.shtml
[9] http://nlp.stanford.edu/software/tagger.shtml
[10] http://www.clips.ua.ac.be/pages/pattern

Table 3.1: Experimental results: the accuracy of the translation.

Lang. pairs	**Baseline**		**CBOW**		**Direct Mapping**		**Proposed$_{\text{w/o surface}}$**		**Proposed**	
	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
Ja → Zh	0.6%	1.6%	5.4%	13.8%	9.3%	22.2%	11.1%	26.2%	**15.5%**	**34.0%**
Zh → Ja	0.3%	1.2%	2.9%	11.3%	11.6%	26.9%	7.8%	21.6%	**13.1%**	**27.9%**
Ja → En	0.2%	1.0%	6.5%	19.1%	22.3%	37.4%	32.3%	51.0%	**32.5%**	**51.9%**
En → Ja	0.3%	1.1%	4.9%	13.3%	5.4%	13.9%	18.5%	36.4%	**19.3%**	**37.1%**
Zh → En	0.2%	0.8%	3.4%	11.8%	**23.3%**	40.6%	22.3%	40.4%	23.1%	**42.0%**
En → Zh	0.2%	1.1%	5.1%	13.7%	4.5%	11.8%	9.1%	22.1%	**9.5%**	**23.0%**
En → Es	0.2%	1.0%	7.1%	18.9%	11.9%	26.1%	28.7%	45.7%	**31.3%**	**49.6%**
Es → En	0.0%	0.6%	7.5%	22.0%	45.7%	61.1%	46.6%	62.4%	**54.7%**	**67.6%**

high-dimensional and sparse, we selected top-10k frequent words as contexts words (in other words, the number of dimensions of the word vectors). We converted the counts into positive point-wise mutual information [16] and normalized the resulting vectors to remove the bias that is introduced by the difference of the word frequency.

Then, we compiled a seed bilingual dictionary (a set of bilingual word pairs) for each language pair that is used to learn and evaluate the translation matrix. We utilized cross-lingual synsets in the Open Multilingual Wordnet[11] to obtain bilingual pairs.

Since our method aims to be used in expanding bilingual dictionaries, we designed datasets assuming such a situation. Considering that more frequent words are likely to be registered in a dictionary, we sorted words in the source language by frequency and used the top-11k words and their translations in the target language as a training/development data, and used the subsequent 1k words and their translations as the test data. Here, if there exist polysemous words, we extract all of them as independent examples. Thus, as shown in Table 3.2, the vocabulary sizes in target side and the amount of training data are always larger than 10k. Since each target word is not always correspond to a single source word, the size of $\mathcal{D}_{train}$ can also be larger than 10k. Note that the Table 3.2 does not include the (Ja → Es), (Es → Ja), (Zh → Es), and (Es → Zh). We cannot extract the bilingual dictionaries for these language pairs because there are no large parallel corpora for these pairs included in Open Multilingual Wordnet.

We have compared our method with the following three methods:

[11] http://compling.hss.ntu.edu.sg/omw/

Table 3.2: Vocabulary size, the amount of the training data, and the translatable context pairs $\mathcal{D}_{train}$ and $\mathcal{D}_{sim}$.

	Vocab. size (source)	Vocab. size(target)	# training data	$\mathcal{D}_{train}$	$\mathcal{D}_{sim}$
(Ja → Zh)	10,000	10,641	42,037	9,552	3,189
(Zh → Ja)	10,000	20,356	69,619	9,552	3,189
(Ja → En)	10,000	15,060	50,300	18,296	2,234
(En → Ja)	10,000	28,275	84,451	18,296	2,234
(Zh → En)	10,000	15,784	41,144	9,292	3,551
(En → Zh)	10,000	14,770	38,854	9,292	3,551
(En → Es)	10,000	10,247	34,034	15,567	12,764
(Es → En)	10,000	19,917	48,125	15,567	12,764

Baseline learns a translation matrix using Eq. 3.3.1 for the same count-based word vectors as the proposed method. Comparison between the proposed method and this method reveals the impact of incorporating the cross-lingual correspondences between dimensions.

CBOW learns a translation matrix using Eq. 3.3.1 for word vectors learned by a neural network (specifically, continuous bag-of-words (CBOW)) [70]. Comparison between this method and the above baseline reveals the impact of the vector representation. Note that the CBOW-based word vectors take rare context words as well as the top-10k frequent words into account. We used word2vec[12] to obtain the vectors for each language.[13] Since Mikolov et al. [70] reported the accurate translation can be obtained when the vectors in the source language is 2-4x larger than that in the target language, we prepared m-dimensional ($m = 100, 200, 300$) vectors for the target language and n-dimensional ($n = 2m, 3m, 4m$) vectors for the source language, and optimized their combinations on the development data.

Direct Mapping exploits the training data to map each dimension in a word vector in the source language to the corresponding dimension in a word vector in the target language, referring to the bilingual pairs in the training data [31]. To deal with words that have more than one translation, we weighted each translation by a reciprocal rank of its frequency among the translations in the target language, as in [91].

[12] https://code.google.com/p/word2vec/

[13] The threshold of sub-sampling of words was set to 1e-3 to reduce the effect of very frequent words, *e.g.*, "*a*" or "*the*."

Note that all methods, including the proposed methods, use the same amount of supervision (training data) and thereby they are completely comparable with each other.

Evaluation procedure

For each word vector in the source language, we translate it into the target language and evaluate the quality of the translation as in [70]: i) measure the cosine similarity between the resulting word vector and all the vectors in the test data (in the target language), ii) next choose the top-n (n = 1,5) word vectors that have the highest similarity against the resulting vector, and iii) then examine whether the chosen vectors include the correct one.

3.5.2 Results

Overall performances

Table 3.9 shows results of the translation between word vectors in each language pair. **Proposed** significantly improved the translation quality against **Baseline**, and performed the best among all of the methods. Although the use of CBOW-based word vectors (**CBOW**) has improved the translation quality against **Baseline**, the performance gain is smaller than that obtained by our new objective. **Proposed$_{\text{w/o surface}}$** uses only the training data to find translatable context pairs by setting $\beta_{sim} = 0$. Thus, its advantage over **Direct Mapping** confirms the importance of learning a translation matrix. On the other hand, in (Zh → Ja) and (Zh → En) translations, **Direct Mapping** performs even better than **Proposed$_{\text{w/o surface}}$**. Also, there is no clear improvement in (Ja → Zh) and (En → Zh). The amount of $\mathcal{D}_{train}$ in Table 3.2 provides the reason for these phenomena. We can find that the sizes of $\mathcal{D}_{train}$ in (Ja, En) or (En, Es) are 1.6x to 2x larger than that of (Ja, Zh) and (Zh, En). This data suggests that if we have enough size of bilingual dictionary, $\mathcal{D}_{train}$ contributes much in improving the translation performance.

In addition, the greater advantage of **Proposed** over **Proposed$_{\text{w/o surface}}$** in the translation between (En, Es) or (Ja, Zh) conforms to our expectation that surface-level similarity is more useful for translation between the language pairs which have often exchanged their vocabulary. Note that the performances of different language pairs cannot be compared. This is because the sizes of the Wikipedia corpora, on which we trained the word vectors, are significantly different.

Table 3.3: The performance changes by introducing $\mathcal{D}_{sim}$ in optimization. There exist 1,000 of test data in total.

	incorrect → correct	**correct → incorrect**
(Ja → Zh)	53	9
(Zh → Ja)	60	7
(Ja → En)	9	7
(En → Ja)	11	3
(Zh → En)	14	6
(En → Zh)	8	4
(En → Es)	52	26
(Es → En)	108	27

To further analyze the effect of the surface similarity of context words on the translation matrix, we show the performance changes due to the $\mathcal{D}_{sim}$ in Table 3.3. We have already discussed $\mathcal{D}_{sim}$ helps translation in (Ja, Zh) and (En, Es). The table shows that only the pairs of (En, Es) have performance deterioration (correct → incorrect) in several examples, while all the language pairs have improved examples (incorrect → correct) This result suggests that the parameters for $\mathcal{D}_{sim}$ learn to ignore minor mistakes and biases the model more aggressively when translating (En, Es).

Impact of the size of training data

Figure 3.1, 3.3, and 3.4 show precision@1s plotted against the size of training data. Remember that the training data is not only used to learn a translation matrix in the methods other than **Direct Mapping** but also is used to map dimensions in **Direct Mapping** and the proposed methods. **Proposed** performs the best among all methods regardless the size of training data. Comparison between **Direct Mapping** and $\textbf{Proposed}_{\textbf{w/o surface}}$ reveals that learning a translation matrix is not always effective when the size of the training data is small, since it may suffer from over-fitting (the size of the translation matrix is too large for the size of training data). We can see that surface-level similarity is beneficial especially when the size of training data is small. Let us focus on (Ja, Zh) and (En, Es), the language pairs whose surface forms are similar to each other. We can find that the **Proposed** outperforms $\textbf{Proposed}_{\textbf{w/o surface}}$ significantly if the size of training data is large. This result indicates that $\mathcal{D}_{sim}$, the clues of surface forms, helps accurate training of the translation matrix.

Table 3.4: Top-5 translations in (Zh → Ja)

	Baseline	**CBOW**	**Direct Mapping**	**Proposed$_{(w/o\ surface)}$**	**Proposed**
校験位 → パリティ /パリティビット/パリティー					
#1	違う	考慮	プリミティブ	縫い目	パリティビット
#2	動く	相	クライアント	言葉	パリティ
#3	持つ	把握	結び目	パリティ	縫い目
#4	周囲	規準	用語	正しい	クライアント
#5	十分	正しい	ディレクトリ	一見	言葉
焼瓶 → フラスコ					
#1	周囲	卵殻	空気	フラスコ	フラスコ
#2	軽い	微粒子	フラスコ	空気	空気
#3	動く	フラスコ	溶解	磨る	活栓
#4	小さい	小片	滴	寒天	磨る
#5	持つ	薄片	寒天	天井	寒天
小農 → 小作農					
#1	周囲	変質	小作農	困窮	困窮
#2	見る	溜め込む	困窮	把握	保護
#3	現れる	無駄	配慮	好ましい	把握
#4	かなり	不潔	把握	配慮	小作農
#5	動く	腐敗	作物	保護	配分

Qualitative analysis

We show the translation examples in Table 3.13, Table 3.5, and Table 3.6. The bolded characters in the tables represent the correct answers. In all three language pairs, **Baseline** outputs similar translations. This phenomenon has reported as Hubness problem [54], which is caused by the ridge regression on high-dimensional spaces [98].

Using **CBOW** provides improvement from **Baseline**. For example, in the (En → Ja) translation task, there are "化け物" and "生霊" as translation candidates of "sorceress", and "微粒子" in the candidates of "xenon". Although the CBOW model helps induce similar/related words as candidates, their # 1 candidates are still not the correct translations.

There are lots of common outputs when comparing **Direct Mapping** and **Proposed$_{(w/o\ surface)}$**. This is because both of them use the bilingual dictionary to map the vectors. However, their ways of utilizing the bilingual information are different. While the former uses a bilingual dictionary to directly map the vectors to other language without any learning methods, the latter captures the information as an additional term in the objective function.

Table 3.5: Top-5 translations in (En → Ja)

	Baseline	CBOW	Direct Mapping	Proposed$_{\text{(w/o surface)}}$	Proposed
sorceress → 魔 法使い/魔女					
# 1	思う	化け物	魔物	魔物	魔法使い
# 2	恐ろしい	生霊	恐ろしい	魔法使い	魔物
# 3	捨てる	狂気	魔法使い	呪い	魔女
# 4	怒る	暴君	思う	魔女	呪い
# 5	邪魔	人殺し	呪い	怪物	エルフ
xenon → キセノ ン					
# 1	逆	微粒子	気体	放射	キセノン
# 2	実際	蒸散	放射	キセノン	放射
# 3	ある程度	変化	粒子	気体	気体
# 4	弱い	吸い込む	特性	粒子	粒子
# 5	小さい	縮む	小さい	重力	重力
abduct → 連れ去る					
# 1	思う	追い払う	逃げる	連れ去る	襲う
# 2	捨てる	騙す	逃げ出す	襲う	連れ去る
# 3	恐ろしい	庇う	襲う	殺害	殺害
# 4	逃げる	責める	思う	殺し	殺し
# 5	怒る	見捨てる	恐ろしい	逃げ出す	逃げ出す

On top of the **Proposed$_{\text{(w/o surface)}}$** model, the **Proposed** can also capture the surface similarities of context words. By considering the surface similarities with $\mathcal{D}_{sim}$, it provides even better performance. As an example, let us compare the translation of "校験位" in (Zh → Ja). While **Proposed$_{\text{(w/o surface)}}$** output the correct answer "パリティ" as the # 3 candidate, the # 1 and 2 candidates of the **Proposed** are both correct. There are a few cases where **Proposed** performs worse than other methods, such as "小農" → "小作農" (Zn → Ja) and "yambo" → "iamb" (Es → En). However, more examples show the superiority of the **Proposed** over other models.

3.6 Experiments: Domain Adaptation for Statistical Machine Translation

In the previous section, we showed the effectiveness of our method for cross-lingual projection of word vectors. In this section, we apply the proposed projection method to a domain adaptation task for SMT to evaluate its effectiveness in a real world application.

Table 3.6: Top-5 translations in (Es → En)

	Baseline	CBOW	Direct Mapping	Proposed$_{(w/o\ surface)}$	Proposed
clericalismo → clericalism					
# 1	call	attitude	struggle	attitude	**clericalism**
# 2	describe	banality	attitude	negativity	attitude
# 3	intend	self-consciousness	turn	struggle	struggle
# 4	make	fatalistic	**clericalism**	**clericalism**	negativity
# 5	ignore	egoism	espouse	fatalistic	chauvinism
papio → baboon					
# 1	call	crab	elephant	cow	**baboon**
# 2	turn	dwarf	antelope	ichthyosaur	crocodile
# 3	make	elephant	cow	parcel	dwarf
# 4	describe	crocodile	parcel	elephant	elephant
# 5	intend	hairy	bovid	crocodile	obscure
yambo → iamb					
# 1	call	fairy	call	**iamb**	interrogative
# 2	turn	pluck	turn	caesura	stanza
# 3	describe	stick	stanza	interrogative	**iamb**
# 4	make	dark	set	gesture	caesura
# 5	intend	croak	**iamb**	stanza	gesture

3.6.1 Settings

First, we prepared two parallel corpora in different domains to carry out an experiment of domain adaptation in the SMT system. One is the "Japanese-English Bilingual Corpus of Wikipedia's Kyoto Articles" (hereinafter KFTT corpus), originally prepared by the National Institute of Information and Communications Technology (NICT) and used as a benchmark in "The Kyoto Free Translation Task"[14][78], a translation task that focuses on Wikipedia articles relates to Kyoto. The other parallel corpus (hereafter RECIPE corpus) is provided by Cookpad Inc.,[15] which is the largest online recipe sharing service in Japan. The KFTT corpus includes many words which are related to Japanese history and the temples or shrines in Kyoto. On the other hand, the RECIPE corpus includes many words related to foods and cookware. We randomly sampled 10k pairs of sentences from the RECIPE corpus as the test corpus for evaluating our domain adaption method. The language models of the target languages are trained with the concatenation of the KFTT corpus and the remaining portion of the RECIPE corpus, while the translation models are trained with only the KFTT corpus. The sizes of the training data and test data are as detailed in Table 3.7.

[14]http://www.phontron.com/kftt/
[15]http://cookpad.com/

Table 3.7: Statistics of the dataset.

Corpus	Japanese		English	
KFTT (training)	29.5MB	(440k sentences)	30.6MB	(440k sentences)
RECIPE (test)	0.8MB	(10k sentences)	0.7MB	(10k sentences)

Table 3.8: Monolingual corpora used to induce semantic representations.

Corpus	Japanese	English
Wikipedia (general domain)	4.4GB	16GB
RECIPE (in-domain)	12MB	9.5MB

We conducted experiments with Moses [49][16] with the language models trained with SRILM [101][17] and the word alignments predicted by GIZA++ [86].[18] 5-gram language models were trained using SRILM with `interpolate` option and `kndiscount` option. Word alignments were obtained using GIZA++ with `grow-diag-final-and` heuristic. The lexical reordering model was obtained with `msd-bidirectional` setting.

Next, we extracted four sets of count-based word vectors from Wikipedia dumps[19] (general-domain monolingual corpora) and the remaining portion of the RECIPE corpus (in-domain monolingual corpora), for Japanese and English, respectively. We considered context windows of five words to both sides of the target word. The function words are then excluded from the extracted context words as described in Section 3.5. Since the count vectors are very high-dimensional and sparse, we selected top-d ($d = 10,000$ for general-domain corpus, $d = 5000$ for in-domain corpus) frequent words as contexts words (in other words, the number of dimensions of the word vectors). We converted the counts into positive point-wise mutual information [16] and normalized the resulting vectors to remove the bias introduced by the difference in the word frequency. The size of the monolingual dataset for inducing semantic representations of words is as detailed in Table 3.8.

[16] http://www.statmt.org/moses/
[17] http://www.speech.sri.com/projects/srilm/
[18] https://github.com/moses-smt/giza-pp
[19] http://dumps.wikimedia.org/ (versions of Nov, 4th, 2014 (ja), Oct, 8th,2014 (en).

Table 3.9: BLEU on RECIPE corpus. * indicates statistically significant improvements in BLEU over the respective baseline systems in accordance with bootstrap resampling [47] at $p < 0.05$.

Method	All		oov sentences	
	ja-en	en-ja	ja-en	en-ja
Baseline (no adaptation)	5.58	3.37	5.36	3.16
Proposed (general-domain)	6.05*	3.48*	5.87*	3.42*
Proposed (in-domain)	**7.08***	**3.57***	**7.00***	**3.63***
Parallel Corpus	20.88	16.69	20.72	17.01

Table 3.10: Statistics of the oov words in test data (the 10k sentences in the RECIPE corpus).

	ja-en	en-ja
The number of oov words (types)	3,464	1,613
The number of oov words (tokens)	21,218	4,639
The number of sentences with oov words	8,742	3,636

Finally, we used Open Multilingual WordNet[20] to train the translation matrices as described in Section 3.3. The hyperparameters were tuned on the development set as follows: $\lambda = 0.1$, $\beta_{train} = 5$, $\beta_{sim} = 5$ for (ja-en, general-domain). $\lambda = 1$, $\beta_{train} = 0.1$, $\beta_{sim} = 0.2$ for (ja-en, in-domain). $\lambda = 0.1$, $\beta_{train} = 5$, $\beta_{sim} = 5$ for (en-ja, general-domain). $\lambda = 0.5$, $\beta_{train} = 1$, $\beta_{sim} = 2$ for (en-ja, in-domain).

3.6.2 Results

Overall results of domain adaptation

We performed domain adaptation as described in Section 3.4 and evaluated the effectiveness of our method through BLEU [88] score. Table 3.9 shows results of the translations of the 10k sentences in the RECIPE corpus between Japanese and English. **All** and **oov sentences** in Table 3.9 show the BLEU scores measured in the whole test set and the scores measured only in the sentences that include oov words, respectively. Statistics of the oov words are shown in Table 3.10.

[20] http://compling.hss.ntu.edu.sg/omw/

Table 3.11: BLEU on in-domain experiments with KFTT corpus.

Method	All		oov sentences	
	ja-en	en-ja	ja-en	en-ja
Baseline (no adaptation)	20.88	16.69	12.57	10.80
Proposed (in-domain)	20.88	16.68	12.53	10.77

Table 3.12: Statistics of the oov words in in-domain setting. Note that the test data used here is exactly the same as Table 3.10, while the training data is different.

	ja-en	en-ja
The number of oov words (types)	1,122	935
The number of oov words (tokens)	1,190	1,015
The number of sentences with oov words	1,002	870

All four methods shown in Table 3.9 use translation models that were trained with the KFTT corpus and are tested with the RECIPE corpus. **Proposed (general)** uses the word vectors extracted from Wikipedia corpus, while **Proposed (in-domain)** uses the vectors extracted from the remaining portion of the RECIPE corpus. In both these methods, we performed domain adaptation by automatically constructing back-off translation models for oov words. **Parallel Corpus** in Table 3.9 uses the remaining portion of the RECIPE corpus as a parallel corpus to learn the translation models, resources of which are assumed to be unavailable in this study. Thus, **Parallel Corpus** is the upper-bound for the task. The low BLEU score for en-ja translation is explained by the direction of the translation being different from the direction when the corpus was built (ja-en) [55]. In addition, the smaller number of oov tokens in en-ja than in ja-en also causes the smaller improvement in BLEU score.

Table 3.9 shows that our methods perform well for the translation task. We found that it was better to use the in-domain monolingual corpora rather than general-domain monolingual corpora to obtain the word vectors. This conforms to our expectation because the contextual information included in the word vectors strongly correlates with the target domains. The **Parallel Corpus** has much higher BLEU than all other methods. This result shows that the domain adaptation task we performed was intrinsically difficult because of the significant differences between the two domains.

Limitation of the proposed method

To illustrate the limitation of our method, we conduct an additional experiment in an in-domain scenario. As the baseline, we first use the RECIPE corpus to train and test the translation models. On top of this baseline, we perform the above oov translation method to translate the unknown words. The results in Table 3.11 show that there is no significant improvement with the **Proposed** method in this setting. Table 3.12 shows the statistics of oov words in the RECIPE corpus in this setting. Comparing with the out-of-domain setting (Table 3.10), we can find that the amount of oov words is extremely small in this in-domain setting. Since our adaptation method focuses on oov words, it does not improve the translation performance if there are only a few unknown words in the test set.

Qualitative analysis

We show hand-picked examples of the translations in Table 3.13 to analyze the methods in more detail. The first two examples show that **Proposed (in-domain)** provides more accurate translations than **Proposed (general)**. Despite our method being able to improve the translations of oov words, the third and the fourth examples indicate that it is not good at improving the translations of **Baseline** that have wrong syntax. The last example shows that some oov words tend to be translated into their related words, mainly because of their similarity in the semantic space.

The examples show that the oov words such as "煮る" (simmer), "トースター" (toaster), and "焼く" (bake) could successfully be translated with **Proposed (in-domain)**. These words almost never appear in the KFTT corpus, since they do not have any relation with Japanese history or the temples in Kyoto. By comparing **Proposed (in-domain)** and **Proposed (general)**, we see that the latter method translated many oov words into related words (e.g., "トースター" (toaster) to "refrigerator", or "煮る" (simmer) to "boil") by mistake. This result also indicates that the word vectors extracted from the in-domain corpus will work better than the vectors extracted from the general-domain corpus.

3.7 Chapter Summary

In this chapter, we presented a practical domain adaptation method for SMT. The key idea of the adaptation method is to leverage a cross-lingual projection of word semantic representations to obtain a translation model for out-of-vocabulary words in

SMT. Assuming monolingual corpora for the source and target languages, we induce vector-based semantic representations of words and obtain a projection (translation matrix) from source-language semantic representations into the target-language semantic space. The first contribution of this work is to propose an accurate method to induce the translation matrix. Our method exploits the translatable context pairs (which can be easily obtained from bilingual dictionaries or by computing Levenshtein distance) to train the translation matrix. In the experiments on word translation task between four languages (including English, Spanish, Japanese, and Chinese), our method outperformed the previous approaches by +8.1 points in averaged Precision.

The second contribution of this work is to present a method to find translation candidates of oov words using the aforementioned method for vector projection. We adopt the cosine similarity to induce the translation probability, which can be used as a back-off translation model only for the oov words. Experimental results on domain adaptation from a Kyoto-related domain to a recipe domain confirmed that our method improved BLEU by 0.5-1.5 and 0.1-0.2 for en-ja and ja-en translations, respectively.

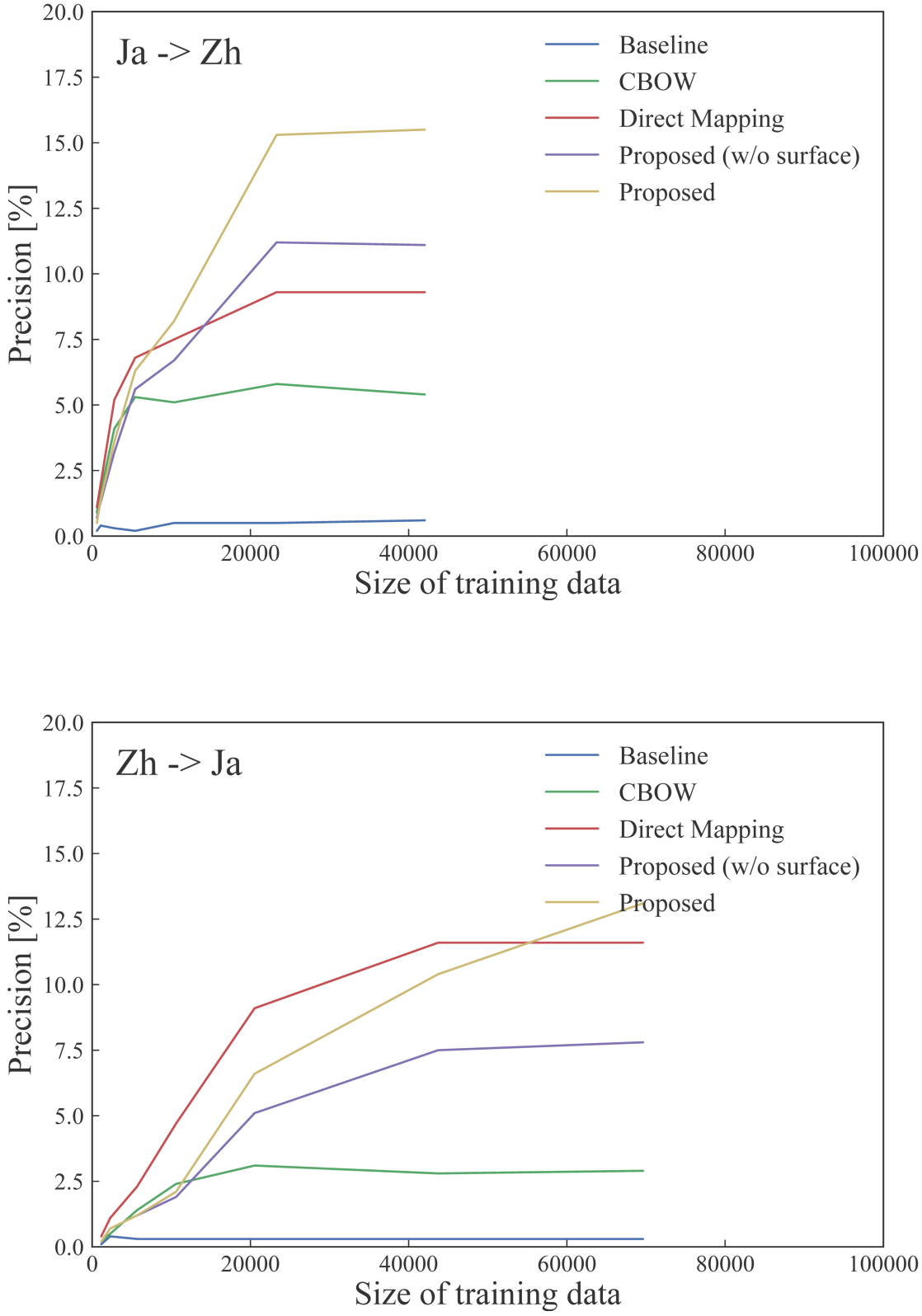


Figure 3.1: Impact of the size of training data. (Upper: (Ja → Zh), Bottom: (Zh → Ja)

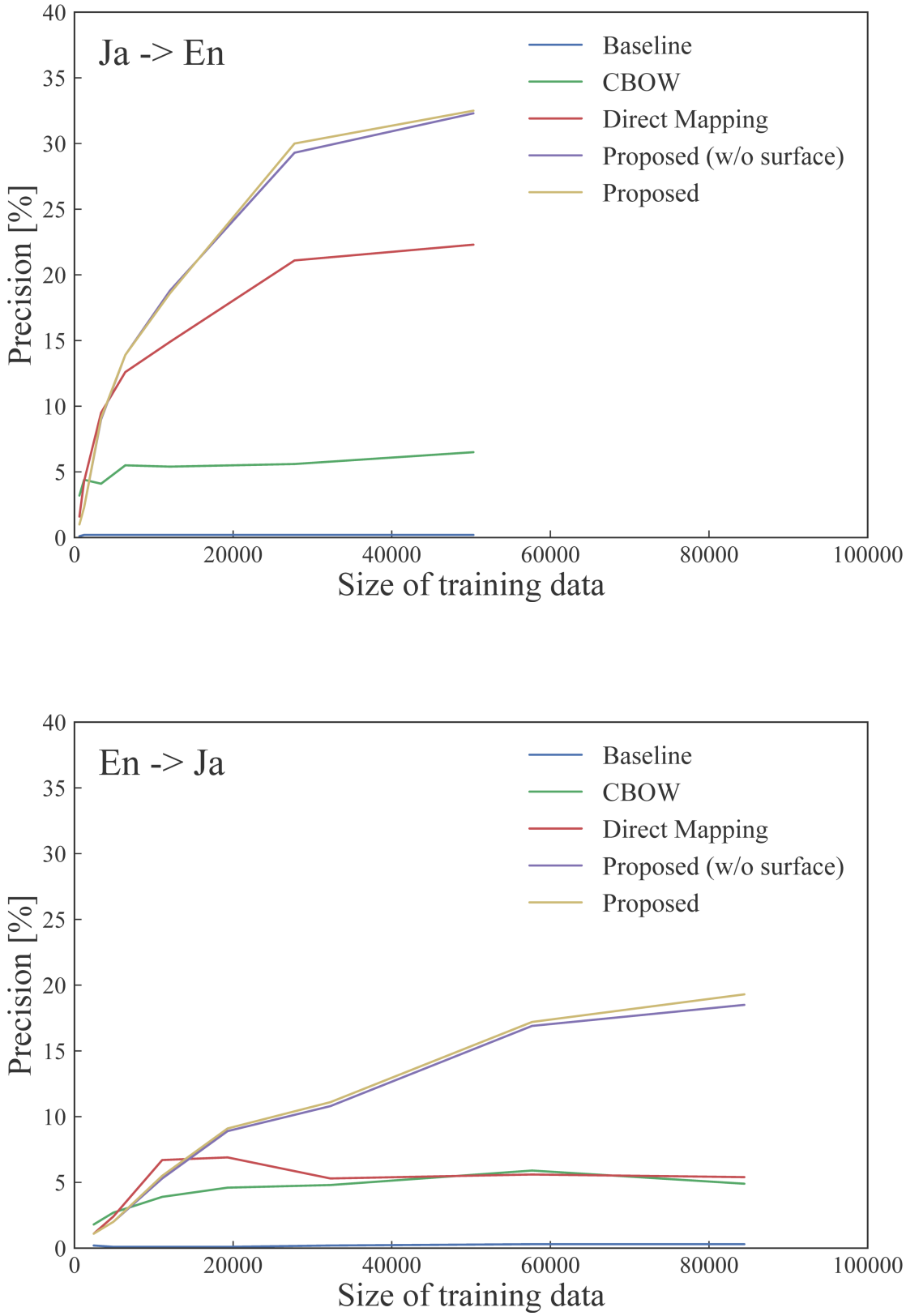


Figure 3.2: Impact of the size of training data. (Upper: (Ja → En), Bottom: (En → Ja)

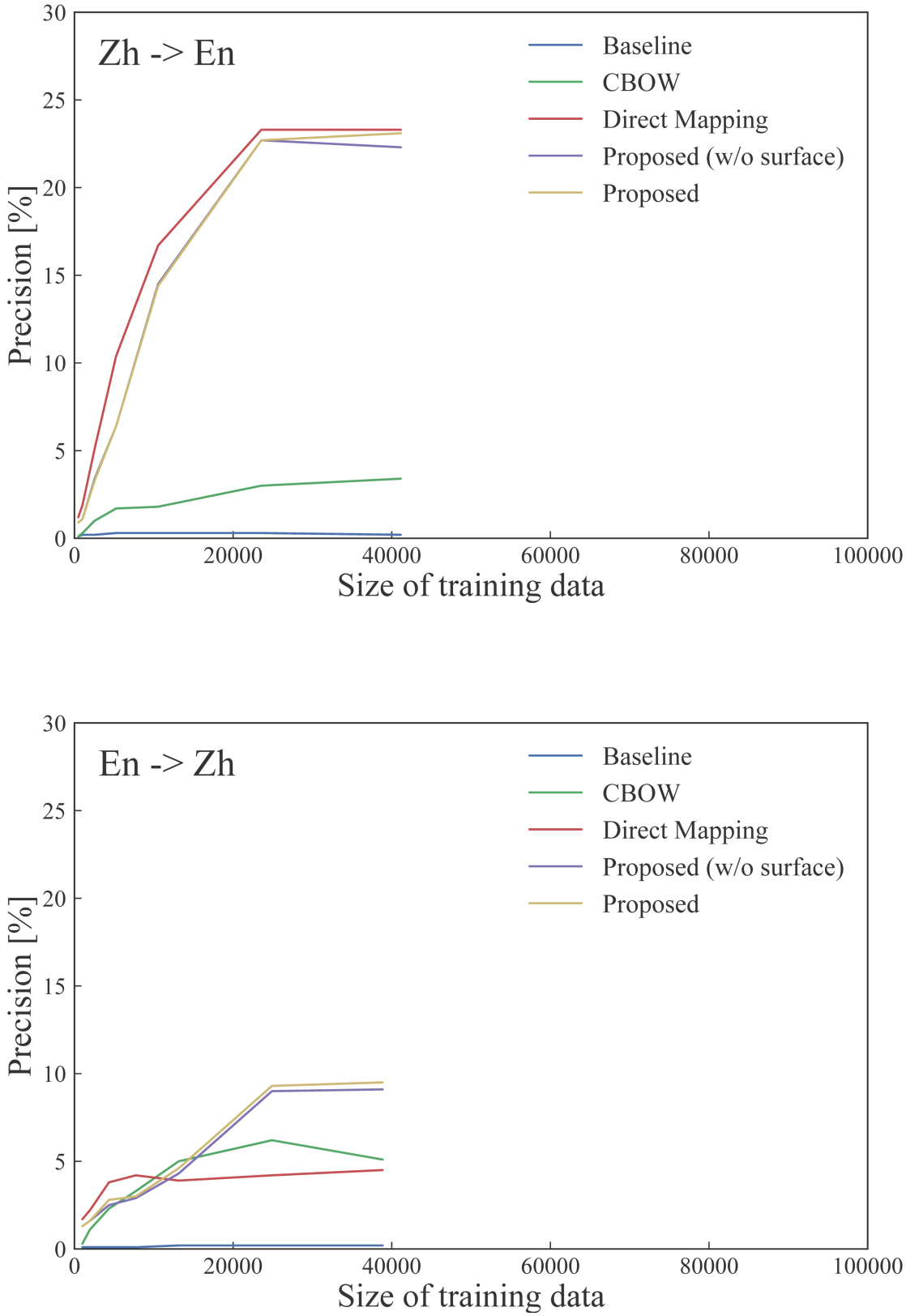


Figure 3.3: Impact of the size of training data. (Upper: (Zh → En), Bottom: (En → Zh)

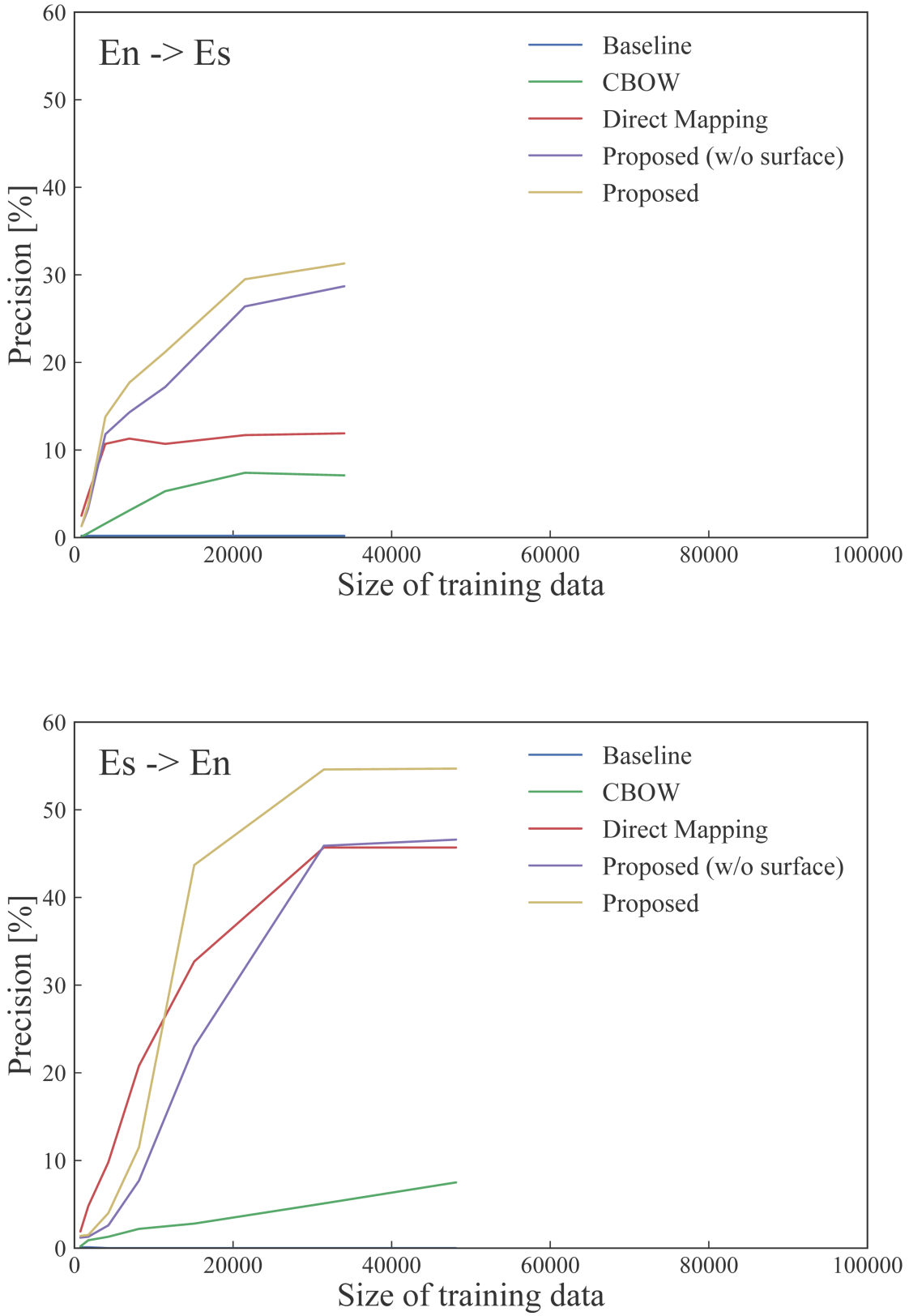


Figure 3.4: Impact of the size of training data. (Upper: (En → Es), Bottom: (Es → En)

Table 3.13: Hand-picked examples of the translations for the 10k sentences in the RECIPE corpus from Japanese to English. Text in bold denotes oov words in the input sentences and their translations. The subscripts of the translation of the oov words refer to a manual word alignment of the oov words.

Input	混ぜ ながら 弱火 で 煮る 。
Ref	simmer over low heat while mixing .
Baseline	煮る$_1$ at low heat while mixing .
Proposed (general)	**boil**$_1$ over a low heat while mixing .
Proposed (in-domain)	**simmer**$_1$ over a low heat while mixing .
Parallel Corpus	**simmer**$_1$ over low heat while stirring .
Input	玉ねぎ 、 ニンニク を みじん切り に 。
Ref	finely chop the onion and garlic .
Baseline	みじん切り$_1$ in the onion and garlic .
Proposed (general)	the garlic and onion in **butter**$_1$.
Proposed (in-domain)	**mince**$_1$ the onion and garlic .
Parallel Corpus	**finely**$_1$**chop**$_1$ the onion and garlic .
Input	オーブン トースター で 焦げ目 が つく まで 焼く 。
Ref	bake until browned in a toaster oven .
Baseline	in トースター$_1$ oven until 焦げ目$_2$ made 焼く$_3$.
Proposed (general)	oven in the **refrigerator**$_1$ until **fenbuconazole**$_2$ made **bread**$_3$.
Proposed (in-domain)	in a **toaster**$_1$ oven , **bake**$_3$ until the **end**$_2$.
Parallel Corpus	**bake**$_3$ in a **toaster**$_1$ oven until **golden**$_2$**brown**$_2$.
Input	しっとり し た 食 感 の 素朴 な ケーキ です 。
Ref	a simple cake with a moist texture .
Baseline	しっとり$_1$ food of a simple cake です$_2$.
Proposed (general)	the food **texture**$_1$ as a simple cake thing .
Proposed (in-domain)	the **moist**$_1$ food that 's simple cake .
Parallel Corpus	a **moist**$_1$**texture**$_1$ of the simple cake .
Input	火 を 消し 、 ごま油 を 入れ 混ぜる 。
Ref	turn off the heat , and stir in the sesame oil .
Baseline	消し$_1$ fire , and put ごま油$_2$ 混ぜる$_3$.
Proposed (general)	heat **butter**$_1$ completely , add the **milk**$_2$.
Proposed (in-domain)	fire , add **coconut**$_2$, and **mix**$_3$.
Parallel Corpus	**turn**$_1$**off**$_1$ the heat , add the **sesame**$_2$**oil**$_2$ and **mix**$_3$.

Chapter 4

Chunk-based Decoder for Neural Machine Translation

4.1 Overview

Neural machine translation (NMT) performs end-to-end translation based on a simple encoder-decoder model [44, 103, 15] and has now overtaken the classical, complex SMT in terms of performance and simplicity [94, 60, 19, 79]. In NMT, an encoder first maps a source sequence into vector representations and a decoder then maps the vectors into a target sequence (§ 4.2). This simple framework allows researchers to incorporate the structure of the source sentence as in SMT by leveraging various architectures as the encoder [44, 103, 15, 23]. Most of the NMT models, however, still rely on a sequential decoder based on a recurrent neural network (RNN) due to the difficulty in capturing the structure of a target sentence that is unseen during translation.

With the sequential decoder, however, there are two problems to be solved. First, it is difficult to model long-distance dependencies [6]. A hidden state h_t in an RNN is only conditioned by its previous output y_{t-1}, previous hidden state h_{t-1}, and current input x_t. This makes it difficult to capture the dependencies between an older output y_{t-N} if they are too far from the current output. This problem can become more serious when the target sequence becomes longer. For example, in Figure 4.1, when we translate the English sentence into the Japanese one, after the decoder predicts the content word "帰っ (go back)", it has to predict four function words "て (suffix)", "しまい (*perfect tense*)", "たい (*desire*)", and "と (to)" before predicting the

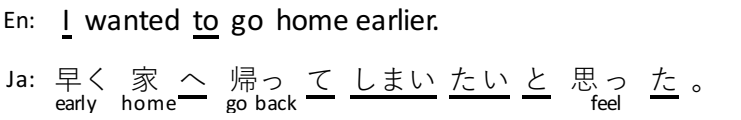


Figure 4.1: Translation from English to Japanese. The function words are underlined.

next content word "思っ (feel)". In such a case, the decoder is required to capture the longer dependencies in a target sentence.

Another problem with the sequential decoder is that it is expected to cover multiple possible word orders simply by memorizing the local word sequences in the limited training data. This problem can be more serious in free word-order languages such as Czech, German, Japanese, and Turkish. In the case of the example in Figure 4.1, the order of the phrase "早く (early)" and the phrase "家 へ (to home)" is flexible. This means that simply memorizing the word order in training data is not enough to train a model that can assign a high probability to a correct sentence regardless of its word order.

In the past, chunks (or phrases) were utilized to handle the above problems in SMT [109, 51] and in example-based machine translation (EBMT) [45]. By using a chunk rather than a word as the basic translation unit, one can treat a sentence as a shorter sequence. This makes it easy to capture the longer dependencies in a target sentence. The order of words in a chunk is relatively fixed while that in a sentence is much more flexible. Thus, modeling intra-chunk (local) word orders and inter-chunk (global) dependencies independently can help capture the difference of the flexibility between the word order and the chunk order in free word-order languages.

In this work, we refine the original RNN decoder to consider chunk information in NMT. We propose three novel NMT models that capture and utilize the chunk structure in the target language (§ 4.3). Our focus is the hierarchical structure of a sentence: each sentence consists of chunks, and each chunk consists of words. To encourage an NMT model to capture the hierarchical structure, we start from a hierarchical RNN that consists of a chunk-level decoder and a word-level decoder (Model 1). Then, we improve the word-level decoder by introducing inter-chunk connections to capture the interaction between chunks (Model 2). Finally, we introduce a feedback mechanism to the chunk-level decoder to enhance the memory capacity of previous outputs (Model 3).

We evaluate the three models on the WAT '16 English-to-Japanese translation task (§ 4.4). The experimental results show that our best model outperforms the best single NMT model reported in WAT '16 [23].

Our contribution is twofold: (1) chunk information is introduced into NMT to improve translation performance, and (2) a novel hierarchical decoder is devised to model the properties of chunk structure in the encoder-decoder framework.

4.2 Preliminaries: Neural Machine Translation

In this section, we briefly introduce the architecture of the attention-based NMT model [6], which is the basis of our proposed models.

4.2.1 Encoder-Decoder Model

An NMT model usually consists of two connected neural networks: an encoder and a decoder. After the encoder maps a source sentence into a fixed-length vector, the decoder maps the vector into a target sentence. The implementation of the encoder can be a convolutional neural network (CNN) [44], a long short-term memory (LSTM) [103, 60], a gated recurrent unit (GRU) [15, 6], or a Tree-LSTM [23]. While various architectures are leveraged as an encoder to capture the structural information in the source language, most of the NMT models rely on a standard sequential network such as LSTM or GRU as the decoder.

Following Bahdanau et al. [6], we use GRU as the recurrent unit in this work. A GRU unit computes its hidden state vector $\mathbf{h}_i$ given an input vector $\mathbf{x}_i$ and the previous hidden state $\mathbf{h}_{i-1}$:

$$\mathbf{h}_i = \text{GRU}(\mathbf{h}_{i-1}, \mathbf{x}_i). \tag{4.1}$$

The function GRU$(\cdot)$ is calculated as

$$\mathbf{r}_i = \sigma(\mathbf{W}_r \mathbf{x}_i + \mathbf{U}_r \mathbf{h}_{i-1} + \mathbf{b}_r), \tag{4.2}$$

$$\mathbf{z}_i = \sigma(\mathbf{W}_z \mathbf{x}_i + \mathbf{U}_z \mathbf{h}_{i-1} + \mathbf{b}_z), \tag{4.3}$$

$$\tilde{\mathbf{h}}_i = \tanh(\mathbf{W}\mathbf{x}_i + \mathbf{U}(\mathbf{r}_i \odot \mathbf{h}_{i-1} + \mathbf{b})), \tag{4.4}$$

$$\mathbf{h}_i = (\mathbf{1} - \mathbf{z}_i) \odot \tilde{\mathbf{h}}_i + \mathbf{z}_i \odot \mathbf{h}_{i-1}, \tag{4.5}$$

where vectors $\mathbf{r}_i$ and $\mathbf{z}_i$ are a reset gate and an update gate, respectively. While the former gate allows the model to forget the previous states, the latter gate decides how much the model updates its content. All the $\mathbf{W}$s and $\mathbf{U}$s, or the $\mathbf{b}$s above are trainable matrices or vectors. $\sigma(\cdot)$ and $\odot$ denote the sigmoid function and element-wise multiplication operator, respectively.

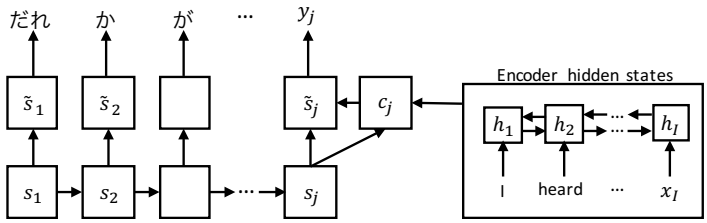


Figure 4.2: Standard word-based decoder.

In this simple model, we train a GRU function that encodes a source sentence $\{x_1, \cdots, x_I\}$ into a single vector $\mathbf{h}_I$. At the same time, we jointly train another GRU function that decodes $\mathbf{h}_I$ to the target sentence $\{y_1, \cdots, y_J\}$. Here, the j-th word in the target sentence y_j can be predicted with this decoder GRU and a nonlinear function $g(\cdot)$ followed by a softmax layer, as

$$\mathbf{c} = \mathbf{h}_I, \tag{4.6}$$

$$\mathbf{s}_j = \text{GRU}(\mathbf{s}_{j-1}, [\mathbf{y}_{j-1}; \mathbf{c}]), \tag{4.7}$$

$$\tilde{\mathbf{s}}_j = g(\mathbf{y}_{j-1}, \mathbf{s}_j, \mathbf{c}), \tag{4.8}$$

$$P(y_j | \mathbf{y}_{<j}, \mathbf{x}) = \text{softmax}(\tilde{\mathbf{s}}_j), \tag{4.9}$$

where $\mathbf{c}$ is a context vector of the encoded sentence and $\mathbf{s}_j$ is a hidden state of the decoder GRU.

Following Bahdanau et al. [6], we use a mini-batch stochastic gradient descent (SGD) algorithm with ADADELTA [119] to train the above two GRU functions (i.e., the encoder and the decoder) jointly. The objective is to minimize the cross-entropy loss of the training data $\mathbf{D}$, as

$$J = \sum_{(\mathbf{x},\mathbf{y}) \in \mathbf{D}} -\log P(\mathbf{y}|\mathbf{x}). \tag{4.10}$$

4.2.2 Attention Mechanism for Neural Machine Translation

To use all the hidden states of the encoder and improve the translation performance of long sentences, Bahdanau et al. [6] proposed using an attention mechanism. In the attention model, the context vector is not simply the last encoder state h_I but rather the weighted sum of all hidden states of the bidirectional GRU, as follows:

$$\mathbf{c}_j = \sum_{i=1}^{I} \alpha_{ji} \mathbf{h}_i. \tag{4.11}$$

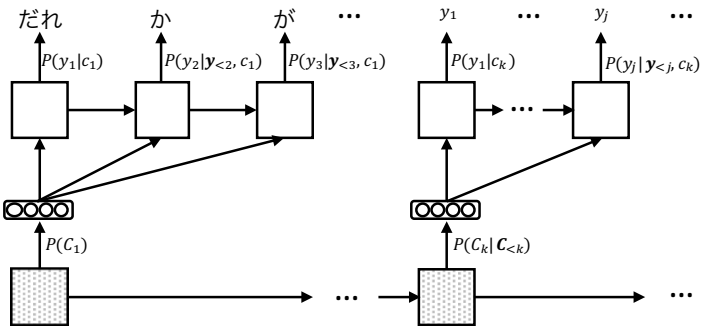


Figure 4.3: Chunk-based decoder. The top layer (word-level decoder) illustrates the first term in Eq. (4.15) and the bottom layer (chunk-level decoder) denotes the second term.

Here, the weight α_{ji} decides how much a source word x_i contributes to the target word y_j. α_{ji} is computed by a feedforward layer and a softmax layer as

$$e_{ji} = \mathbf{v} \cdot \tanh(\mathbf{W}_e \mathbf{h}_i + \mathbf{U}_e \mathbf{s}_j + \mathbf{b}_e), \tag{4.12}$$

$$\alpha_{ji} = \frac{\exp(e_{ji})}{\sum_{j'=1}^{J} \exp(e_{j'i})}, \tag{4.13}$$

where $\mathbf{W}_e$, $\mathbf{U}_e$ are trainable matrices and the $\mathbf{v}$, $\mathbf{b}_e$ are trainable vectors.[1] In a decoder using the attention mechanism, the obtained context vector $\mathbf{c}_j$ in each time step replaces $\mathbf{c}$s in Eqs. (4.7) and (4.8). An illustration of the NMT model with the attention mechanism is shown in Figure 4.2.

The attention mechanism is expected to learn alignments between source and target words, and plays a similar role to the translation model in phrase-based SMT [51].

4.3 Proposed: Neural Machine Translation with Chunk-based Decoder

Taking non-sequential information such as chunks (or phrases) structure into consideration has proved helpful for SMT [109, 51] and EBMT [45]. Here, we focus on two important properties of chunks [1]: (1) The word order in a chunk is almost always fixed, and (2) A chunk consists of a few (typically one) content words surrounded by zero or more function words.

[1] We choose this implementation following [62], while [6] use s_{j-1} instead of s_j in Eq. (5.4).

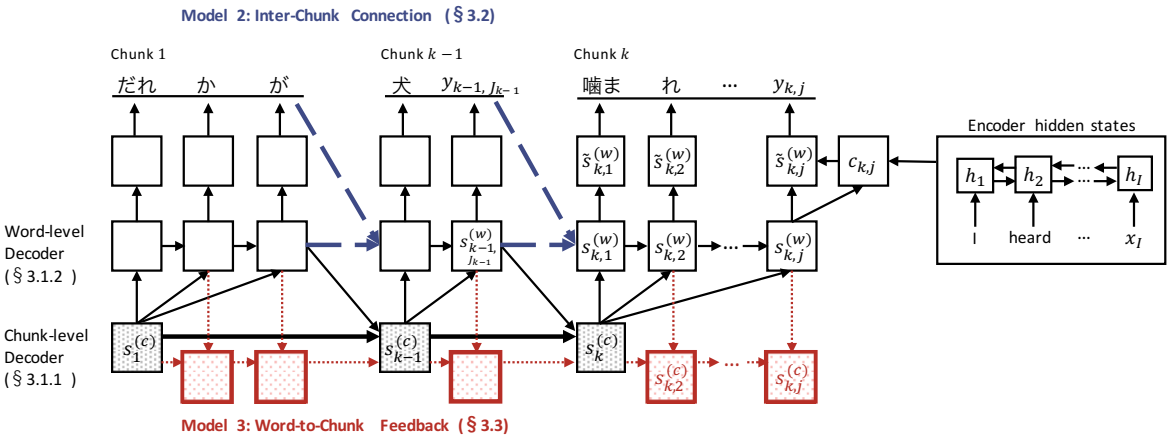


Figure 4.4: Proposed model: NMT with chunk-based decoder. A chunk-level decoder generates a chunk representation for each chunk while a word-level decoder uses the representation to predict each word. The solid lines in the figure illustrate Model 1. The dashed blue arrows in the word-level decoder denote the connections added in Model 2. The dotted red arrows in the chunk-level decoder denote the feedback states added in Model 3; the connections in the thick black arrows are replaced with the dotted red arrows.

To fully utilize the above properties of a chunk, we propose modeling the intra-chunk and the inter-chunk dependencies independently with a "chunk-by-chunk" decoder (See Figure 4.3). In the standard word-by-word decoder described in § 4.2, a target word y_j in the target sentence $\mathbf{y}$ is predicted by taking the previous outputs $\mathbf{y}_{<j}$ and the source sentence $\mathbf{x}$ as input:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{j=1}^{J} P(y_j|\mathbf{y}_{<j}, \mathbf{x}), \tag{4.14}$$

where J is the length of the target sentence. Not assuming any structural information of the target language, the sequential decoder has to memorize long dependencies in a sequence. To release the model from the pressure of memorizing the long dependencies over a sentence, we redefine this problem as the combination of a word prediction problem and a chunk generation problem:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{k=1}^{K} \left\{ P(c_k|\mathbf{c}_{<k}, \mathbf{x}) \prod_{j=1}^{J_k} P(y_j|\mathbf{y}_{<j}, c_k, \mathbf{x}) \right\}, \tag{4.15}$$

where K is the number of chunks in the target sentence and J_k is the length of the k-th chunk (see Figure 4.3). The first term represents the generation probability of a chunk

c_k and the second term indicates the probability of a word y_j in the chunk. We model the former term as a chunk-level decoder and the latter term as a word-level decoder. As demonstrated later in § 4.4, both K and J_k are much shorter than the sentence length J, which is why our decoders do not have to capture the long dependencies like the standard decoder does.

In the above formulation, we model the information on words and their orders in a chunk. No matter which language we target, we can assume that a chunk usually consists of some content words and function words, and the word order in the chunk is almost always fixed [1]. Although our idea can be used in several languages, the optimal network architecture could depend on the word order of the target language. In this work, we design models for languages in which content words are followed by function words, such as Japanese and Korean. The details of our models are described in the following sections.

4.3.1 Model 1: Basic Chunk-based Decoder

The model described in this section is the basis of our proposed decoders. It consists of two parts: a chunk-level decoder (§ 4.3.1) and a word-level decoder (§ 4.3.1). The part drawn in black solid lines in Figure 4.4 illustrates the architecture of Model 1.

Chunk-level Decoder

Our chunk-level decoder (see Figure 4.3) outputs a chunk representation. The chunk representation contains the information about words that should be predicted by the word-level decoder.

To generate the representation of the k-th chunk $\tilde{\mathbf{s}}_k^{(c)}$, the chunk-level decoder (see the bottom layer in Figure 4.4) takes the last states of the word-level decoder $\mathbf{s}_{k-1,J_{k-1}}^{(w)}$ and updates its hidden state $\mathbf{s}_k^{(c)}$ as:

$$\mathbf{s}_k^{(c)} = \text{GRU}(\mathbf{s}_{k-1}^{(c)}, \mathbf{s}_{k-1,J_{k-1}}^{(w)}), \tag{4.16}$$

$$\tilde{\mathbf{s}}_k^{(c)} = \mathbf{W}_c \mathbf{s}_k^{(c)} + \mathbf{b}_c. \tag{4.17}$$

The obtained chunk representation $\tilde{\mathbf{s}}_k^{(c)}$ continues to be fed into the word-level decoder until it outputs all the words in the current chunk.

Word-level Decoder

Our word-level decoder (see Figure 4.4) differs from the standard sequential decoder described in § 4.2 in that it takes the chunk representation $\tilde{s}_k^{(c)}$ as input:

$$\mathbf{s}_{k,j}^{(w)} = \text{GRU}(\mathbf{s}_{k,j-1}^{(w)}, [\tilde{\mathbf{s}}_k^{(c)}; \mathbf{y}_{k,j-1}; \mathbf{c}_{k,j-1}]), \tag{4.18}$$

$$\tilde{\mathbf{s}}_{k,j}^{(w)} = g(\mathbf{y}_{k,j-1}, \mathbf{s}_{k,j}^{(w)}, \mathbf{c}_{k,j}), \tag{4.19}$$

$$P(y_{k,j}|\mathbf{y}_{<j}, \mathbf{x}) = \text{softmax}(\tilde{\mathbf{s}}_{k,j}^{(w)}). \tag{4.20}$$

In a standard sequential decoder, the hidden state iterates over the length of a target sentence and then generates an end-of-sentence token. In other words, its hidden layers are required to memorize the long-term dependencies and orders in the target language. In contrast, in our word-level decoder, the hidden state iterates only over the length of a chunk and then generates an end-of-chunk token. Thus, our word-level decoder is released from the pressure of memorizing the long (inter-chunk) dependencies and can focus on learning the short (intra-chunk) dependencies.

4.3.2 Model 2: Inter-Chunk Connection

The second term in Eq. (4.15) only iterates over one chunk ($j = 1$ to J_k). This means that the last state and the last output of a chunk are not being fed into the word-level decoder at the next time step (see the black part in Figure 4.4). In other words, $\mathbf{s}_{k,1}^{(w)}$ in Eq. (4.18) is always initialized before generating the first word in a chunk. This may have a bad influence on the word-level decoder because it cannot access any previous information at the first word of each chunk.

To address this problem, we add new connections to Model 1 between the first state in a chunk and the last state in the previous chunk, as

$$\mathbf{s}_{k,1}^{(w)} = \text{GRU}(\mathbf{s}_{k-1,J_{k-1}}^{(w)}, [\tilde{\mathbf{s}}_k^{(c)}; \mathbf{y}_{k-1,J_{k-1}}; \mathbf{c}_{k-1,J_{k-1}}]). \tag{4.21}$$

The dashed blue arrows in Figure 4.4 illustrate the added inter-chunk connections.

4.3.3 Model 3: Word-to-Chunk Feedback

The chunk-level decoder in Eq. (4.16) is only conditioned by $\mathbf{s}_{k-1,J_{k-1}}^{(w)}$, the last word state in each chunk (see the black part in Figure 4.4). This may affect the chunk-level

decoder because it cannot memorize what kind of information has already been generated by the word-level decoder. The information about the words in a chunk should not be included in the representation of the next chunk; otherwise, it may generate the same chunks multiple times, or forget to translate some words in the source sentence.

To encourage the chunk-level decoder to memorize the information about the previous outputs more carefully, we add feedback states to our chunk-level decoder in Model 2. The feedback state in the chunk-level decoder is updated at every time step $j(>1)$ in k-th chunk, as

$$\mathbf{s}^{(c)}_{k,j} = \mathrm{GRU}(\mathbf{s}^{(c)}_{k,j-1}, \mathbf{s}^{(w)}_{k,j}). \tag{4.22}$$

The red part in Figure 4.4 illustrate the added feedback states and their connections. The connections in the thick black arrows are replaced with the dotted red arrows in Model 3.

4.4 Experiments

4.4.1 Settings

Dataset

To examine the effectiveness of our decoders, we chose Japanese, a free word-order language, as the target language. Japanese sentences are easy to break into well-defined chunks (called *bunsetsus* [38] in Japanese). For example, the accuracy of *bunsetsu*-chunking on newspaper articles is reported to be over 99% [74, 117]. The effect of chunking errors in training the decoder can be suppressed, which means we can accurately evaluate the potential of our method. We used the English-Japanese training corpus in the Asian Scientific Paper Excerpt Corpus (ASPEC) [75], which was provided in WAT '16. To remove inaccurate translation pairs, we extracted the first two million out of the 3 million pairs following the setting that gave the best performances in WAT '15 [80].

Corpus	# words	# chunks	# sentences
Train	49,671,230	15,934,129	1,663,780
Dev.	54,287	-	1,790
Test	54,088	-	1,812

Table 4.1: Statistics of the target language (Japanese) in extracted corpus after preprocessing.

Preprocessings

For Japanese sentences, we performed tokenization using KyTea 0.4.7[2] [81]. Then we performed *bunsetsu*-chunking with J.DepP 2015.10.05[3] [115–117]. Special end-of-chunk tokens were inserted at the end of the chunks. Our word-level decoders described in § 4.3 will stop generating words after each end-of-chunk token. For English sentences, we performed the same preprocessings described on the WAT '16 Website.[4] To suppress having possible chunking errors affects the translation quality, we removed extremely long chunks from the training data. Specifically, among the 2 million preprocessed translation pairs, we excluded sentence pairs that matched any of following conditions: (1) The length of the source sentence or target sentence is larger than 64 (3% of whole data); (2) The maximum length of a chunk in the target sentence is larger than 8 (14% of whole data); and (3) The maximum number of chunks in the target sentence is larger than 20 (3% of whole data). Table 5.1 shows the details of the extracted data.

Postprocessing

To perform unknown word replacement [61], we built a bilingual English-Japanese dictionary from all of the three million translation pairs. The dictionary was extracted with the MGIZA++ 0.7.0[5] [86, 33] word alignment tool by automatically extracting the alignments between English words and Japanese words.

Model Architecture

Any encoder can be combined with our decoders. In this work, we adopted a single-layer bidirectional GRU [15, 6] as the encoder to focus on confirming the impact

[2]http://www.phontron.com/kytea/
[3]http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/jdepp/
[4]http://lotus.kuee.kyoto-u.ac.jp/WAT/baseline/dataPreparationJE.html
[5]https://github.com/moses-smt/mgiza

ρ of ADADELTA	0.95
ϵ of ADADELTA	$1e^{-6}$
Initial learning rate	1.0
Gradient clipping	1.0
Mini-batch size	64
d_{hid} (dimension of hidden states)	1024
d_{emb} (dimension of word embeddings)	1024

Table 4.2: Hyperparameters for training.

of the proposed decoders. We used single layer GRUs for the word-level decoder and the chunk-level decoder. The vocabulary sizes were set to 40k for source side and 30k for target side, respectively. The conditional probability of each target word was computed with a deep-output [89] layer with maxout [35] units following [6]. The maximum number of output chunks was set to 20 and the maximum length of a chunk was set to 8.

Training Details

The models were optimized using ADADELTA following [6]. The hyperparameters of the training procedure were fixed to the values given in Table 4.2. Note that the learning rate was halved when the BLEU score on the development set did not increase for 30,000 batches. All the parameters were initialized randomly with a Gaussian distribution. It took about a week to train each model with an NVIDIA TITAN X (Pascal) GPU.

Evaluation

Following the WAT '16 evaluation procedure, we used BLEU [88] and RIBES [42] to evaluate our models. The BLEU scores were calculated with `multi-bleu.pl` in Moses 2.1.1[6] [48]; RIBES scores were calculated with `RIBES.py` 1.03.1[7] [42]. Following Cho et al. [14], we performed beam search[8] with length-normalized log-probability to decode target sentences. We saved the trained models that performed best on the development set during training and used them to evaluate the systems with the test set.

[6]http://www.statmt.org/moses/
[7]http://www.kecl.ntt.co.jp/icl/lirg/ribes/index.html
[8]Beam size is set to 20.

System Encoder/	Decoder Type	Hyperparameter $\|V_{src}\|$	$\|V_{trg}\|$	d_{emb}	d_{hid}	BLEU	RIBES	Dec. time [ms/sent.]
Word- /	Word-based [22]	88k	66k	512	512	34.64	81.60	-
/	Word-based (our implementation)	40k	30k	1024	1024	36.33	81.22	84.1
	+ chunked training data via J.DepP	40k	30k	1024	1024	35.71	80.89	101.5
Tree- /	Word-based [23]	88k	66k	512	512	34.91	81.66	(363.7)
/	Char-based [22]	88k	3k	256	512	31.52	79.39	(**8.8**)
Word- /	Proposed Chunk-based (Model 1)	40k	30k	1024	1024	34.70	81.01	165.2
	+ Inter-chunk connection (Model 2)	40k	30k	1024	1024	35.81	81.29	165.2
	+ Word-to-chunk feedback (Model 3)	40k	30k	1024	1024	**37.26**	**82.23**	163.7

Table 4.3: The settings and results of the baseline systems and our systems. $|V_{src}|$ and $|V_{trg}|$ denote the vocabulary size of the source language and the target language, respectively. d_{emb} and d_{hid} are the dimension size of the word embeddings and hidden states, respectively. Note that the Tree-to-Seq models are tested on CPUs instead of GPUs. Only single NMT models (w/o ensembling) reported in WAT '16 are listed here. Full results are available on the WAT '16 Website.

Baseline Systems

The baseline systems and the important hyperparamters are listed in Table 4.3. Eriguchi et al. [22]'s baseline system (the first line in Table 4.3) was the best single (w/o ensembling) word-based NMT system that were reported in WAT '16. For a fairer evaluation, we also reimplemented a standard attention-based NMT system that uses exactly the same encoder, training procedure, and the hyperparameters as our proposed models, but has a word-based decoder. We trained this system on the training data without chunk segmentations (the second line in Table 4.3) and with chunk segmentations given by J.DepP (the third line in Table 4.3). The chunked corpus fed to the third system is exactly the same as the training data of our proposed systems (sixth to eighth lines in Table 4.3). In addition, we also include the Tree-to-Sequence models [22, 23] (the fourth and fifth lines in Table 4.3) to compare the impact of capturing the structure in the source language and that in the target language. Note that all systems listed in Table 4.3, including our models, are single models without ensemble techniques.

4.4.2 Results

Proposed Models vs. Baselines

Table 4.3 shows the experimental results on the ASPEC test set. We can observe that our best model (Model 3) outperformed all the single NMT models reported in WAT '16. The gain obtained by switching Word-based decoder to Chunk-based decoder (+0.93 BLEU and +1.01 RIBES) is larger than the gain obtained by switching word-based encoder to Tree-based encoder (+0.27 BLEU and +0.06 RIBES). This result shows that capturing the chunk structure in the target language is more effective than capturing the syntax structure in the source language. Compared with the character-based NMT model [22], our Model 3 performed better by +5.74 BLEU score and +2.84 RIBES score. One possible reason for this is that using a character-based model rather than a word-based model makes it more difficult to capture long-distance dependencies because the length of a target sequence becomes much longer in the character-based model.

Comparison between Baselines

Among the five baselines, our reimplementation without chunk segmentations (the second line in Table 4.3) achieved the best BLEU score while the Eriguchi et al. [23]'s system (the fourth line in Table 4.3) achieved the best RIBES score. The most probable reasons for the superiority of our reimplementation over the Eriguchi et al. [22]'s word-based baseline (the first line in Table 4.3) is that the dimensions of word embeddings and hidden states in our systems are higher than theirs.

Feeding chunked training data to our baseline system (the third line in Table 4.3) instead of normal data had bad effects by -0.62 BLEU score and by -0.33 RIBES score. We evaluated the chunking ability of this system by comparing the positions of end-of-chunk tokens generated by this system with the chunk boundaries obtained by J.DepP. To our surprise, this word-based decoder could output chunk separations as accurate as our proposed Model 3 (both systems achieved F_1-score > 97). The results show that even a standard word-based decoder has the ability to predict chunk boundaries if they are given in training data. However, it is difficult for the word-based decoder to utilize the chunk information to improve the translation quality.

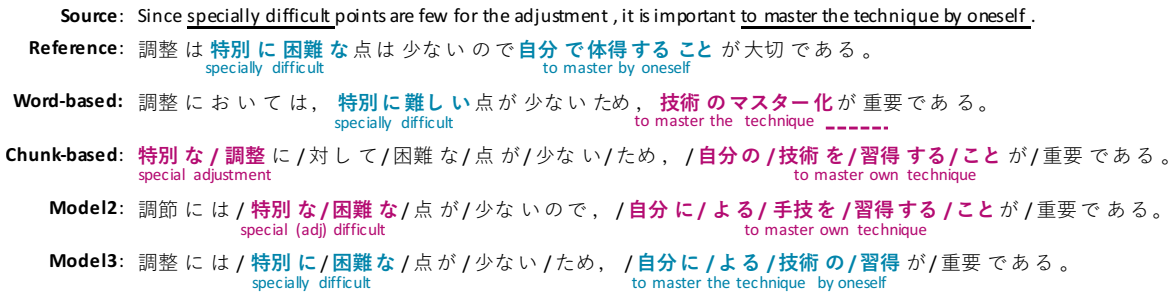


Figure 4.5: Translation examples. "/" denote chunk boundaries that are automatically determined by our decoders. Words colored blue and red respectively denote correct translations and wrong translations.

Decoding Speed

Although the chunk-based decoder runs 2x slower than our word-based decoder, it is still practically acceptable (6 sentences per second). The character-based decoder (the fifth line in Table 4.3) is less time-consuming mainly because of its small vocabulary size ($|V_{trg}| = 3$k).

Qualitative Analysis

To clarify the qualitative difference between the word-based decoder and our chunk-based decoders, we show translation examples in Figure 4.5. Words in blue and red respectively denote correct translations and wrong translations. The word-based decoder (our implementation) has completely dropped the translation of "by oneself." On the other hand, Model 1 generated a slightly wrong translation "自分 の 技術 を 習得 する こと (to master own technique)." In addition, Model 1 has made another serious word-order error "特別 な 調整 (special adjustment)." These results suggest that Model 1 can capture longer dependencies in a long sequence than the word-based decoder. However, Model 1 is not good at modeling global word order because it cannot access enough information about previous outputs. The weakness of modeling word order was overcome in Model 2 thanks to the inter-chunk connections. However, Model 2 still suffered from the errors of function words: it still generates a wrong chunk "特別 な (special)" instead of the correct one "特別 に (specially)" and a wrong chunk "よ る" instead of "よ り." Although these errors seem trivial, such mistakes with function words bring serious changes of sentence meaning. However, all of these problems have disappeared in Model 3. This phenomenon supports the importance of the feedback states to provide the decoder with a better ability to choose more accurate words in chunks.

Decoder	C-BLEU	C-RIBES
Word-based (our implementation)	7.56	50.73
+ chunked training data via J.DepP	7.40	51.18
Proposed Chunk-based (Model 1)	7.59	50.47
+ Inter-chunk connection (Model 2)	7.78	51.48
+ Word-to-chunk feedback (Model 3)	**8.69**	**52.82**

Table 4.4: Chunk-based BLEU and RIBES with the systems using the word-based encoder.

4.5 Discussion

In this section, we present further analyses of our model to fully understand its ability to capture chunk structures and the performance in a low-resource scenario.

4.5.1 Chunk-level Evaluation

Can our models capture local (intra-chunk) and global (inter-chunk) word orders better than the conventional encoder-decoder model? To answer this question, we evaluated the translation quality at the chunk level. First, we performed *bunsetsu*-chunking on the reference translations in the test set. Then, for both reference translations and the outputs of our systems, we combined all the words in each chunk into a single token to regard a chunk as the basic translation unit instead of a word. Finally, we computed the chunk-based BLEU (C-BLEU) and RIBES (C-RIBES).

The results are listed in Table 4.4. For the word-based decoder (the first line in Table 4.4), we performed *bunsetsu*-chunking by J.DepP on its outputs to obtain chunk boundaries. As another baseline (the second line in Table 4.4), we used the chunked sentences as training data instead of performing chunking after decoding. The results show that our models (Model 2 and Model 3) outperform the word-based decoders in both C-BLEU and C-RIBES. This indicates that our chunk-based decoders can produce more correct chunks in a more correct order than the word-based models.

4.5.2 Impact of the Size of Training Data

In order to further investigate the effectiveness of the proposed models in various situations, we conduct a set of experiments in a low-resource setting. First, we shuffled the ASPEC corpus and randomly sampled its subsets to build six small

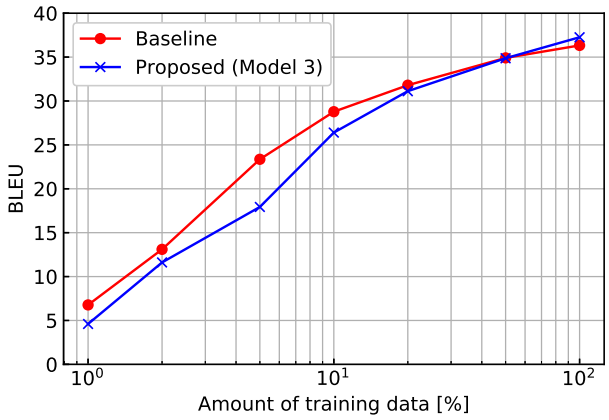


Figure 4.6: Impact of the size of training data on BLEU

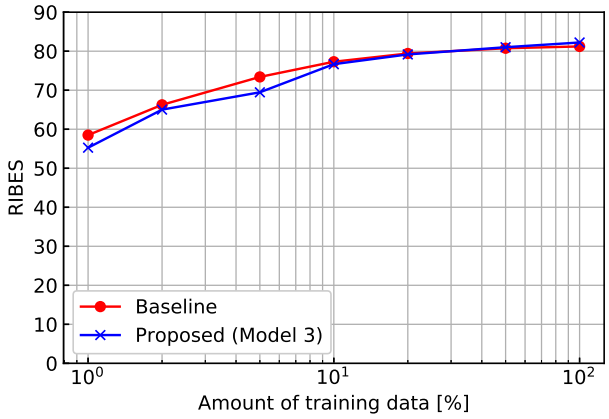


Figure 4.7: Impact of the size of training data on RIBES

datasets. The size of the six datasets are n% (n = 1,2,5,10,20,50) of the original full-size dataset. Figure 4.6 and Figure 4.7 show the BLEU and RIBES on the six datasets, respectively. We found that the Model 3 performs worse than the baseline if the size of the training corpus is small (especially when $n \leqq 20$).

Why does the chunk-based decoder require more data than the conventional word-based decoder does? To answer this question, we set up two hypotheses that may explain the experimental results. The first hypothesis is that the chunk-based decoder needs more data because it has more parameters than the word-based decoder has. Our second hypothesis is that it is because the chunk-based decoder learns a more difficult problem; while the word-based decoder is solely learning the translation task, our model needs to jointly learn chunking and translation with a

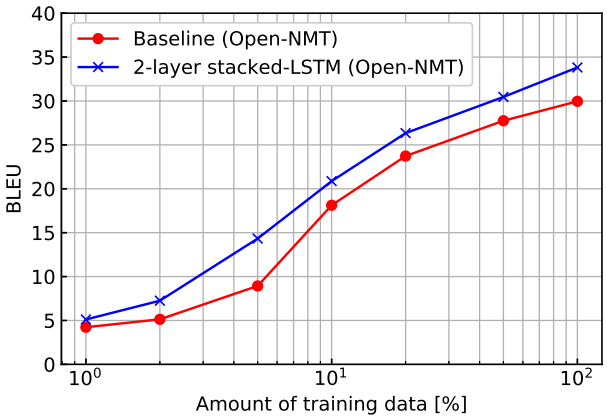


Figure 4.8: Comparison of the single-layer RNN and the stacked-RNN on BLEU

single model. In the following, we examine these two hypotheses by conducting additional experiments.

Comparison with the stacked-RNN decoder

In order to confirm the impact of the number of parameters that the model has, we conduct an additional experiment using stacked-RNN. Since the Model 3 consists of two decoders and they are connected to each other, its architecture looks similar to a two-layer stacked RNN and the number of their parameters are also comparable. In previous work, it is shown that stacking the RNN layers to make model *deeper* is effective in improving the translation quality [61, 103]. However, if a large number of parameters is the reason for the worse performance in the low-resource setting, stacking the RNN should also has a negative effect when the data size is small.

Since our implementation of chunk-based decoder is hard to extend, we instead used Open-NMT,[8] a Pytorch[9] implementation of word-based encoder-decoder model. We used exactly the same dataset and hyper-parameters as other experiments, with different optimization[10] process and post processings.[11]

Figure 4.8 and Figure 4.9 show the the experimental results on BLEU and RIBES, respectively. Surprisingly, the two figures indicate that stacking the decoder RNNS does not have negative effects on translation performance (in either BLEU or RIBES) even in the low-resource setting. Considering the two-layer stacked-RNN has the

[8] `https://http://opennmt.net/`
[9] `https://https://pytorch.org/`
[10] We used Adam, without learning rate halving in this analysis.
[11] We did not use unknown-word replacement.

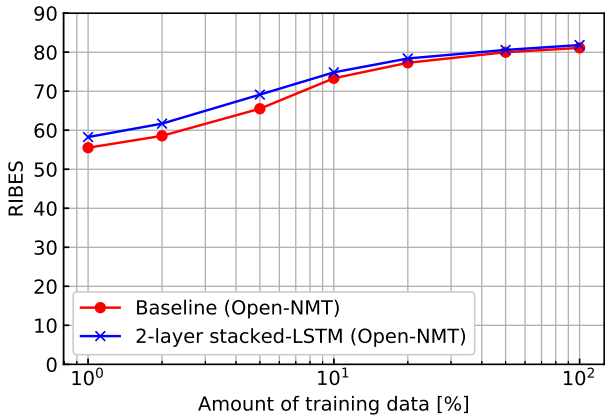


Figure 4.9: Comparison of the single-layer RNN and the stacked-RNN on RIBES

number of parameters that are comparable to the Model 3, this result suggests that our first hypothesis is not correct.

Chunking performance in the low-resource setting

Our second hypothesis is that the worse performance with small data is caused by the difficulty of the problem that the proposed models need to solve. To verify this hypothesis, we evaluate the performance of chunking in the low-resource setting. First, we removed the chunk separations that were recognized by the Model 3 automatically. Next, we performed *bunsetsu*-chunking with J.DepP. Assuming the outputs of J.DepP as oracles, we finally evaluate the chunking accuracy of the Model 3.

Figure 4.10 shows the impact of the data size on chunking performance. We can see that chunking performance becomes much worse as we reduce the training data. If we decrease the size of the training corpus from the resource-rich setting ($n = 100$) to a low-resource setting ($n = 1$), the chunking accuracy drops from 0.87 to 0.76. This result suggests that not only translation but also chunk boundary detection can be a difficult task when the data size becomes small. Because our model needs to learn the two tasks jointly, it would be affected by the lack of data more seriously than the baseline model does.

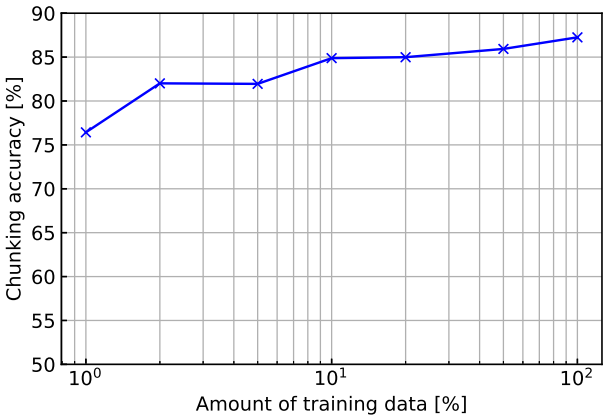


Figure 4.10: Impact of the size of training data on chunking performance (sentence-level accuracy).

4.6 Chapter Summary

In this chapter, we proposed an NMT model that can capture the chunk structure in the target language. As the attention mechanism in NMT plays a similar role to the translation model in phrase-based SMT, our chunk-based decoders are intended to capture the notion of chunks in chunk-based (or phrase-based) SMT. We designed three models that have hierarchical RNN-like architectures, each of which consists of a word-level decoder and a chunk-level decoder. Since the chunk structure can be learned by adding chunk boundaries to training data explicitly, no additional preprocessing like Part-of-Speech tagging or syntactic parsing is required during testing time. We performed experiments on the WAT '16 English-to-Japanese translation task and found that our best model outperforms the strongest baselines by +0.93 BLEU score and by +0.57 RIBES score.

Chapter 5

Learning to Describe Phrases with Local and Global Contexts

5.1 Overview

When we read news text with emerging entities, text in unfamiliar domains, or text in foreign languages, we often encounter expressions (words or phrases) whose senses we are unsure of. In such cases, we may first try to figure out the meanings of those expressions by reading the surrounding words (*local* context) carefully. Failing to do so, we may consult dictionaries, and in the case of polysemous words, choose an appropriate meaning based on the context. Learning novel word senses via dictionary definitions is known to be more effective than contextual guessing [13, 30]. However, very often, hand-crafted dictionaries do not contain definitions of expressions that are rarely used or newly created. Ultimately, we may need to read through the entire document or even search the web to find other occurances of the expression (*global* context) so that we can guess its meaning.

Can machines help us do this work? Ni and Wang [83] have proposed a task of generating a definition for a phrase given its local context. However, they follow the strict assumption that the target phrase is newly emerged and there is only a single local context available for the phrase, which makes the task of generating an accurate and coherent definition difficult (perhaps as difficult as a human comprehending the phrase itself). On the other hand, Noraset et al. [85] attempted to generate a definition of a word from an embedding induced from massive text (which can be seen as global context). This is followed by Gadetsky et al. [32] that refers to a local context to disambiguate polysemous words by choosing relevant dimensions of

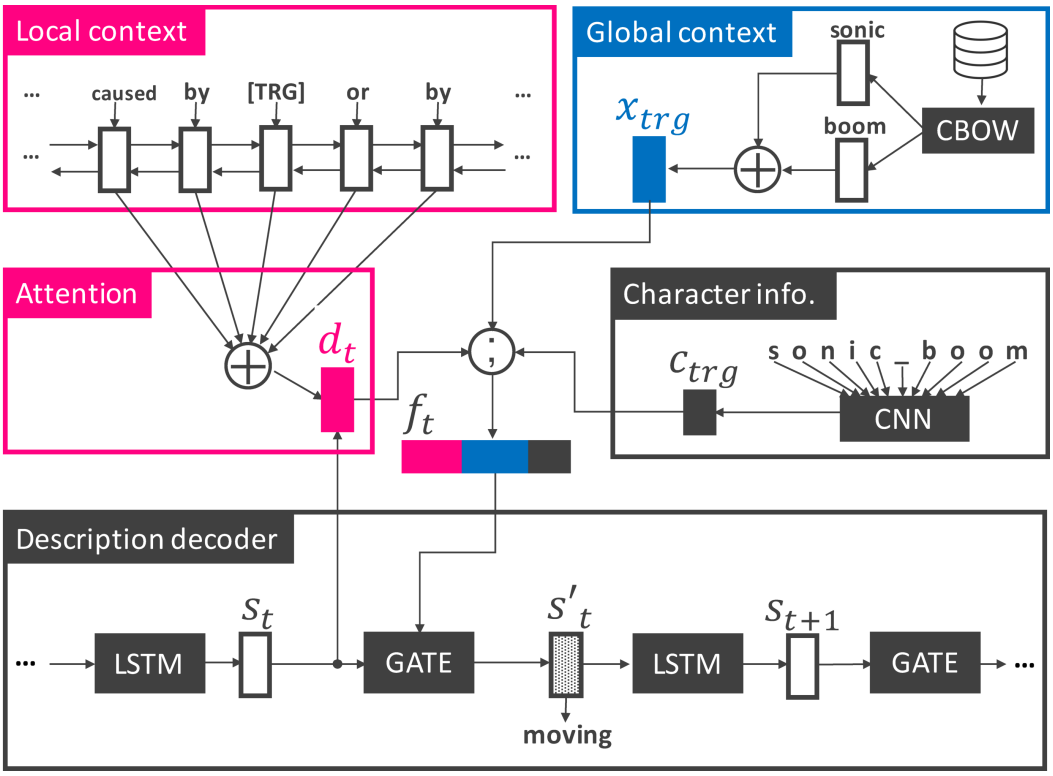


Figure 5.1: **Lo**cal & **G**lobal **C**ontext-**a**ware **D**escription generator (**LOG-CaD**).

their word embeddings. Although these research efforts revealed that both local and global contexts are useful in generating definitions, none of these studies exploited both contexts directly to describe unknown phrases.

In this study, we tackle a task of describing (defining) a phrase when given its local and global contexts. We present **LOG-CaD**, a neural description generator (Figure 5.1 on Page 1) to directly solve this task. Given an unknown phrase without sense definitions, our model obtains a phrase embedding as its global context by composing word embeddings while also encoding the local context. The model therefore combines both pieces of information to generate a natural language description.

Considering various applications where we need definitions of expressions, we evaluated our method with four datasets including WordNet [85] for general words, the Oxford dictionary [32] for polysemous words, Urban Dictionary [83] for rare idioms or slangs, and a newly-created Wikipedia dataset for entities.

Our contributions are as follows:

- We set up **a general task of defining phrases given their contexts**. This task is a generalization of three related tasks [85, 83, 32] and involves various situations where we need definitions of unknown phrases.
- We build **a large-scale dataset** from Wikipedia and Wikidata for the proposed task. We will release the dataset to the public as well as all of the code to promote the reproducibility of the experiments.
- We propose a method for **generating natural language descriptions for phrases with local and global contexts**.
- Empirical **results are strong**; this method achieves the state-of-the-art performance for our new dataset and the three existing datasets used in the related studies [85, 83, 32].

5.2 Context-aware Phrase Description Generation

In this section, we define our task of describing a phrase in a specific context. Given an undefined phrase $X_{trg} = \{x_i, \cdots, x_j\}$ with its context $X = \{x_1, \cdots, x_I\}$ $(1 \leq i \leq j \leq I)$, our task is to output a description $Y = \{y_1, \cdots, y_T\}$. Here, X_{trg} can be a word or a short phrase and is included in X. Y is a definition-like concrete and concise sentence that describes the X_{trg}.

For example, given a phrase "sonic boom" with its context "the shock wave may be caused by *sonic boom* or by explosion," the task is to generate a description such as "sound created by an object moving fast." If the given context has been changed to "this is the first official tour to support the band's latest studio effort, 2009's *Sonic Boom*," then the appropriate output would be "album by Kiss."

The process of description generation can be modeled with a conditional language model as

$$p(Y|X, X_{trg}) = \prod_{t=1}^{T} p(y_t|y_{<t}, X, X_{trg}). \tag{5.1}$$

5.3 Proposed: LOG-CaD: Local & Global Context-aware Description Generator

In this section, we describe our idea of utilizing local and global contexts in the description generation task, and present the details of our model.

Local & Global Contexts for Describing Unknown Phrases

When we find an unfamiliar phrase in text and it is not defined in dictionaries, how can we humans come up with its meaning? As discussed in Section 5.1, we may first try to figure out the meaning of the phrase from the immediate context, and then read through the entire document or search the web to understand implicit information behind the text. In this work, we refer to the explicit contextual information included in a given sentence with the target phrase (i.e., the X in Eq. (5.1)) as "local context," and the implicit contextual information in massive text as "global context." While both local and global contexts are crucial for humans to understand unfamiliar phrases, are they also useful for machines to generate descriptions? To verify this idea, we propose to incorporate both local and global contexts to describe an unknown phrase.

Model

Figure 5.1 on Page 1 shows an illustration of our **LOG-CaD** model. Similarly to the standard encoder-decoder model with attention [6, 60], it has a context encoder and a description decoder. The challenge here is that the decoder needs to be conditioned not only on the local context, but also on its global context. To incorporate the different types of contexts, we propose to use a gate function similar to Noraset et al. [85] to dynamically control how the global and local contexts influence the description.

We use bi-directional and uni-directional LSTMs [34] as our context encoder and description decoder (Figure 5.1), respectively. Given a sentence X and a phrase X_{trg}, the context encoder generates a sequence of continuous vectors $\boldsymbol{H} = \{\boldsymbol{h}_1 \cdots, \boldsymbol{h}_I\}$ as

$$\boldsymbol{h}_i = \text{Bi-LSTM}(\boldsymbol{h}_{i-1}, \boldsymbol{x}_i), \tag{5.2}$$

where $\boldsymbol{x}_i$ denotes the word embedding of word x_i. Then, the description decoder computes the conditional probability of a description Y with Eq. (5.1), which can be

approximated with another LSTM as

$$\boldsymbol{s}_t = \text{LSTM}(\boldsymbol{y}_{t-1}, \boldsymbol{s}'_{t-1}), \tag{5.3}$$

$$\boldsymbol{d}_t = \text{ATTENTION}(\boldsymbol{H}, \boldsymbol{s}_t), \tag{5.4}$$

$$\boldsymbol{c}_{trg} = \text{CNN}(X_{trg}), \tag{5.5}$$

$$\boldsymbol{s}'_t = \text{GATE}(\boldsymbol{s}_t, \boldsymbol{x}_{trg}, \boldsymbol{c}_{trg}, \boldsymbol{d}_t), \tag{5.6}$$

$$p(y_t|y_{<t}, X_{trg}) = \text{softmax}(\boldsymbol{W}_{s'}\boldsymbol{s}'_t + \boldsymbol{b}_{s'}), \tag{5.7}$$

where $\boldsymbol{s}_t$ is a hidden state of the decoder LSTM, and $\boldsymbol{y}_{t-1}$ is a jointly-trained word embedding of the previous output word y_{t-1}.

Considering the fact that the local context can be relatively long (e.g., around 20 words on average in the Wikipedia dataset that will be introduced in the next section), it is hard for the decoder to focus on important words in local contexts. In order to deal with this problem, the ATTENTION(·) function in Eq. (5.4) decides which words in the local context X to focus on at each time step. $\boldsymbol{d}_t$ is computed with an attention mechanism [60] as

$$\boldsymbol{d}_t = \sum_{i=1}^{T} \alpha_i \boldsymbol{h}_i, \tag{5.8}$$

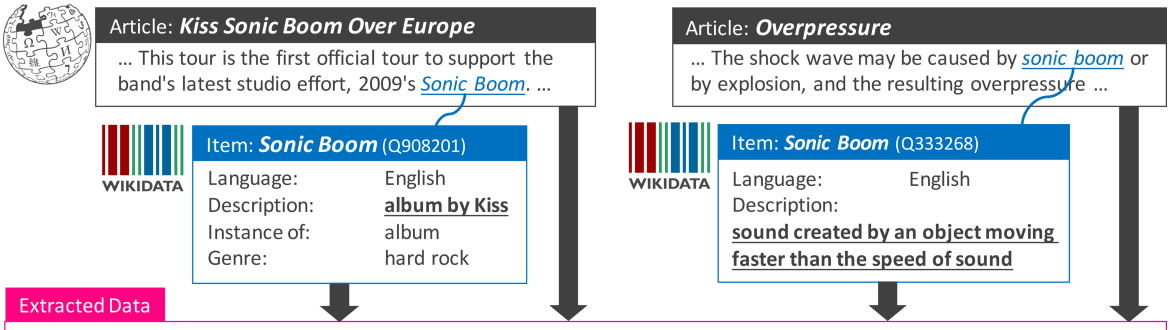
$$\alpha_i = \text{softmax}(\boldsymbol{U}_h \boldsymbol{h}_i^{\mathrm{T}} \boldsymbol{U}_s \boldsymbol{s}_t), \tag{5.9}$$

where $\boldsymbol{U}_h$ and $\boldsymbol{U}_s$ are matrices that map the encoder and decoder hidden states into a common space, respectively.

In order to capture the surface information of X_{trg}, we construct character-level CNNs (Eq. (5.5)) following [85]. Note that the input to the CNNs is a sequence of words in X_{trg}, which are concatenated with special character "_," such as "sonic_boom." Following Noraset et al. [85], we set the CNN kernels of length to 2-6 and the size to 10, 30, 40, 40, 40 respectively with a stride of 1 to obtain a 160-dimensional vector $\boldsymbol{c}_{trg}$.

In addition to the local context and the character-information, we also utilize the global context obtained from massive text. We achieve this by two different strategies proposed by Noraset et al. [85]. First, we feed a phrase embedding $\boldsymbol{x}_{trg}$ to initialize the decoder as

$$\boldsymbol{y}_0 = \boldsymbol{x}_{trg}. \tag{5.10}$$



Extracted Data

Input:	sonic boom	
Context:	#1 This tour is the first official tour to support the band's latest studio effort, 2009's **[TRG].**	#2 The shock wave may be caused by **[TRG]** or by explosion, and the resulting overpressure receives particular attention when measuring the effects of nuclear weapons or thermobaric bombs.
Description:	album by Kiss	sound created by an object moving faster than the speed of sound

Figure 5.2: Context-aware description dataset extracted from Wikipedia and Wikidata.

Here, phrase embedding $\boldsymbol{x}_{trg}$ is calculated by simply summing up all the embeddings of words that consistute the phrase X_{trg}. Note that we use a randomly-initialized vector if no pre-trained embedding is available for the words in X_{trg}.

As described in the previous section, we use both local and global contexts. In order to capture the interaction between two types of contexts and the description decoder, we adopt a GATE(·) function (Eq. (5.6)) that updates the LSTM output $\boldsymbol{s}_t$ to $\boldsymbol{s}'_t$ depending on the global context $\boldsymbol{x}_{trg}$, local context $\boldsymbol{d}_t$, and character-level information $\boldsymbol{c}_{trg}$ as

$$\boldsymbol{f}_t = [\boldsymbol{x}_{trg}; \boldsymbol{d}_t; \boldsymbol{c}_{trg}] \tag{5.11}$$
$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z[\boldsymbol{f}_t; \boldsymbol{s}_t] + \boldsymbol{b}_z), \tag{5.12}$$
$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r[\boldsymbol{f}_t; \boldsymbol{s}_t] + \boldsymbol{b}_r), \tag{5.13}$$
$$\tilde{\boldsymbol{s}}_t = \tanh(\boldsymbol{W}_s[(\boldsymbol{r}_t \odot \boldsymbol{f}_t); \boldsymbol{s}_t] + \boldsymbol{b}_s), \tag{5.14}$$
$$\boldsymbol{s}'_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{s}_t + z_t \odot \tilde{\boldsymbol{s}}_t, \tag{5.15}$$

where $\sigma(\cdot)$, $\odot$ and ; denote the sigmoid function, element-wise multiplication, and vector concatenation, respectively. $\boldsymbol{W}$ and $\boldsymbol{b}$ are weight matrices and bias terms, respectively. Here, the update gate $\boldsymbol{z}_t$ controls how much the original hidden state $\boldsymbol{s}_t$ is to be changed, and the reset gate $\boldsymbol{r}_t$ controls how much the information from $\boldsymbol{f}_t$ contributes to word generation at each time step.

5.4 Proposed: Wikipedia Dataset

Our goal is to describe rare/new expressions such as proper nouns in a variety of domains. However, among the three existing datasets, WordNet and Oxford dictionary mainly target the descriptions of relatively common words, and thus are non-ideal test beds for this goal. On the other hand, although the Urban Dictionary dataset contains descriptions of rarely-used phrases, the domain of its targeted words and phrases is limited to Internet slang.

In order to confirm that our model can generate the description of rarely-used phrases as well as words, we constructed a new dataset for context-aware phrase description generation from Wikipedia[1] and Wikidata[2] which contains a wide variety of entity descriptions with contexts. The overview of the data extraction process is shown in Figure 5.2. Each entry in the dataset consists of (1) a phrase, (2) its description, and (3) context (a sentence). For preprocessing, we applied Stanford Tokenizer[3] to the descriptions of Wikidata items and the articles in Wikipedia. Next, we removed phrases in parentheses from the Wikipedia articles, since they tend to be paraphrasing in other languages and work as noise. To obtain the contexts of each item in Wikidata, we extracted the sentence which has a link referring to the item through all the first paragraphs of Wikipedia articles and replaced the phrase of the links with a special token **[TRG]**. Wikidata items with no description or no contexts are ignored. This utilization of links makes it possible to resolve the ambiguity of words and phrases in a sentence without human annotations, which is a major advantage of using Wikipedia. Note that we used only links whose anchor texts are identical to the title of the Wikipedia articles, since the users of Wikipedia sometimes link mentions to related articles.

5.5 Experiments

We evaluate our method by applying it to describe words in WordNet [71] and Oxford Dictionary,[4] phrases in Urban Dictionary[5] and Wikidata.[6] For all of these datasets, a given word or phrase has an inventory of senses with corresponding

[1]https://dumps.wikimedia.org/enwiki/20170720/
[2]https://dumps.wikimedia.org/wikidatawiki/entities/20170802/
[3]https://nlp.stanford.edu/software/tokenizer.shtml
[4]https://en.oxforddictionaries.com/
[5]https://www.urbandictionary.com/
[6]Dataset will be made available upon publication.

definitions and usage examples. These definitions are regarded as ground-truth descriptions.

5.5.1 Settings

Datasets

To evaluate our model on the word description task on WordNet, we followed Noraset et al. [85] and extracted data from WordNet[7] using the dict-definition[8] toolkit. Each entry in the data consists of three elements: (1) a word, (2) its definition, and (3) a usage example of the word. We split this dataset to obtain Train, Validation, and Test sets. If a word has multiple definitions/examples, we treat them as different entries. Note that the words are mutually exclusive across the three sets. The only difference between our dataset and theirs is that we extract the tuples only if the words have their usage examples in WordNet. Since not all entries in WordNet have usage examples, our dataset is a small subset of Noraset et al. [85].

In addition to WordNet, we use the Oxford Dictionary following Gadetsky et al. [32], the Urban Dictionary following Ni and Wang [83] and our Wikipedia dataset described in the previous section. Table 5.1 and Table 5.2 show the properties and statistics of the new dataset and the three existing datasets, respectively.

To simulate a situation in a real application where we might not have access to global context for all phrases, we did not train domain-specific word embeddings on each domain. Instead, we use the same pre-trained CBOW[9] vectors as global context following previous work [85, 32]. If the expression to be described consists of multiple words, its phrase embedding is calculated by simply summing up all the CBOW vectors of words in the phrase, such as "sonic" and "boom." (See Figure 5.1 on Page 1). If pre-trained CBOW embeddings are unavailable, we instead use a special **[UNK]** vector (which is randomly initialized with a uniform distribution) as word embeddings. Note that our pre-trained embeddings only cover 26.79% of the words in the expressions to be described in our Wikipedia dataset, while it cover all words in WordNet dataset (See Table 5.2). Even if no reliable word embeddings are available, all models can capture the character information through character-level CNNS (See Figure 5.1 on Page 1).

[7] https://wordnet.princeton.edu/

[8] https://github.com/NorThanapon/dict-definition

[9] `GoogleNews-vectors-negative300.bin.gz` at https://code.google.com/archive/p/word2vec/

Corpus	# Phrases	# Entries	Length of Context	Length of Description
WordNet				
Train	7,938	13,883	5.81	6.61
Valid	998	1,752	5.64	6.61
Test	1,001	1,775	5.77	6.85
Oxford Dictionary				
Train	33,128	97,855	17.74	11.02
Valid	8,867	12,232	17.80	10.99
Test	8,850	12,232	17.56	10.95
Urban Dictionary				
Train	190,696	411,384	10.89	10.99
Valid	26,876	57,883	10.86	10.95
Test	26,875	38,371	11.14	11.50
Wikipedia				
Train	151,995	887,455	18.79	5.89
Valid	8,361	44,003	19.21	6.31
Test	8,397	57,232	19.02	6.94

Table 5.1: Statistics of the word/phrase description datasets.

Models

We implemented four methods: (1) **Global** [85], (2) **Local** [83] with CNN, (3) **I-Attention** [32], and our proposed model, (4) **LOG-CaD**. The **Global** model is our reimplementation of the strongest model (S + G + CH) in Noraset et al. [85]. It can access the global context of a phrase to be described, but has no ability to read the local context. The **Local** model is the reimplementation of the best model (dual encoder) in Ni and Wang [83]. In order to make a fair comparison of the effectiveness of local and global contexts, we slightly modify the original implementation by Ni and Wang [83]; as the character-level encoder in the **Local** model, we adopt CNNs that are exactly the same as the other two models instead of the original LSTMs. The **I-Attention** is our reimplementation of the best model (S + I-Attention) in Gadetsky et al. [32]. Similar to our model, it uses both local and global contexts. Unlike our model, however, their model cannot directly use the local context to predict the words in descriptions. This is because the **I-Attention** model indirectly uses the

Corpus	Domain	Inputs	Cov. emb.
WordNet	General	words	100.00%
Oxford Dictionary	General	words	83.04%
Urban Dictionary	Internet slangs	phrases	21.00%
Wikipedia	Proper-nouns	phrases	26.79%

Table 5.2: Domains, expressions to be described, and the coverage of pre-trained embeddings of the expressions to be described.

	Global	Local	I-Attn.	Proposed
# Layers of Enc-LSTMs	-	2	2	2
Dim. of Enc-LSTMs	-	600	600	600
Dim. of Attn. vectors	-	300	300	300
Dim. of input word emb.	300	-	300	300
Dim. of char. emb.	160	160	160	160
# Layers of Dec-LSTMs	2	2	2	2
Dim. of Dec-LSTMs	300	300	300	300
Vocabulary size	10k	10k	10k	10k
Dropout rate	0.5	0.5	0.5	0.5

Table 5.3: Hyperparameters of the models

local context only to filter out unrelated information in phrase embeddings. All four models (Table 5.3) are implemented with the PyTorch framework.[10]

5.5.2 Results

Automatic Evaluation

Table 5.4 shows the BLEU [88] scores of the output descriptions. We can see that the **LOG-CaD** model consistently outperforms the three baselines in all four datasets. This result indicates that using both local and global contexts helps describe the unknown words/phrases correctly. While the **I-Attention** model also uses local and global contexts, its performance was always lower than the **LOG-CaD** model. This result shows that using local context to predict description is more effective than using it to disambiguate the meanings in global context.

In particular, the low BLEU scores of **Global** and **I-Attention** models on Wikipedia dataset suggest that it is necessary to learn to ignore the noisy information in global

[10] `http://pytorch.org/`

Model	WordNet	Oxford	Urban	Wikipedia
Global	24.10	15.05	6.05	44.77
Local	22.34	17.90	9.03	52.94
I-Attention	23.77	17.25	10.40	44.71
LOG-CaD	**24.79**	**18.53**	**10.55**	**53.85**

Table 5.4: BLEU scores on four datasets.

Model	Annotated score
Local	2.717
LOG-CaD	**3.008**

Table 5.5: Averaged human annotated scores on Wikipedia dataset.

context if the coverage of pre-trained word embeddings is extremely low (see the third and fourth rows in Table 5.2on Page 5). We suspect that the Urban Dictionary task is too difficult and the results are unreliable considering its extremely low BLEU scores and high ratio of unknown tokens in generated descriptions.

Manual Evaluation

To compare the proposed model and the strongest baseline in Table 5.4 (i.e., the **Local** model), we performed a human evaluation on our dataset. We randomly selected 100 samples from the test set of the Wikipedia dataset and asked three native English speakers to score the output descriptions from 5 (correct) to 1 (wrong). The averaged scores are reported in Table 5.5. Pair-wise bootstrap resampling test [47] for the annotated scores has shown that the superiority of **LOG-CaD** over the **Local** model is statistically significant ($p < 0.01$).

Qualitative Analysis

Table 5.6 and Table 5.7 show the examples of a phrase in Wikipedia and a word in the WordNet dataset, respectively. When comparing the two datasets shown in the two tables, the quality of generated descriptions of Wikipedia dataset is significantly better than that of WordNet dataset. The main reason for this result is that the size of training data of the Wikipedia dataset is 64x larger than the WordNet dataset (See Table 5.1).

Input:	daniel o'neill	
Context:	#1	#2
	after being enlarged by publisher **daniel o'neill** it was reportedly one of the largest and most prosperous newspapers in the united states.	in 1967 he returned to belfast where he met fellow belfast artist **daniel o'neill**.
Reference:	american journalist	irish artist
Global:	american musician	
Local:	american publisher	british musician
I-Attention:	american musician	american musician
LOG-CaD:	american writer	british musician

Table 5.6: Descriptions for a phrase in Wikipedia.

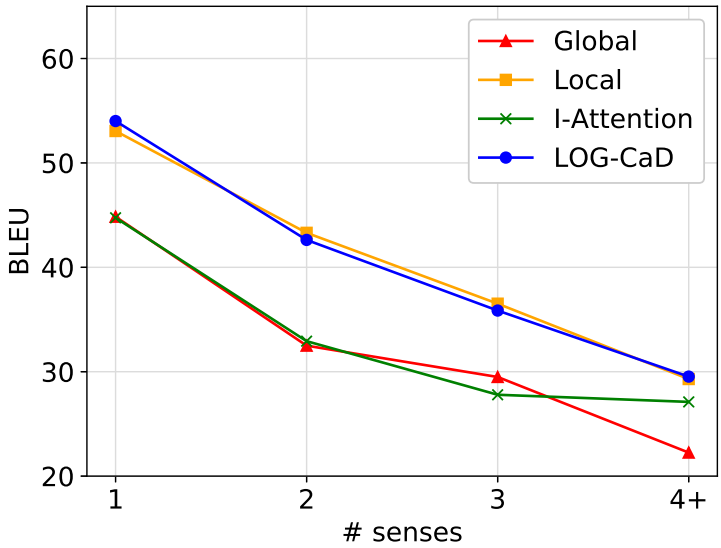


Figure 5.3: Number of senses of the phrase.

For all examples in both datasets in Table 5.6 and Table 5.7, the **Global** model can only generate a single description for each input phrase because it cannot access any local context. In the Wikipedia dataset, both the **Local** and **LOG-CaD** models can describe the word/phrase considering its local context. For example, both the **Local** and **LOG-CaD** models could generate "american" in the description for "daniel o'neill" given "united states" in Context #1, while they could generate "british" given "belfast" in Context #2. On the other hand, the **I-Attention** model could not

Input:	waste	
Context:	#1	#2
	if the effort brings no compensating gain it is a **waste**	We **waste** the dirty water by channeling it into the sewer
Reference:	useless or profitless activity	to get rid of
Global:	to give a liquid for a liquid	
Local:	a state of being assigned to a particular purpose	to make a break of a wooden instrument
I-Attention:	a person who makes something that can be be be done	to remove or remove the contents of
LOG-CaD:	a source of something that is done or done	to remove a liquid

Table 5.7: Descriptions for a word in WordNet.

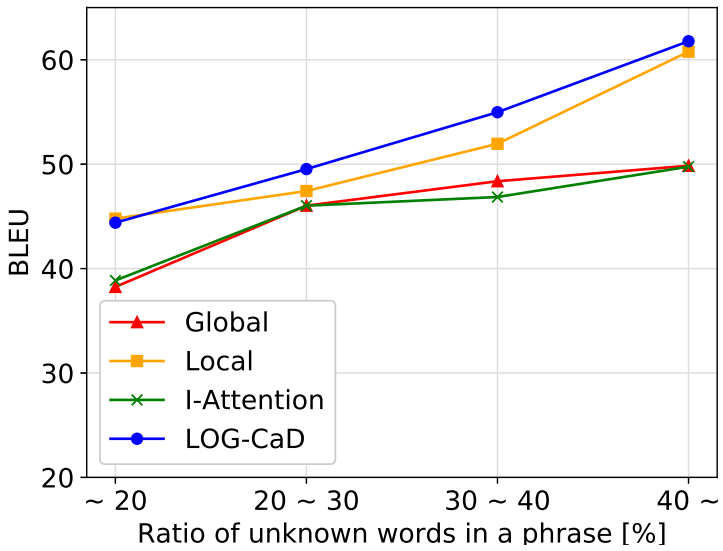


Figure 5.4: Unknown words ratio in the phrase.

describe the two phrases, taking into account the local contexts. We will present an analysis of this phenomenon in the next section.

5.6 Discussion

In this section, we present analyses on how the local and global contexts contribute to the description generation task. First, we discuss how the local context helps the

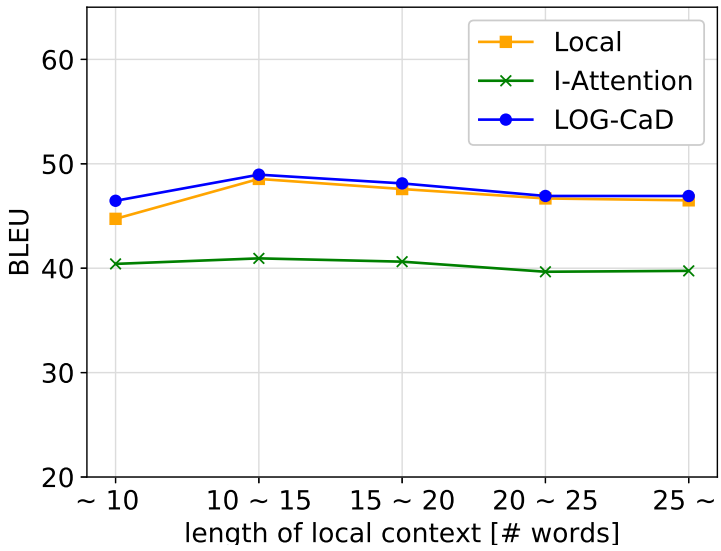


Figure 5.5: Impact of various parameters of a phrase to be described on BLEU scores of the generated descriptions.

models to describe a phrase. Then, we analyze the impact of global context under the situation where local context is unreliable.

5.6.1 How do the models utilize *local* contexts?

Local context helps us (1) disambiguate polysemous words and (2) infer the meanings of unknown expressions. Can machines also utilize the local context? In this section, we discuss the two roles of local context in description generation.

Considering that the pre-trained word embeddings are obtained from word-level co-occurrences in a massive text, more information is mixed up into a single vector as the more senses the word has. While Gadetsky et al. [32] designed the **I-Attention** model to filter out unrelated meanings in the global context given local context, they did not discuss the impact the number of senses has on the performance of definition generation. To understand the influence of the ambiguity of phrases to be defined on the generation performance, we did an analysis on our Wikipedia dataset. Figure 5.5(a) shows that the description generation task becomes harder as the phrases to be described become more ambiguous. In particular, when a phrase has an extremely large number of senses, (i.e., #senses ≥ 4), the **Global** model drops its performance significantly. This result indicates that the local context is necessary to disambiguate the meanings in the global context.

As shown in Table 5.2 on Page 5, a large proportion of the phrases in our Wikipedia dataset includes unknown words (i.e., only 26.79% of words in the phrases have their pre-trained embeddings). This fact indicates that the global context in this dataset is not fully reliable. Then our next question is, how does the lack of information from global context affect the performance of phrase description? Figure 5.5(b) shows the impact of unknown words in the phrases to be described on the performance. As we can see from the result, the advantage of **LOG-CaD** and **Local** models over **Global** and **I-Attention** models becomes larger as the unknown words increases. This result suggests that we need to fully utilize local contexts especially in practical applications where the phrases to be defined have many unknown words. Here, Figure 5.5(b) also shows a counterintuitive phenomenon that BLEU scores increase as the ratio of unknown words in a phrase increase. This is mainly because unknown phrases tend to be persons' names such as writers, actors, or movie directors. Since these entities have fewer ambiguities, they can be described in extremely short sentences that are easy for our method to decode (e.g., "finnish writer" or "american television producer").

5.6.2 How do the models utilize *global* contexts?

As discussed earlier, local contexts are important to describe unknown expressions, but how about global contexts? Assuming a situation where we cannot obtain much information from local contexts (*e.g.*, infer the meaning of "boswellia" from a short local context "Here is a boswellia"), global contexts should be essential to understand the meaning. To confirm this hypothesis, we analyzed the impact of the length of local contexts on BLEU scores. Figure 5.5(c) shows that when the length of local context is extremely short ($l \leq 10$), the **LOG-CaD** model becomes much stronger than the **Local** model. This result indicates that not only local context but also global context help models describe the meanings of phrases.

5.6.3 Differences between the Description Generation Task and the Word Sense Disambiguation Task

As discussed in Section 2.3, our task of describing phrases is closely related to word sense disambiguation (WSD) [76], which identifies a pre-defined sense for the target word with its context. The most significant difference between the two tasks is that the pre-defined senses are given in WSD, which are not available in the description generation task. To investigate the applicability of our model to the

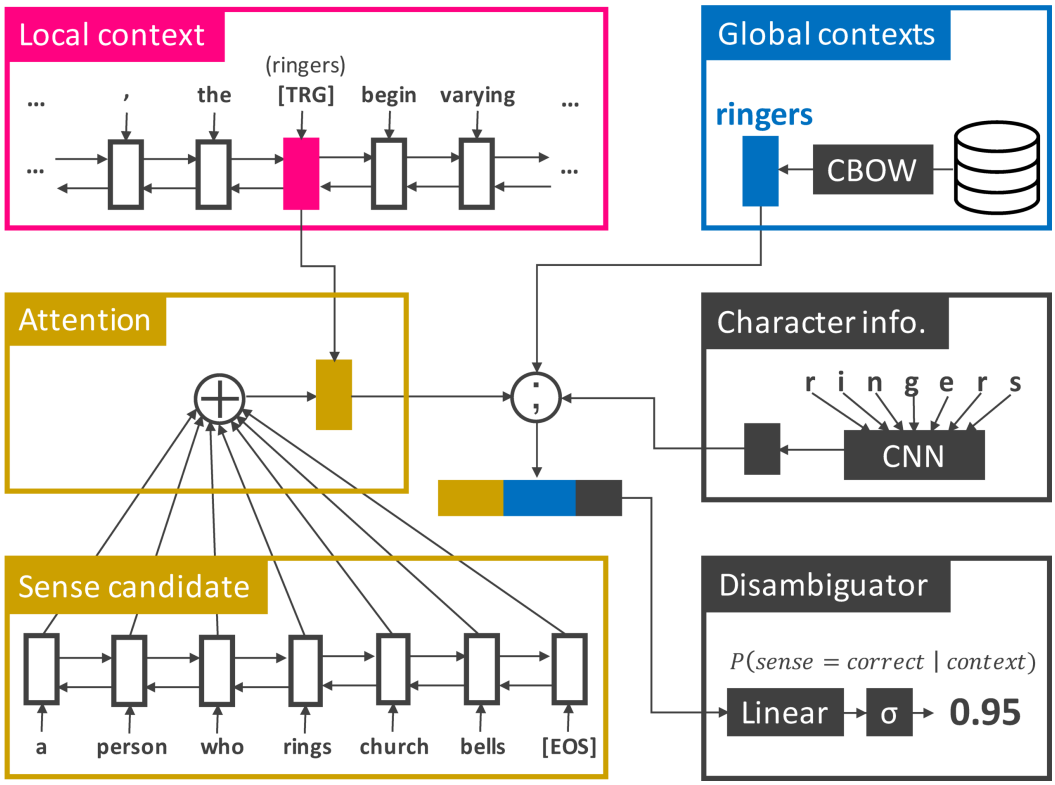


Figure 5.6: The modified version of the proposed model for WSD task

WSD, we conducted an experiment on knowledge-based WSD. In the following, we describe the models, datasets, and the results of this experiment.

Model

While the output of the description generation model is a sequence of the discrete symbols (i.e., words), the output of the WSD task is a continuous score that represents to what extent the given sense is appropriate to the specific context. Thus, we need to modify our **LOG-CaD** model as shown in Figure 5.6. Firstly, we add another bi-LSTM encoder that allows the model to read a candidate definition in the sense inventories. Secondly, we remove the LSTM decoder and adopt a linear layer as the output layer that predicts a single value. This encoder maps the candidate definition into a single continuous vector. Finally, since the supervised WSD task can be seen as a ranking problem, we use the margin ranking loss instead of the cross-entropy loss to optimize the model.

Data source	#Sentences	#Tokens	#Sense types	#Word types
Senseval-2	242	5,766	1,335	1,093
Senseval-3	352	5,541	1,167	977
SemEval-07	135	3,201	375	330
SemEval-13	306	8,391	827	751
SemEval-15	138	2,604	659	512
SemCor	37,176	802,443	33,362	22,436

Table 5.8: Statistics of the WSD datasets after the standarization proposed by Raganato et al. [92].

Method	F-1 measure
Random baseline	39.0
Proposed	44.7
UKB [2, 3]	**67.3**

Table 5.9: F-1 measure on the Senseval/SemEval dataset.

Dataset

Following the recent proposed evaluation framework on WSD [92], we use SemCor [72] dataset as training set and the Senseval/SemeEval dataset as test set. The SemCor is the largest corpus manually annotated with WordNet senses [92]. Our test set consists of the five datasets from Senseval2 [21], Senseval3 [100], SemEval07 [90], SemEval13 [77], and SemEval15 [73], and is standardized to the same format. The statistics of the dataset are shown in Table 5.8.

Result

Table 5.9 shows the performance of our model and UKB[2, 3], which is the state-of-the-art knoledge-based WSD system. Although the proposed model performed better than the random baseline, it is significantly worse than the UKB. Note that the UKB utilizes the WordNet graph and contexts, while our system can only access the contexts. This result suggests that the **LOG-CaD** cannot replace the WSD systems, but the two methods are complementary to each other. If the word to be defined is included in a hand-made knowledge base, using the WSD systems would help human understanding better than our system. On the other hand, **LOG-CaD** can be utilized in other situations where (1) we need to understand the meanings of new words/phrases that are not included in the knowledge base, or (2) the WSD system

cannot select the available sense with high confidence, or (3) we discover a new usage of the existing expressions with the word sense induction [87] methods.

5.7 Chapter Summary

In this chapter, we examined a method to generate a natural language description for an unknown phrase with a specific context. The first contribution of this work is to construct a Wikipedia-based description generation dataset. Compared to the existing datasets for definition generation, our newly constructed dataset has three advantages: large, diverse, and versatile. It contains 989k entries, which is the largest description-generation dataset in the world. Since we built the dataset using Wikipedia and Wikidata, the dataset covers lots of domains. It should also be noted that this dataset covers phrases as well as words, while most previous work focused only on unknown words.

The second contribution of this work is to present **LOG-CaD**, a state-of-the-art description generation model. **LOG-CaD** is a variant of encoder-decoder models that capture the given local context by an encoder and global contexts by the target phrase's embedding induced from a massive text. In the experiments on three existing datasets and our newly built Wikipedia dataset, our model outperformed the strongest baselines by +1.38 BLEU score (averaged over four datasets).

Chapter 6

Conclusion

In this thesis, we explored methods to help human understanding of multilingual text. As described in Section 1, there exist two barriers that avoid humans from understanding the multilingual text: the language differences and the domain differences. To overcome the two barriers, we started from the area of intersection area of the language and domain differences: the out-of-domain problem of machine translation (Chapter 3). Next, we tackled the language barrier between English and Japanese, a language pair which have significant differences in their grammars and vocabularies (Chapter 4). Finally, we explored a way to solve the domain differences between text and the readers who are not the experts in the domain of the text (Chapter 5) In the following sections, we summarize the solutions for the above problems and the achievements obtained.

6.1 Accurate and Instant Translation Model Adaptation for Statistical Machine Translation

In order to obtain fresh and diverse information from multilingual text, we need a machine translation system that can translate text in any domain. Several domain adaptation methods had been proposed to achieve this goal. Most work on domain adaptation for machine translation has been focusing on a scenario where a small or pseudo in-domain parallel corpus is available. However, in actual scenarios where users want to exploit machine translation for multilingual text, it is impractical to prepare all the in-domain parallel data since the target domains are usually unknown and they vary among users and situations.

In Chapter 3, we presented a practical domain adaptation method for SMT. Since it does not require any in-domain parallel data, our method can be used to help multilingual text understanding, where the target domains are unknown and vary depending on users and situations. The key idea of the adaptation method is to leverage a cross-lingual projection of word semantic representations to obtain a translation model for out-of-vocabulary words in SMT. Assuming monolingual corpora for the source and target languages, we induce vector-based semantic representations of words and obtain a projection (translation matrix) from source-language semantic representations into the target-language semantic space.

Since the existing methods for vector projection lack in accuracy, we first designed a projection model that exploits two types of translatable context pairs, which are taken from the training data and guessed by surface-level similarity. In the experiments on word translation task between four languages (including English, Spanish, Japanese, and Chinese), this projection model outperformed the previous methods by +8.1 points in precision. We then apply the obtained translation matrix to find translation candidates of oov words and use the cosine similarity to induce the translation probability. Experimental results on domain adaptation from a Kyoto-related domain to a recipe domain confirmed that our method improved BLEU by 0.5-1.5 and 0.1-0.2 for en-ja and ja-en translations, respectively.

This adaptation pipeline freed us from the constraint that the in-domain parallel data was needed. However, there still exist problems that were not solved in this work. First, the performance of machine translation is low especially in the language pairs whose word orders are significantly different (e.g., English and Japanese). This problem might be overcome by (1) using the state-of-the-art machine translation algorithm (i.e., NMT) and (2) modeling the differences of the language structures in its framework. Second, the unknown words for *humans* have not been dealt with. The oov words we focused on in this work are unknown words for *machines*, which are caused by the domain differences between the train and test data. The unknown words for *humans*, which are caused by the lack of domain knowledge of the readers, might be solved by describing the meanings of the words in natural language. Our efforts to tackle the above two problems will be described in Chapter 4 and Chapter 5, respectively.

6.2 Chunk-based Decoder for Neural Machine Translation

The domain adaptation method in Chapter 3 improved the quality of the translation for multi-domain text. From the viewpoint of machine translation, there are two limitations to this work. First, since the method can only cope with the oov words, it does not improve performance if the vocabulary difference is not the major problem (which was described in Section 3.6.2). The translation quality depends not only on the domain differences between train/test data but also on the structural differences between source/target languages. Second, the overall performance of SMT is low compared to the state-of-the-art NMT models.

In order to directly tackle these problems, in Chapter 4, we tried to improve the performance of NMT. In this work, we focus on the translation from English to Japanese, a language pair that has significant differences in their vocabulary and syntactic structures. As the attention mechanism in NMT plays a similar role to the translation model in phrase-based SMT, our proposed chunk-based decoders are intended to capture the notion of chunks in chunk-based (or phrase-based) SMT. We designed three models that have hierarchical RNN-like architectures, each of which consists of a word-level decoder and a chunk-level decoder. The performed experiments on the WAT '16 English-to-Japanese translation task showed that our best model outperforms the conventional word-based decoder strongest baselines by +0.93 BLEU score and by +0.57 RIBES score.

In the paradigm of conventional NMT, machine translation had been formulated as a problem of sequence-to-sequence mapping. Here, the outputs from the mapping functions were usually word or subword sequences, which did not consider any linguistic structure. Leveraging the linguistic structures in the target language is a hard problem since the target sentences cannot be observed in test time. The main contribution of this work is the methodology of considering the structures of the target language. The effectiveness of leveraging target structures has been confirmed in the work published concurrently to or later than our work[4, 24, 82, 107, 111, 112, 120].

6.3 Learning to Describe Phrases with Local and Global Contexts

As described in Chapter 1, we need to extract information from the diverse text is written in several languages. The domain adaptation method in Chapter 3 approached this problem from the viewpoint of machine translation. It enabled the SMT system to translate oov words that are not included in the train data. However, from the perspective of human understanding, we still have problems other than translation. If we readers are not the experts of a text, we may find several words or phrases whose senses we are unsure of even if the text is written in our native language. These unknown expressions prevent our effective understanding of the text because we need to consult dictionaries while reading. In addition, many domain-specific terms and new entities may not be included in the dictionaries.

In Chapter 5, we examined a method to describe the meaning of unknown phrases automatically. We first set up a task of generating a natural language description for an unknown word/phrase in several domains, aiming to help us acquire the senses of the unknown expressions when reading a multi-domain text. Compared to the existing datasets for definition generation, our newly constructed dataset has three advantages: large, diverse, and versatile. It contains 989k entries, which is the largest description-generation dataset in the world. Since we built the dataset using Wikipedia and Wikidata, the dataset covers lots of domains. It should also be noted that this dataset covers phrases as well as words, while most previous work focused only on unknown words. We approached this task by using a variant of encoder-decoder models that capture the given local context by an encoder and global contexts by the target word's embedding induced from a massive text. We performed experiments on three existing datasets and the one newly built from Wikipedia. Our proposed model achieved state-of-the-art performances in all of the four datasets. In particular, when tested on the multi-domain scenario, it performed much better than the strongest baseline (+0.9 in BLEU, +0.3 in a 5-level human annotated score).

The main contribution of this work was twofold: (1) to present a practical dataset for description generation task in the multi-domain scenario, and (2) to propose a state-of-the-art model for description generation. While the dataset proposed in the previous work was either small or limited to specific domains, it was difficult to use them to build a practical description generation model. The domain diversity of the domains and the massive size of our presented dataset have enabled us to

build a more practical model for multi-domain text. Besides, we showed that our description generation method performs better than any other models which had been proposed previously. This result indicates the effectiveness of our method to help humans' understanding their unfamiliar words/phrases.

6.4 Contributions to Humans' Understanding of Multilingual Text

At the beginning of this thesis, we discussed the problem of multilingual text understanding. The multilingual text is written in several languages, and its contents vary in several domains. Two requirements for the understandable text were defined: the high translation quality of the text written in unfamiliar languages, and the high comprehensibility of the terms in unfamiliar domains.

In the following, we discuss how this thesis contributes to solving this problem. This thesis had presented the methodologies of (1) accurate and instant domain adaptation for SMT (Chapter 3), (2) chunk-aware decoding for NMT(Chapter 4), and (3) description generation in various domains(Chapter 5) . From the viewpoint of improving translation quality, we explored (1) and (2). These two methodologies proposed here are not competing but rather complementary to each other. While the phrase-by-phrase decoding in conventional SMT had significantly contributed to improving translation performance, there was no way to apply it to the current NMT models. In work (2), we shed light on this issue and presented a method to introduce the phrase structure in the neural decoders. On the other hand, even the state-of-the-art NMT models suffer from the domain differences between the train and test data [50]. Aiming to solve this problem, in work (1), we explored an adaptation method that does not require any in-domain text and contributed to the out-of-domain translation.

From the perspective of the comprehensibility of out-of-domain terms, we proposed (3). The work (3) tackled the problem of domain differences of multilingual text from a different aspect from (1). While (1) focused on resolving the domain differences between the dataset (i.e., train and test data), (3) focused on the domain differences between text and readers (e.g., a situation where computer science majored student reads a text in medical domain). Since we intended to directly help humans' understanding in (3), we took an approach of using natural language to describe the words/phrases that are unfamiliar to the readers.

To conclude, this thesis tackled the problems in assisting humans' understanding of the multilingual text. Since both the language barriers and domain barriers are serious problems to be solved, we presented three approaches to cope with them. The three approaches presented do not conflict with each other, but are focusing on the different sides of the problems. We expect that this thesis will provide a promising future direction for research of multilingual text processing.

6.5 Future Work

This thesis contributed to the research area of multilingual text understanding by improving machine translation and helping human understanding. However, there remain important issues to be addressed in this research area. In the following, we summarize the future work of multilingual text understanding according to its topic.

6.5.1 Domain Adaptation for Machine Translation

Domain adaptation is the key technology when machine translation is used to translate the multi-domain text. Despite the recent successes of NMT, its poor performance in out-of-domain settings is one of the most severe problems to be solved [50]. There are two difficulties in the domain adaptation in multilingual text understanding: the target domains are unknown and vary. In this thesis, we tried to resolve the difference of vocabulary between different domains without using in-domain parallel data. Besides the vocabulary problem, the differences of topic, genre, style, level of formality of the text are also the essential factors to be considered into the machine translation models.

6.5.2 Neural Machine Translation

The largest changes of NMT from SMT is that it maps the words, phrases, and sentences into continuous vector spaces rather than directly treat them as discrete symbols. This property of NMT led us to multilingual NMT [28, 43, 58], the methodologies to perform translation for several language pairs with a single model. While the multilingual NMT is a promising approach towards multilingual text understanding, they simply treat the input and output text as sequences of word or subword. Can the linguistic structures of the source and target languages help improve translation performance?

How can we incorporate those features into the framework of multilingual NMT? These are the essential topics to be addressed in the future.

6.5.3 Sense Identification for Unknown/New Expressions

The presented task of describing unknown phrases is closely related to word sense disambiguation (WSD) [76], which identifies a pre-defined sense for the target word with its context. Since it requires a substantial amount of training data for disambiguation, it cannot handle expressions that are not registered in the dictionary. Our description generation task avoids this difficulty by directly generating descriptions for phrases or words, and also allows us to flexibly tailor a fine-grained definition for the specific context. However, the quality of the generated descriptions is much worse than the definitions written by a human.

In this point of view, we may combine a WSD system with our description generator. For the known expressions, we can utilize the WSD to output a more accurate definition. If an unknown expression or a known expression with new usage is the input, we take advantage of our method to provide a description for any input. To achieve this goal, we need an accurate classifier that predicts how likely the expression is used in a new meaning. This will also be an interesting research direction.

Bibliography

[1] Abney, S. P. (1991). Parsing by chunks. In *Principle-based parsing*, pages 257–278. Springer.

[2] Agirre, E., Lopez de Lacalle, O., and Soroa, A. (2018). The risk of sub-optimal use of open source nlp software: Ukb is inadvertently state-of-the-art in knowledge-based wsd. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 29–33. Association for Computational Linguistics.

[3] Agirre, E., López de Lacalle, O., and Soroa, A. (2014). Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 40(1):57–84.

[4] Aharoni, R. and Goldberg, Y. (2017). Towards string-to-tree neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 132–140. Association for Computational Linguistics.

[5] Androutsopoulos, I. and Malakasiotis, P. (2010). A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38:135–187.

[6] Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *Proceedings of the Third International Conference on Learning Representations (ICLR)*.

[7] Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 238–247.

[8] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

[9] Bottou, L. (2004). Stochastic learning. In *Advanced lectures on machine learning*, pages 146–168. Springer.

[10] Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. (2007). Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867.

[11] Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

[12] Chandar A P, S., Lauly, S., Larochelle, H., Khapra, M., Ravindran, B., Raykar, V. C., and Saha, A. (2014). An autoencoder approach to learning bilingual word representations. In *Advances in Neural Information Processing Systems 27 (NIPS)*, pages 1853–1861.

[13] Chen, Y. (2012). Dictionary use and vocabulary learning in the context of reading. *International Journal of Lexicography*, 25(2):216–247.

[14] Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of the Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST)*, pages 103–111.

[15] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

[16] Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

[17] Connor, M. and Roth, D. (2007). Context sensitive paraphrasing with a global unsupervised classifier. In *Proceedings of the 18th European Conference on Machine Learning (ECML)*, pages 104–115.

[18] Costa-Jussà, M. R. (2015). Domain adaptation strategies in statistical machine translation: a brief overview. *The Knowledge Engineering Review*, 30(05):514–520.

[19] Cromieres, F., Chu, C., Nakazawa, T., and Kurohashi, S. (2016). Kyoto university participation to WAT 2016. In *Proceedings of the Third Workshop on Asian Translation (WAT)*, pages 166–174.

[20] Daume III, H. and Jagarlamudi, J. (2011). Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 407–412.

[21] Edmonds, P. and Cotton, S. (2001). SENSEVAL-2: Overview. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5.

[22] Eriguchi, A., Hashimoto, K., and Tsuruoka, Y. (2016a). Character-based decoding in tree-to-sequence attention-based neural machine translation. In *Proceedings of the Third Workshop on Asian Translation (WAT)*, pages 175–183.

[23] Eriguchi, A., Hashimoto, K., and Tsuruoka, Y. (2016b). Tree-to-sequence attentional neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 823–833.

[24] Eriguchi, A., Tsuruoka, Y., and Cho, K. (2017). Learning to parse and translate improves neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 72–78. Association for Computational Linguistics.

[25] Erk, K. (2006). Unknown word sense detection as outlier detection. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (NAACL)*, pages 128–135.

[26] Erk, K. (2012). Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.

[27] Faruqui, M. and Dyer, C. (2014). Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 462–471.

[28] Firat, O., Cho, K., and Bengio, Y. (2016). Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 866–875. Association for Computational Linguistics.

[29] Firth, J. R. (1957). A synopsis of linguistic theory. *Studies in Linguistic Analysis*, pages 1–32.

[30] Fraser, C. A. (1998). The role of consulting a dictionary in reading and vocabulary learning. *Canadian Journal of Applied Linguistics*, 2(1-2):73–89.

[31] Fung, P. (1998). A statistical view on bilingual lexicon extraction: From parallel corpora to non-parallel corpora. In *Proceedings of the Third Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 1–17.

[32] Gadetsky, A., Yakubovskiy, I., and Vetrov, D. (2018). Conditional generators of words definitions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 266–271.

[33] Gao, Q. and Vogel, S. (2008). Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57.

[34] Gers, F. A., Schmidhuber, J., and Cummins, F. (1999). Learning to forget: Continual prediction with lstm. In *Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN)*, pages 850–855.

[35] Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. (2013). Maxout networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 1319–1327.

[36] Gouws, S. and Søgaard, A. (2015). Simple task-specific bilingual word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technology (NAACL-HLT)*, pages 1386–1390.

[37] Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

[38] Hashimoto, S. (1934). *Kokugoho Yosetsu*. Meiji Shoin.

[39] Hermann, K. M. and Blunsom, P. (2014). Multilingual models for compositional distributed semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 58–68.

[40] Irvine, A., Morgan, J., Carpuat, M., III, H. D., and Munteanu, D. (2013a). Measuring machine translation errors in new domains. *Transactions of the Association for Computational Linguistics*, 1:429–440.

[41] Irvine, A., Quirk, C., and Daumé III, H. (2013b). Monolingual marginal matching for translation model adaptation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1077–1088.

[42] Isozaki, H., Hirao, T., Duh, K., Sudoh, K., and Tsukada, H. (2010). Automatic evaluation of translation quality for distant language pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 944–952.

[43] Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. (2017). Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

[44] Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1700–1709.

[45] Kim, J. D., Brown, R. D., and Carbonell, J. G. (2010). Chunk-based EBMT. In *Proceedings of the 14th workshop of the European Association for Machine Translation (EAMT)*.

[46] Klementiev, A., Irvine, A., Callison-Burch, C., and Yarowsky, D. (2012). Toward statistical machine translation without parallel corpora. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 130–140.

[47] Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 388–395.

[48] Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 177–180.

[49] Koehn, P. and Knight, K. (2002). Learning a translation lexicon from monolingual corpora. In *Proceedings of ACL Workshop on Unsupervised lexical acquisition*, pages 9–16.

[50] Koehn, P. and Knowles, R. (2017). Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation (WNMT)*, pages 28–39. Association for Computational Linguistics.

[51] Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 48–54.

[52] Krause, J., Johnson, J., Krishna, R., and Fei-Fei, L. (2016). A hierarchical approach for generating descriptive image paragraphs. In *arXiv:1611.06607 [cs.CV]*.

[53] Lau, J. H., Cook, P., McCarthy, D., Gella, S., and Baldwin, T. (2014). Learning word sense distributions, detecting unattested senses and identifying novel senses using topic models. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 259–270.

[54] Lazaridou, A., Dinu, G., and Baroni, M. (2015). Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 270–280.

[55] Lembersky, G., Ordan, N., and Wintner, S. (2012). Adapting translation models to translationese improves SMT. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 255–265.

[56] Li, J., Luong, M.-T., and Jurafsky, D. (2015). A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 1106–1115.

[57] Lin, R., Liu, S., Yang, M., Li, M., Zhou, M., and Li, S. (2015). Hierarchical recurrent neural network for document modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 899–907.

[58] Lu, Y., Keung, P., Ladhak, F., Bhardwaj, V., Zhang, S., and Sun, J. (2018). A neural interlingua for multilingual machine translation. In *Proceedings of the Third Conference on Machine Translation (WMT)*, pages 84–92. Association for Computational Linguistics.

[59] Lund, K. and Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208.

[60] Luong, M.-T. and Manning, C. D. (2016). Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1054–1063.

[61] Luong, M.-T., Sutskever, I., Le, Q., Vinyals, O., and Zaremba, W. (2015a). Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the Seventh International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 11–19.

[62] Luong, T., Pham, H., and Manning, C. D. (2015b). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421.

[63] Madnani, N. and Dorr, B. J. (2010). Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387.

[64] Mansour, S. and Ney, H. (2014). Unsupervised adaptation for statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 457–465.

[65] Mathur, P., Keseler, F. B., Venkatapathy, S., and Cancedda, N. (2014). Fast domain adaptation of SMT models without in-domain parallel data. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*, pages 1114–1123.

[66] Max, A. (2009). Sub-sentencial paraphrasing by contextual pivot translation. In *Proceedings of the 2009 Workshop on Applied Textual Inference*, pages 18–26.

[67] Max, A., Bouamor, H., and Vilnat, A. (2012). Generalizing sub-sentential paraphrase acquisition across original signal type of text pairs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 721–731.

[68] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations (ICLR)*.

[69] Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Proceedings of Interspeech 2010*.

[70] Mikolov, T., Le, Q. V., and Sutskever, I. (2013b). Exploiting similarities among languages for machine translation. *arXiv preprint*.

[71] Miller, G. A. (1995). Wordnet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

[72] Miller, G. A., Chodorow, M., Landes, S., Leacock, C., and Thomas, R. G. (1994). Using a semantic concordance for sense identification. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

[73] Moro, A. and Navigli, R. (2015). Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297.

[74] Murata, M., Uchimoto, K., Ma, Q., and Isahara, H. (2000). Bunsetsu identification using category-exclusive rules. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING)*, pages 565–571.

[75] Nakazawa, T., Yaguchi, M., Uchimoto, K., Utiyama, M., Sumita, E., Kurohashi, S., and Isahara, H. (2016). ASPEC: Asian scientific paper excerpt corpus. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*, pages 2204–2208.

[76] Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69.

[77] Navigli, R., Jurgens, D., and Vannella, D. (2013). Semeval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231.

[78] Neubig, G. (2011). The Kyoto free translation task. http://www.phontron.com/kftt.

[79] Neubig, G. (2016). Lexicons and minimum risk training for neural machine translation: NAIST-CMU at WAT2016. In *Proceedings of the Third Workshop on Asian Translation (WAT)*, pages 119–125.

[80] Neubig, G., Morishita, M., and Nakamura, S. (2015). Neural reranking improves subjective quality of machine translation: NAIST at WAT2015. In *Proceedings of the Second Workshop on Asian Translation (WAT)*, pages 35–41.

[81] Neubig, G., Nakata, Y., and Mori, S. (2011). Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 529–533.

[82] Nguyen Le, A., Martinez, A., Yoshimoto, A., and Matsumoto, Y. (2017). Improving sequence to sequence neural machine translation by utilizing syntactic dependency information. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (IJCNLP)*.

[83] Ni, K. and Wang, W. Y. (2017). Learning to explain non-standard English words and phrases. In *Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 413–417.

[84] Nirenburg, S. and Somers, H. L. (2003). *Readings in machine translation*. MIT Press.

[85] Noraset, T., Liang, C., Birnbaum, L., and Downey, D. (2017). Definition modeling: Learning to define word embeddings in natural language. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*, pages 3259–3266.

[86] Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

[87] Pantel, P. and Lin, D. (2002). Discovering word senses from text. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, KDD '02, pages 613–619.

[88] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

[89] Pascanu, R., Gulcehre, C., Cho, K., and Bengio, Y. (2014). How to construct deep recurrent neural networks. In *Proceedings of the Second International Conference on Learning Representations (ICLR)*.

[90] Pradhan, S., Loper, E., Dligach, D., and Palmer, M. (2007). Semeval-2007 task-17: English lexical sample, srl and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92.

[91] Prochasson, E., Morin, E., and Kageura, K. (2009). Anchor points for bilingual lexicon extraction from small comparable corpora. In *Proceedings of the 12th Machine Translation Summit (MT SUMMIT XII)*, pages 284–291.

[92] Raganato, A., Camacho-Collados, J., and Navigli, R. (2017). Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (ACL)*, pages 99–110. Association for Computational Linguistics.

[93] Razmara, M., Siahbani, M., Haffari, R., and Sarkar, A. (2013). Graph propagation for paraphrasing out-of-vocabulary words in statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1105–1115.

[94] Sennrich, R., Haddow, B., and Birch, A. (2016). Edinburgh neural machine translation systems for WMT 16. In *Proceedings of the First Conference on Machine Translation (WMT)*, pages 371–376.

[95] Serban, I. V., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A., and Bengio, Y. (2017). A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*.

[96] Shalev-Shwartz, S., Singer, Y., Srebro, N., and Cotter, A. (2011). Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical programming*, 127(1):3–30.

[97] Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656.

[98] Shigeto, Y., Suzuki, I., Hara, K., Shimbo, M., and Matsumoto, Y. (2015). Ridge regression, hubness, and zero-shot learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 135–151. Springer.

[99] Siddharthan, A. (2014). A survey of research on text simplification. *International Journal of Applied Linguistics*, 165(2):259–298.

[100] Snyder, B. and Palmer, M. (2004). The english all-words task. In Mihalcea, R. and Edmonds, P., editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43.

[101] Stolcke, A. et al. (2002). SRILM-an extensible language modeling toolkit. In *Proceedings of the seventh International Conference on Spoken Language Processing (ICSLP)*, pages 901–904.

[102] Su, J., Tan, Z., Xiong, D., Ji, R., Shi, X., and Liu, Y. (2017). Lattice-based recurrent neural network encoders for neural machine translation. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*.

[103] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems (NIPS)*, pages 3104–3112.

[104] Tai, K. S., Socher, R., and Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 1556–1566.

[105] Tsunakawa, T., Okazaki, N., Liu, X., and Tsujii, J. (2009). A Chinese-Japanese lexical machine translation through a pivot language. *ACM Transactions on Asian Language Information Processing (TALIP)*, 8(2):9:1–9:21.

[106] Turney, P. D., Pantel, P., et al. (2010). From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research (JAIR)*, 37(1):141–188.

[107] Wang, X., Pham, H., Yin, P., and Neubig, G. (2018). A tree-based decoder for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4772–4777.

[108] Watanabe, T., Imamura, K., Kazawa, H., Graham, N., Nakazawa, T., and Okumura, M. (2014). *Machine Translation (In Japanese)*. CORONA PUBLISHING.

[109] Watanabe, T., Sumita, E., and Okuno, H. G. (2003). Chunk-based statistical translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 303–310.

[110] Wu, H., Wang, H., and Zong, C. (2008). Domain adaptation for statistical machine translation with domain dictionary and monolingual corpora. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING)*, pages 993–1000.

[111] Wu, S., Zhang, D., Yang, N., Li, M., and Zhou, M. (2017). Sequence-to-dependency neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 698–707.

[112] Wu, S., Zhang, D., Zhang, Z., Yang, N., Li, M., and Zhou, M. (2018). Dependency-to-dependency neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2132–2141.

[113] Xiao, M. and Guo, Y. (2014). Distributed word representation learning for cross-lingual dependency parsing. In *Proceedings of the 18th Computational Natural Language Learning (CoNLL)*, pages 119–129.

[114] Yamamoto, H. and Sumita, E. (2007). Bilingual cluster based models for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 514–523.

[115] Yoshinaga, N. and Kitsuregawa, M. (2009). Polynomial to linear: Efficient classification with conjunctive features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1542–1551.

[116] Yoshinaga, N. and Kitsuregawa, M. (2010). Kernel slicing: Scalable online training with conjunctive features. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pages 1245–1253.

[117] Yoshinaga, N. and Kitsuregawa, M. (2014). A self-adaptive classifier for efficient text-stream processing. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*, pages 1091–1102.

[118] Yu, H., Wang, J., Huang, Z., Yang, Y., and Xu, W. (2016). Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4584–4593.

[119] Zeiler, M. D. (2012). ADADELTA: An adaptive learning rate method. In *arXiv:1212.5701 [cs.LG]*.

[120] Zhou, H., Tu, Z., Huang, S., Liu, X., Li, H., and Chen, J. (2017). Chunk-based bi-scale decoder for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Publications

Publications related to the thesis

Journal papers

[1] 石渡祥之佑，鍜治伸裕，吉永直樹，豊田正史，喜連川優．“文脈語間の対訳関係を用いた単語の意味ベクトルの翻訳．”人工知能学会論文誌，人工知能学会，第31巻4号, 2016.

International conference

[2] Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Masashi Toyoda, and Masaru Kitsuregawa. “Learning to Describe Unknown Phrases with Local and Global Contexts.” In *Proc. NAACL-HLT*, ACL, 2019. (To appear)

[3] Shonosuke Ishiwatari, Jingtao Yao, Shujie Liu, Mu Li, Ming Zhou, Naoki Yoshinaga, Masaru Kitsuregawa, and Weijia Jia. “Chunk-based Decoder for Neural Machine Translation.” In *Proc. ACL*, ACL, 2017.

[4] Shonosuke Ishiwatari, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa. “Instant Translation Model Adaptation by Translating Unseen Words in Continuous Vector Space.” In *Proc. CICLing*, Springer, 2016.

[5] Shonosuke Ishiwatari, Nobuhiro Kaji, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa. “Accurate Cross-lingual Projection between Count-based Word Vectors by Exploiting Translatable Context Pairs.” In *Proc. CoNLL*, ACL, 2015.

Domestic conference

[6] 石渡祥之佑，林佑明，Graham Neubig，吉永直樹，豊田正史，喜連川優．“系列編集モデルに基づく単語ベクトルからの定義文生成”. 言語処理学会年次大会，2018.

[7] 石渡祥之佑，Jingtao Yao，Shujie Liu，Mu Li，Ming Zhou，吉永直樹，喜連川優，Weijia Jia． “ニューラル機械翻訳のための句に基づくデコーダ”. 言語処理若手の会，2017.

[8] 石渡祥之佑，吉永直樹，豊田正史，喜連川優． “未知語の分布表現の翻訳に基づく機械翻訳のドメイン適応”. 言語処理学会年次大会，2016.

[9] 石渡祥之佑，鍛治伸裕，吉永直樹，喜連川優． “文脈語間の対訳関係を用いた単語ベクトルの翻訳”. 言語処理若手の会，2015.

[10] 石渡祥之佑，鍛治伸裕，吉永直樹，豊田正史，喜連川優． “ウェブ上の言語資源を用いた単語のベクトル表現の翻訳”. WebDB Forum，2014.

Publications non-related to the thesis

International conference

[11] Shonosuke Ishiwatari*, Ryota Hinami*, Kazuhiko Yasuda, and Yusuke Matsui. “Mantra: Fully-Automatic Manga Translation.” Submitted to *SIGGRAPH*, Technical Papers, 2019. (* Joint First Authors)

[12] Masato Neishi*, Jin Sakuma*, Satoshi Tohda*, Shonosuke Ishiwatari, Naoki Yoshinaga, and Masashi Toyoda. “A Bag of Useful Tricks for Practical Neural Machine Translation: Embedding Layer Initialization and Large Batch Size.” In *Proc. WAT*, AFNLP, 2017. (* Joint First Authors)

[13] Shoetsu Sato, Shonosuke Ishiwatari, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa. “UT Dialogue System at NTCIR-12 STC.” In *Proc. NTCIR-12 STC*, NTCIR, 2016.

Awards

[14] 石渡祥之佑. Innovative Technologies 2018. デジタルコンテンツ協会，2018.

[15] 石渡祥之佑. Best Presentation Award. 生産技術研究所 PhD Student Live，2017.

[16] 石渡祥之佑. 人工知能学会創設30周年記念特集論文 優秀賞. 人工知能学会，2016.

[17] Shonosuke Ishiwatari. Best Poster Award. CICLing 2016.

[18] 石渡祥之佑. 専攻長賞. 東京大学情報理工学系研究科 電子情報学専攻，2016.

[19] 石渡祥之佑. 奨励賞. YANS 2015.

[20] 石渡祥之佑. 学生奨励賞. WebDB Forum 2014.

[21] 石渡祥之佑. 企業賞（Yahoo! JAPAN）. WebDB Forum 2014.

[22] 石渡祥之佑. 優秀発表賞. 東京大学 音声・言語・コミュニケーション研究会学生交流会，2013.