THE UNIVERSITY OF TOKYO

MASTER THESIS

Data Editing and Generation for Domain Adaptation in Neural Machine Translation

（ニューラル機械翻訳における分野適応用データの編集及び生成）

Author:
Yitong WENG

Supervisor:
Dr. Koiti HASIDA

A thesis submitted in fulfillment of the requirements
for the degree of Master of Information Science and Technology

in the

Hasida Laboratory
Graduation School of Information Science and Technology

July 30, 2018

Declaration of Authorship

I, Yitong WENG, declare that this thesis titled, "Data Editing and Generation for Domain Adaptation in Neural Machine Translation" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 翁 亦 童

Date: 2018. 7. 30

THE UNIVERSITY OF TOKYO

Abstract

Graduation School of Information Science and Technology

Master of Information Science and Technology

Data Editing and Generation for Domain Adaptation in Neural Machine Translation

by Yitong WENG

Methods of data editing and generation are shown to improve the effect of fine-tuning, working as a main way of domain adaptation in Neural Machine Translation. The proposed methods include partial repetition and reordering to edit the data, and data generation using bilingual dictionary and synonyms given by Word2Vec models.

Acknowledgements

I take this opportunity to express gratitude to all of the faculty members and students of Hasida laboratory, who have given me a lot of advice and help during the 2 years of my master course. Especially, I would like to thank Prof. Hasida, who has made time every week for laboratory meeting, discussing my research in details and giving instructions for thesis writing precisely. I also thank my parents for both substantial and spiritual support, which has ensured that I could concentrate on my research here as a international student.

Contents

List of Figures

List of Tables

List of Abbreviations

NMT	**N**eural **M**achine **T**ranslation
GOC	**G**eneration **O**rigin **C**orpus
IDD	**I**n-**D**omain **D**ictionary

Chapter 1

Introduction

Machine Translation (MT), is to translate a natural language to another using programs without manual work. Earliest systems about MT can be traced from 1950s, and had been developed a lot in 1980s (Hutchins, 1994). In recent years, with the quick growing of computer industry, MT has seen rapid developments.

Statistical Machine Translation (SMT), had been the main approach of MT for a long time. Unlike traditional approaches like Rule-based Machine Translation(RBMT), the construction of a SMT system does not require linguistic specialists, instead, a large bilingual dataset containing translation pairs between the source language and target language (usually called corpus) is important to the translation ability of the system.

However, in about 2014, researches about Neural Machine Translation (NMT) came in to sight. Based on recent researches of neural networks using latest achievements in biology field, NMT quickly gathered attentions from researchers. NMT systems are end-to-end, with simpler construction than SMT systems.

Vital achievements of NMT researches including the Encoder-Decoder model (Cho et al., 2014), Long Short-Term Memory (LSTM) model (Sutskever, Vinyals, and Le, 2014) attention-based model (Bahdanau, Cho, and Bengio, 2014) and local attention-based model (Luong, Pham, and Manning, 2015). In 2016, Google released its online translation service using NMT(Wu et al., 2016), which showed appreciable translation ability compared to its conventional Phrase-based Machine Translation service, drawing attention extensively even in public.

However, the performance of NMT models depends greatly on the domain of data. When applied to topics or domains that are not included in the training data, the NMT system tends to perform poorly. It is not easy to obtain a bilingual corpus in a given specialized domain with a sufficiently large

size. Thus, to achieve good translation ability in specialized domain, a process called domain adaptation is usually conducted, which utilizes a smaller dataset in the target domain and a model trained in the source domain in advance. A conventional method of NMT domain adaptation named fine-tuning, which uses a relatively small corpus in the target domain to conduct retraining on an existing NMT model, has been proved to be effective in this scenario (Luong and Manning, 2015).

The effect of fine-tuning is greatly related to the size and quality of the dataset for retraining. When given limited data in the target domain, fine-tuning cannot always bring satisfying results. This study aims to explore methods of editing and generating data in the specialized domain for fine-tuning, in order to improve the performance of adapted NMT models.

This thesis is structured as follows. In Chapter 2, we list the related work to our study. Chapter 3 proposes methods of data editing. Chapter 4 proposes methods of data generation using bilingual dictionary. Chapter 5 gives discussion about the proposed methods and Chapter 6 gives a conclusion.

Chapter 2

Related Work

Domain adaptation is a concept in the field of machine learning, which refers to the process to make a model well trained in one domain available in another domain. For example, when a great amount of labeled data in domain A are available but no or less labeled data in domain B are available, it is easy to train a model using the labeled data to give predictions in domain A, but not the same with domain B. The research called domain adaptation aims to adapt the model to make it able to give good predictions in domain B. The domain A here is called the source domain, and domain B is called the target domain.

The main approach of domain adaptation in NMT is fine-tuning, which conducts retraining on a well trained model in the source domain using a relatively small dataset in the target domain.

An early try of adapting written language to spoken language of English-German language pair was conducted (Luong and Manning, 2015). A dataset containing 4.5 million sentence pairs in English-German was first used to train a NMT model. Fine-tuning was then conducted by retraining the model with a dataset in the spoken language domain containing 200 thousand sentence pairs. As a result, the model fine-tuned gave a +5.2 BLEU score over the original one. The experiment showed the effect of fine-tuning. However, the dataset used for fine-tuning had a considerable size, which is not easily to reach in some situations.

Furthermore, a method named mixed fine-tuning was proposed to solve the issue of over-fitting (Chu, Dabre, and Kurohashi, 2017), which refers to the phenomenon that a fine-tuned model becomes badly-performing in domains other than the target domain. In this study, experiments showed that creating fine-tuning data as a mixture of data in the target domain and source domain performed well on both domains.

Facing the situation that specialized domain lacks organized bilingual

corpus, data generation for NMT training or fine-tuning is considered a meaningful research topic. As is easier to find domain-specific dictionaries than preparing bilingual corpora in the target domain, using dictionaries to generate corpora is considerable. The use of bilingual dictionaries to generate a pseudo bilingual corpus is discussed (Zhang and Zong, 2016). The generation method proposed in this study used statistical machine translation (SMT) to create the outputs in the target language, which is considered to cost a lot. Besides, there is not always a SMT model which performs well enough in the target domain.

Chapter 3

Tuning Data Editing

The size of training data is believed to be one of the most influential factors in the process of NMT. Notice that we are often talking about the size of a *real* dataset, informing it is a simple random sample of a corpus. In the situation of fine-tuning towards a specific domain, it is difficult to obtain new data to enrich the NMT model with the vocabulary and expressions in the target domain, which means that the *real* size of data is limited.

However, we cannot assert that the original form and structure of a dataset always lead to the best training result. In this chapter, we tried to explore if specific types of editing on the tuning data can help improve the effect of fine-tuning.

3.1 Simple Repetition

A preliminary try was first made to identify whether a simple repetition works.

3.1.1 Method

Two types of simple repetition were considered.

- *by-sentence*:
  Repeat each sentence data just after itself for both sides of bilingual data.
- *by-data*:
  Repeat all the sentences at the end of the file for both sides of bilingual data.

TABLE 3.1: The settings of Amazon EC2 instance used in this paper.

Settings	Value
Instance type	p2.xlarge
AMI ID	Deep Learning AMI (Ubuntu) Version 6.0 (ami-d2c759aa)
Virtual CPU number	4
Memory (GiB)	61
Volume size (GiB)	100

3.1.2 Experiments

Environment

We deployed our experiment environment on *Amazon Elastic Compute Cloud (Amazon EC2)*. Amazon EC2 is a web service that provides compute capacity in the cloud. By choosing an instance with GPU support, we were able to conduct GPU calculation, which greatly increased the speed of training and translation. The detailed information of the instance used in this paper is listed in Table 3.1.

We used OpenNMT library (Klein et al., 2017) to conduct our experiments.

OpenNMT is an open source (MIT) initiative for NMT and neural sequence modeling. Besides the basic function of neural model training and translation, it provides us with a retraining feature, which allows training a model on incremental data. We use this function to conduct fine-tuning on a trained base model.

It has 3 main implementations including OpenNMT-lua (a.k.a. OpenNMT), OpenNMT-py, and OpenNMT-tf. In this paper, we used OpenNMT-lua library to conduct all the experiments.

Data Preparation

We used *NTCIR Patent Machine Translation (NTCIR PatentMT)* (English - Japanese, 3.0M sentence pairs) as the training data for the base model. It is composed with patents in different fields and classified by the International Classification of Patents for Inventions, shown in Table 3.2.

The whole corpus of PatentMT was used as the source domain (hereinafter called "out-of-domain") data in this experiment.

TABLE 3.2: The section titles of International Classification of Patents for Inventions

A. Human Necessities
B. Performing Operations; Transporting
C. Chemistry; Metallurgy
D. Textiles; Paper
E. Fixed Constructions
F. Mechanical Engineering; Lighting; Heating; Weapons; Blasting
G. Physics
H. Electricity

TABLE 3.3: The name definitions of models and descriptions about the data they used for fine-tuning in the experiment of simple repetition.

Name	**Description**
0. original	100 sentence pairs randomly selected from Kusuri-no-Shiori.
1. by-sentence	Each sentence in data 0 is repeated just after itself.
2. by-data	All sentences in data 0 are repeated at the end of the file.

For target domain (hereinafter called "in-domain") data, we used *Kusuri-no-Shiori*, a drug information database in which each document describes the effects and precautions of a drug. Since it contains a great number of words and expressions that do not exist in the PatentMT, we consider that it is suitable as an in-domain dataset.

We randomly selected 100 translation pairs from Kusuri-no-Shiori as the in-domain data for fine-tuning.

By the 2 types of simple repetition shown in Figure 3.1, we obtained 2 more datasets. Each dataset was used in the experiment to train a NMT model. The name definitions of models and descriptions about the data they used for fine-tuning are listed in Table 3.3.

For test data, we randomly selected 50 translation pairs from Kusuri-no-Shiori.

Furthermore, before training, we conducted some pre-processes for all the data we would use in the experiments. We used *KyTea* to do word segmentation for all the Japanese data. For English data, we transferred all letters to lower-case and isolated punctuations, which made them to behave as words.

We are not going to repeat these pre-processes in the following sections and chapters in this paper, but all the data are processed in this way before

TABLE 3.4: Important parameters used in base model training in the experiment of simple repetition. Parameters listed are defined in the documentation of OpenNMT library.

Parameter	Value
model_type	seq2seq
layers	2
encoder_type	rnn
rnn_size	500
rnn_type	LSTM
attention	global
global_attention	general
learning_rate	1
end_epoch	13

TABLE 3.5: Important parameters used in fine-tuning in the experiment of simple repetition. Parameters listed are defined in the documentation of OpenNMT library.

Parameter	Value
learning_rate	1
end_epoch	10
shuffle	false
sort	false
update_vocab	none

used used for training and fine-tuning.

Training of Base Model

We conducted a new session of training using the out-of-domain data. Table 3.4 lists important parameters used for the training. After training for 13 epochs, we obtained an English-Japanese model with a perplexity of 3.17. This is used as the out-of-domain base model in the following experiments.

Fine-tuning

We conducted re-training on the base model using 3 pre-processed datasets. Parameters used for fine-tuning are listed in Table 3.5. Some parameters are omitted because they cannot be changed when continuing a training on a NMT model, so that they are the same as listed in Table 3.4.

TABLE 3.6: BLEU scores for models tuned by in-domain data with simple repetition.The models are named as described in Table 3.3

Model	Score
0. original	28.45
1. by-sentence	27.58
2. by-data	28.04

Results

We conducted translation to the test data using the models obtained above. BLEU (Papineni et al., 2002) was used to calculate a score which shows the similarity between translation results and reference file.

The BLEU scores of translation results for the models are listed in Table 3.6. The models are named as described in Table 3.3.

It was shown from the results that simple repetition gave an unsatisfying performing. The model 'by-sentence' performed the worst, showing that feeding the same sentences for several times together does not improve training.

3.2 Partial Repetition

Besides simple repetition, a method of partial repetition was considered. Since the performance of NMT basically decreases as the length of sentences increases, we hypothesize that decomposing sentences into clauses, words and noun phrases in the tuning data will help improve the score of in-domain translation. By adding these decomposition products into the tuning data, we can also expect an increase of its size, although it is not the *real* size.

Method

Manual decomposing was conducted in this work. Since we used Japanese-English as the language pair in this work, decomposing sentences into clauses by punctuations seemed difficult due to the great difference between the word order and sentence composition of the two languages. Thus, we basically chose some of the field-specific words and noun phrases that can be easily recognized in both English and Japanese data.

whole-part, mixed	…… また、この薬を飲んでいることをご家族やまわりの方にもお知らせください。 この薬を飲んでいること 低血糖症状が起こった時には十分量の糖分（砂糖、ブドウ糖、清涼飲料水など）をとるようにしてください。 低血糖症状 十分量の糖分 砂糖、ブドウ糖、清涼飲料水 ……
part-whole, mixed	…… この薬を飲んでいること また、この薬を飲んでいることをご家族やまわりの方にもお知らせください。 低血糖症状 十分量の糖分 砂糖、ブドウ糖、清涼飲料水 低血糖症状が起こった時には十分量の糖分（砂糖、ブドウ糖、清涼飲料水など）をとるようにしてください。 ……
whole-part, separate	…… また、この薬を飲んでいることをご家族やまわりの方にもお知らせください。 低血糖症状が起こった時には十分量の糖分（砂糖、ブドウ糖、清涼飲料水など）をとるようにしてください。 …… …… この薬を飲んでいること 低血糖症状 十分量の糖分 砂糖、ブドウ糖、清涼飲料水 ……
part-whole, separate	…… この薬を飲んでいること 低血糖症状 十分量の糖分 砂糖、ブドウ糖、清涼飲料水 …… …… また、この薬を飲んでいることをご家族やまわりの方にもお知らせください。 低血糖症状が起こった時には十分量の糖分（砂糖、ブドウ糖、清涼飲料水など）をとるようにしてください。 ……

FIGURE 3.1: Examples in Japanese data for 4 types of alignments concerning adding the decomposed sentences into the original tuning data. Sentences appearing in the original data are underlined.

Concerning adding the decomposed sentences (hereinafter called the 'repeating parts') into the tuning data, we considered 4 types of alignments.

- *whole-part, mixed*:
  All repeating parts of one original sentence are added right after it.
- *part-whole, mixed*:
  All repeating parts of one original sentence are added right before it.
- *whole-part, separated*:
  All repeating parts are added after all the original data.
- *part-whole, separated*:
  All repeating parts are added before all the original data.

Examples in Japanese data for 4 types of alignments are shown in Figure 3.1. Corresponding processes are taken on the English side.

The experiments were conducted with all the 4 ways mentioned above.

TABLE 3.7: Important parameters used in fine-tuning in the experiment of partial repetition. Parameters listed are defined in the documentation of OpenNMT library.

Parameter	Value
learning_rate	1
end_epoch	10
shuffle	false
sort	false
update_vocab	none

3.2.1 Experiments

Data Preparation

We conducted partial repetition on the same in-domain data which was used in the experiments of simple repetition. By decomposing the sentences into clauses, words and noun phrases, we expanded the data from 100 to 250 translation pairs in the 4 ways of alignment mentioned above. For test data, we have randomly selected 250 translation pairs from Kusuri-no-Shiori and divided them into 5 sets (50 translation pairs per set).

Fine-tuning

We conducted re-training on the base model trained in the experiments of simple repetition using 4 in-domain datasets with partial repetition and the original in-domain dataset without repetition. Parameters used for fine-tuning are listed in Table 3.7.

Results

We conducted translation to 5 test datasets and original tuning data (whole sentences only, 100 translation pairs) using the models obtained above. The BLEU scores of translation results for the models are listed in Table 3.8.

As shown in Table 3.8, for the tuning data, model 3 marked the best score of 69.0 BLEU points with a substantial improvement by +10.2 BLEU points, and all the other models also improved the scores by over +6 BLEU points. For the test data, the adapted models performed better than baseline in 4 datasets out of 5, and the combined test dataset showed improvements of +0.8 ~ +1.68 BLEU points. This result shows that partial repetition of original

TABLE 3.8: BLEU scores for models tuned by in-domain data with partial repetition in 4 types of alignments. 'Tuning' column contains the scores of translation on the original tuning data. 'Test All' column is the average of the 'Test1' ~ 'Test5' column.

Models	Tuning	Test1	Test2	Test3	Test4	Test5	Test All
0. whole	58.8	37.7	**32.1**	23.2	25.1	21.2	27.86
1. whole-part, mixed	67.9	40.0	31.5	27.3	**26.9**	22.0	**29.54**
2. part-whole, mixed	65.9	**42.3**	30.9	25.6	24.3	20.8	28.78
3. whole-part, separated	**69.0**	**42.4**	28.2	**27.9**	24.1	**23.3**	**29.18**
4. part-whole, separated	65.4	40.6	30.0	27.1	25.2	20.4	28.66

tuning data is effective in improving the adaptation when the size of tuning data is limited.

3.2.2 Discussion

The difference among the scores of the 5 test datasets is probably due to the different similarity between them and the tuning data. As we are concerned with the effect of domain adaptation, we consider that the scores of tuning data set themselves count the most, and test datasets which are translated relatively well count more than others. Thus, Model 3 is considered the best, followed by Model 1. As Model 1 ~ 4 actually contain the same data with different orders among translation pairs, this order may have a large influence on the learning effect. Hence, we investigate a second technique as reordering of the tuning data before the fine-tuning process.

3.3 Reordering

As mentioned above, Model 3 *(whole-part, separated)* produced the best result, followed by Model 1 *(whole-part, mixed)*. The two models here have a common order in which the original translation pairs are used before the partial repeated translation pairs. Therefore, we may well suppose that to sort the training data in a descending order of the text length may bring an improvement to fine-tuning effect. To check this hypothesis, we conducted the following experiments.

Method

We conducted sorting both in ascending order and descending order for the tuning data. We chose the source language side as the representative of the

bilingual data. The following pseudo codes show how we conducted the process.

Pseudo Codes 1
Parallel sorting on the bilingual data by the length of source sentences.

Input: *sourceData*: List, *targetData*: List, *isDescending*: Boolean
Output: *sourceDataSorted*: List, *targetDataSorted*: List

```
size ← sourceData.GetSize()
data ← new List
for i = 0 → size − 1 do
    data[i] ← new Array{sourceData[i], targetData[i]}
end for
data.SortBy(element[0].CountBlank()), isDescending)
for i = 0 → size − 1 do
    sourceDataSorted[i] ← data[i][0]
    targetDataSorted[i] ← data[i][1]
end for
return sourceDataSorted, targetDataSorted
```

3.3.1 Experiments

Data Preparation

We sorted the data prepared for the experiment of partial repetition in both descending and ascending order by the length of the Japanese side of the translation pairs. Furthermore, we did the same process to the original tuning data.

For testing, we used the same data as in the experiment of partial repetition.

Fine-tuning

We trained 4 NMT models using the *whole-part* data (sorted in descending/ascending orders) and the *whole* data (sorted in descending/ascending orders) on the same out-of-domain base model for 10 epochs each. The other parameters were set the same as Table 3.7.

Results

Tests with the same test datasets were conducted on the newly trained models. The results are shown in Table 2. For comparison, a part of the results

TABLE 3.9: BLEU scores for models tuned by in-domain data with partial repetition and reordering. 'Tuning' column contains the scores of translation on the original tuning data. 'Test All' column is the average of the 'Test1' ~ 'Test5' columns.

Models	Tuning	Test1	Test2	Test3	Test4	Test5	Test All
0. whole	58.8	37.7	**32.1**	23.2	25.1	21.2	27.86
1. whole-part, mixed	67.9	40.0	31.5	27.3	**26.9**	22.0	**29.54**
2. part-whole, mixed	65.9	**42.3**	30.9	25.6	24.3	20.8	28.78
3. whole-part, separated	**69.0**	**42.4**	28.2	**27.9**	24.1	**23.3**	**29.18**
4. part-whole, separated	65.4	40.6	30.0	27.1	25.2	20.4	28.66
5. whole-part, descending	65.4	38.0	34.3	26.0	26.9	22.1	29.46
6. whole-part, ascending	**69.0**	**42.8**	35.5	**29.6**	**27.9**	23.2	**31.80**
7. whole, descending	65.4	38.1	**36.5**	27.9	**28.3**	21.8	30.52
8. whole, ascending	65.4	40	34.8	27.6	26.6	**23.9**	30.58

(Model 0 and Model 3) from the experiment of partial repetition are also included in Table 2.

For Model 7 and Model 8, we can see an improvement of +6.6 BLEU points for tuning data, and over +2.5 BLEU points for the combined test data, which shows that no matter whether the data are sorted in descending or ascending order, reordering results in improvements. The two orders are not very different in the results, as scores of combined test data are quite similar.

For Model 5 and Model 6, comparing with the best performing Model 3 tuned by repeated data in the former experiment, we can recognize improvements up to +2.62 BLEU points for the test data. However, the scores for tuning data itself do not always go over the one with relatively random order. In details, Model 6, which is tuned by data in ascending order, performs better than Model 3 in all the test datasets, but Model 5 does not always do. Also, Model 6 performs the best among all the models in 3 test datasets out of 5, and overall, gets 31.80 BLEU points on the combined test data, which turns out to be the highest score. According to this result, Model 6 was considered to be the best model among the experiments conducted in this chapter.

3.3.2 Discussion

In the experiment of partial repetition, we added repeated parts of sentences to the data before fine-tuning. Basically, this process proved to work well no matter where the partial data were inserted. However, though not strongly, inserting the partial data after the whole sentences worked better. As NMT systems tend to remember recently provided data better, we assumed that learning shorter instances later helps the system remember words and phrases

in the relevant field, where these words or phrases are unique and typical, or rarely appear in other domains, resulting in a better adaptation effect.

We further supposed from this result that if we sort the data in the descending order by the length of sentences, we may obtain a better result as most of short, domain-typical words and phrases are learned later. However, the results did not seem to support this hypothesis. The ascending order showed a better result when the reordering process is conducted to the partially repeated data, which contains more short texts than the original one. Considering the human learning process, it is natural to learn words and phrases first, followed by longer sentences containing them. By sorting the data by sentence length in ascending order, the NMT system may better 'understand' (precisely encode) the long sentences as it has already learned from their constituents. This explanation is contradictory with the one we have given to the last experiment, however, the difference between scores in that experiment is rather subtle, and the supposed tendency does not show strong reproducibility.

Furthermore, it is not ignorable that the reordering process also seemed effective for original data, which seldom involve instance sets with whole-part relationships. In this situation, the reordering process does not induce learning from part to whole, as is mentioned above. Instead, it simply places the same or similar sentences together. However, in the following experiments, it was proved that this conclusion is not reproducible.

Chapter 4

Tuning Data Generation

In Chapter 3, we proposed methods to edit in-domain data in order to improve fine-tuning effect. However, these methods cannot increase the *real* size of the dataset, which reflects the richness of language. Thus, in this Chapter, we considered methods to generate data for fine-tuning using in-domain bilingual dictionary, which is easier to obtain than in-domain bilingual corpus.

4.1 Experiment: Synonym Replacement

Assume that we have obtained a bilingual dictionary that contains a great number of words in the target domain. As low performance of NMT in specialized domains is usually due to domain-specific words unknown to the NMT model, domain-adaptation task is mainly about the domain-specific words. Thus, we can expect that an in-domain dictionary (hereinafter IDD) works partly like a bilingual corpus. However, it cannot be used as tuning data without augmentation, otherwise the length of the output of NMT model will be limited in only several words.

To augment the dictionary, it is simple to think about importing the words in the dictionary to sentences from other data. Thus, an extra dataset is needed for this method. we will refer to it as a generation origin corpus (hereinafter GOC). A GOC can be out-of-domain or in-domain. If it is out-of-domain, this method generates data without any in-domain data, which means a very low adaptation cost. If it is in-domain, we can still expect the generation improve the fine-tuning effect. Experiments were conducted using both in-domain data and out-of-domain data as the GOC.

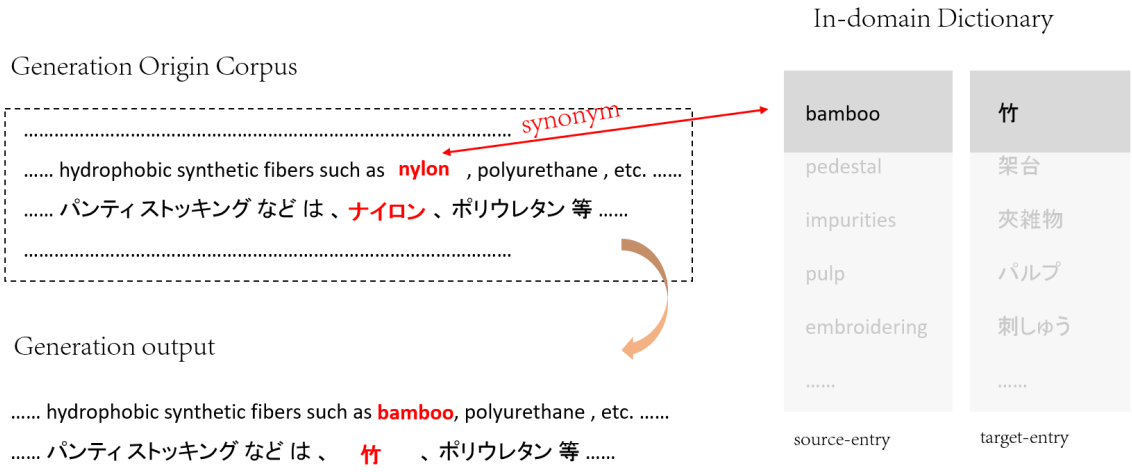

FIGURE 4.1: An example of synonym replacement in English-Japanese translation.

4.1.1 Similarity Evaluation

As for importing the entries of in-domain dictionary to the sentences from GOC, we considered that replacing them with synonyms may work well. If the replacement takes place between words with the same part of speech and similar meaning, NMT model may be able to learn the domain-specific words from the generated sentences without decreasing its translation ability for other sentences.

To evaluate the similarity of two words, we have adopted Word2VecMikolov et al., 2013 to create words embeddings. The models of Word2Vec evaluates the similarity of two words by calculating the distance of two vectors representing the words surrounding them in the given corpora.

Given the source part of an entry in the IDD, a list of words that is similar to it using Word2Vec can be obtained. By setting a threshold of similarity or giving a maximum number of the candidates, the candidates for the words to be replaced by the source part of the entry can be determined. In the following part of this Chapter, we will refer to the source part of an entry of the IDD as an *source-entry*, and the candidate determined in this step as a *source-candidate*. Similarly, *target-entry* and *target-candidate* are referred to as the ones in the target part. Figure 4.1 shows an example of replacement among the four words mentioned in English-Japanese translation.

4.1.2 Pairing and Generation

In this step, we searched the source part of the GOC for each source-candidate. When a sentence containing a source-candidate was found, we next checked if its translation (the target sentence in the same translation pair) contained a target-candidate that was given according to the source-candidate by a common bilingual dictionary or a translation tool. If so, a new sentence pair was generated by replacing source-candidate with source-entry, and target-candidate target source-entry. The number of generated sentence pairs was counted, so that the process stops when the count reaches the upper limit given as a parameter.

The overall process is described in the following pseudo codes.

Pseudo Codes 2 Pairing and replacement between entries of the IDD and their synonyms in GOC.

Input: *IDD*: List, *GOC*: List, *maxCountPerEntry*: Integer
Output: *dataGenerated*: List

```
1:  dataGenerated ← new List
2:  for entryPair ∈ IDD do
3:      sourceCandidateList ← Word2Vec.GetSynonyms(entryPair.source)
4:      for sourceCandidate ∈ sourceCandidateList do
5:          count ← 0
6:          sourceCandidate.target ← Translate(sourceCandidate)
7:          for sentencePair ∈ GOC do
8:              if sentencePair.source contains sourceCandidate and
          sentencePair.target contains sourceCandidate.target  then
9:                  sentencePairGen.source ←
          sentencePair.source.Replace(sourceCandidate, entryPair.source)
10:                 sentencePairGen.target ←
          sentencePair.target.Replace(sourceCandidate.target, entryPair.target)
11:                 dataGenerated adds sentencePairGen
12:                 count ← count + 1
13:             end if
14:             if count == maxCountPerEntry then
15:                 break
16:             end if
17:         end for
18:     end for
19: end for
20: return dataGenerated
```

TABLE 4.1: The settings the environment for generation experiment in this Chapter.

Settings	Value
Python Version	2.7.12
genism Version	3.0.1
Google Cloud Translate Version	1.3.0

4.1.3 Experiment

We conducted experiments of data generation as described above, and fine-tuning using the generated data on an out-of-domain NMT model as well as translation tests.

Environment

We chose genismRehurek and Sojka, 2010, a Python library for topic modeling, as an implementation of Word2Vec.

The detailed information of the environment for generation experiment in this Chapter is listed in Table 4.1.

Data Preparation

Same with Chapter 3, we used NTCIR Patent MT as bilingual data to conduct experiments. According to the International Classification codes of the patents (shown in Table 3.2), we divided the data into several parts, in which the patents are classified into specialized domains. We chose the part **PHYSICS** as the out-of-domain corpus (1.03M sentence pairs), because it has the largest size. We then trained the base model using the out-of-domain model using the same settings in Table 3.4 in the previous experiments as conducted translation tests using it on the test data in all the other parts of the corpus. According to the result shown in Table 4.2, we chose the part **TEXTILES; PAPER** as the in-domain data (6898 sentence pairs) because it has a rather small size and low score, which indicated that the difference between it and the out-of-domain was apparent.

From the in-domain data, we have randomly selected 100 sentence pairs as test data and the rest as tuning data to avoid duplication.

For the in-domain dictionary, a translation test was first conducted on the in-domain test data using base model to highlight words that are expected to be domain-specific as they were basically not successfully translated and

TABLE 4.2: The result of translation tests and size of all the parts of NTCIR Patent MT using model trained on part G.

Field	BLEU	Training Size / MiB
A. Human Necessities	31.19	11.3
B. Performing Operations; Transporting	30.44	61.9
C. Chemistry; Metallurgy	29.92	8.5
D. Textiles; Paper	29.68	1.5
E. Fixed Constructions	28.66	1.5
F. Mechanical Engineering; Lighting; Heating; Weapons; Blasting	30.29	30.8
G. Physics	37.07	234.2
H. Electricity	34.45	139.6

replaced with a <unk> token. We then picked out a list of 38 word pairs as the in-domain dictionary (English - Japanese) for the following experiments.

Data Generation

The genism model was trained on the default English corpus. For out-of-domain translation, we used *Google Cloud Translation API*.

Based on the method described above, we conducted the experiment of data generation. Experimental settings are listed in Table 4.3. Both out-of-domain data and in-domain data were used as the GOC in the experiments for comparison.

Table 4.4 shows the basic information of the generation result. According to the table, 8 words in 38 are not included in the vocabulary of trained Word2Vec model, which certainly means they were not successfully put into the generated data. Besides, generations with some words failed because no candidates of them were found in the GOC, or their translations were not found in the corresponding sentences.

Training of Base Model

We conducted a new session of training using the out-of-domain data. Table 4.5 lists important parameters used for the training.

After training, we have obtained an English-Japanese model on the **PHYSICS** domain with a perplexity of 2.41. This is used as the out-of-domain base model in the following experiments.

TABLE 4.3: Experimental settings for data generation by the program described above based on Word2Vec.

Word2Vec Settings	Value
Training data size	97,657 KiB
Minimum count of appearance to be valid	5
Window	10
Other Settings	Value
Number of candidates per entry	20
Maximum number of sentences per candidate	10

TABLE 4.4: Summary of generation result using an out-of-domain corpus and an in-domain corpus as the generation original corpus(GOC).

	From out-of-domain	**From in-domain**
Total of entries	38	38
Out-of-vocabulary (OOV)	8	8
Failure (except OOV)	4	6
Total of generated sentences	2315	788

TABLE 4.5: Important parameters used in base model training in the experiment of tuning data generation. Parameters listed are defined in the documentation of OpenNMT library.

Parameter	**Value**
model_type	seq2seq
layers	2
encoder_type	rnn
rnn_size	500
rnn_type	LSTM
attention	global
global_attention	general
learning_rate	1
end_epoch	13

TABLE 4.6: Important parameters used in fine-tuning in the experiment of tuning data generation. Parameters listed are defined in the documentation of OpenNMT library.

Parameter	Value
learning_rate	1
end_epoch	10
shuffle	true
sort	false
update_vocab	merge

Fine-tuning

We then used the in-domain data and generated data to conduct fine-tuning on the base NMT model. Besides, mix data of in-domain data and generated data, as well as a generated data without replacement were also used for analysis.

Table 4.6 lists important parameters used for the training. Different from the experiments in Chapter 3, 'shuffle' were set to 'true' (default) and 'update_vocab' was set to 'merge' to improve learning of unknown words.

Results

Translation tests of the in-domain test data were then conducted with all the models trained before.

Data composition of the models and their BLEU scores, as the results of translation tests, are listed in Table 4.7. Model 'base' is the base model. Model 'orig' is tuned by the data generated from the in-domain data. Model 'out' is tuned by the data generated from the out-of-domain data. Model 'in-noreplace' is tuned by the data generated by a same process with Model 'in', just without the step of replacement. Model 'orig' is tuned by the whole in-domain data (conventional method of fine-tuning). Model 'in+orig' is tuned by a mix of in-domain data and the data used for Model 'orig'. Model 'out+orig' is tuned by a mix of in-domain data and the data used for Model 'out'.

In Table 4.7, the last column shows the scores of models on a part of the test data. In the test(dict) data, we have removed all sentences that do not contain any of the entries in the IDD. In this way, the score on this test data emphasized the effect of fine-tuning process, where we aimed to utilize the IDD and let the model learn the entries.

TABLE 4.7: Data composition of the tuned models and their BLEU Scores on in-domain test data and a part of the test data which all sentences include at least 1 entry in the IDD. Model 'base' is the base model. Model 'orig' is tuned by the data generated from the in-domain data. Model 'out' is tuned by the data generated from the out-of-domain data. Model 'in-noreplace' is tuned by the data generated by a same process with Model 'in', just without the step of replacement. Model 'orig' is tuned by the whole in-domain data (conventional method of fine-tuning). Model 'in+orig' is tuned by a mix of in-domain data and the data used for Model 'orig'. Model 'out+orig' is tuned by a mix of in-domain data and the data used for Model 'out'.

Models	data generated from	in-domain data	size	test score	test(dict) score
base			0kB	31.10	27.73
in	in-domain		154kB	25.66	24.14
out	out-of-domain		475kB	26.17	24.46
in-noreplace		part	154kB	24.60	22.31
orig		all	1,229kB	39.54	37.41
in+orig	in-domain	all	1,355kB	39.93	38.66
out+orig	out-of-domain	all	1,676kB	38.57	37.09

For model 'orig' and 'out', all scores turned to be lower than the base model. This was probably due to the small size of generated data compared to the in-domain test data. However, by comparing model 'orig' and 'in-noreplace', we can find that the process of replacement brought an improvement of +1.83 BLEU points on the test(dict) data as the two sets of tuning data were nearly the same except for the replacement of words. Moreover, we found that model 'orig' and 'out' succeeded in translating sentences with the words learned from the IDD through the data generation and fine-tuning process.

Figure 4.2 shows an example of successful fine-tuning, in which model 'in' is able to give the word '𠮟', while the base model gives an 'unknown' token.

By mixing generated data and in-domain data, we trained Model 'in+orig' and 'out+orig'. Model 'in+orig' shows a highest score of 38.66 BLEU points on the test(dict) data, which goes over model 'orig' by +1.25 BLEU points. From the result, we can recognize that adding the generated data into the in-domain data before fine-tuning process can result in an improvement of the fine-tuning effect. The proposed method can serve as an approach to expand the fine-tuning data which is not large enough.

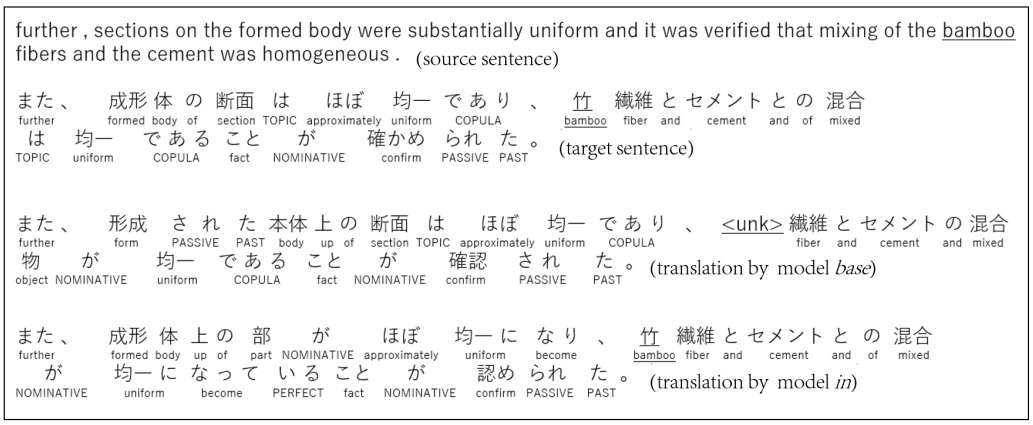

FIGURE 4.2: An example of successful fine-tuning by generated data. The unknown word 'bamboo' is recognized and successfully translated by the model 'orig' in Table 4.7.

4.1.4 Discussion

The proposed method succeeded in the situation where a set of in-domain data is available but needs expansion. However, it did not brought improvements in other situations, such as when we cannot obtain any in-domain bilingual data.

The reason why models 'in' and 'out' performed worse than the base model was probably the size of the tuning data. Model 'in-noreplace' worked as a control experiment to examine this assumption. The tuning data of Model 'in-noreplace' was actually only a selection of the in-domain data, which brought a large improvement of +8.44 or +9.58 BLEU points after fine-tuning (Model 'orig'). The difference between the two sets of tuning data was basically the size (788 compared to 6898). Thus, we suppose that increasing the size of generated data will improve the scores obviously.

We should suppose that the in-domain dictionary contains a large number of domain-specific words because actually such dictionaries are not difficult to obtain. However, the dictionary we have used in the experiments was produced non-automatically and thus is not complete with a rather small size. A dictionary that is well produced and complete should bring better performance.

Besides the size of the IDD, the percentage of failure in the generation process was also not negligible. There are 3 types of failure: (1) Out-of-Vocabulary; (2) Search failure; (3) Pairing failure.

(1) refers to the entries that are not included in the training data of Word2Vec model, so that it cannot give a list of similar words because it has not even

seen them. To avoid this type of failure, we could prepare larger data for the Word2Vec model training. As we have used the test data provided by the author, it is hopeful to solve this issue after we change to a more abundant corpus.

(2) refers to the entries whose similar words (candidates) are not found in the GOC. This is a difficult issue when in-domain data serve as the GOC, as it is not easy to obtain a large set of them. In the first place, if a large set of in-domain data is available, we would not have to do the process of data generation.

(3) refers to the entries whose candidates are found in some sentences, but their translations are not found in the corresponding sentences. This is also apparent because words can be translated in different ways. When we get a translation from a simple translation API, only 1 of the different answers is chosen, resulting in a high probability that the target part of the sentence pair does not contain exactly the same word. To solve this issue, a second similar-search can be considered. Instead of the target-candidate, searching for any word inside a similar word list given by the Word2Vec model can increase the possibility of pairing success. Considering accuracy in this step, the min similarity can be set in a comparatively high level. We believe these solutions can bring a lower percentage of generation failure and thus result in a larger size of generated data.

Since the size of IDD seemed the most important factor. We then tried to produce a larger IDD and conduct repeating experiments using it.

4.2 Experiment: Using a Larger IDD

4.2.1 Experiment

Data Preparation

Using the same methods in the previous Section, we picked out a list of 102 word pairs as the in-domain dictionary (English - Japanese) for the following experiments.

The GOC (out-of-domain, in-domain) used were the same as the previous experiments. For test data, we used the whole test dataset (1000 sentences randomly selected from the in-domain dataset) this time.

TABLE 4.8: Data composition of the tuned models and their BLEU Scores on in-domain test data. Model name definition is the same as in Table 4.7. 'List length' is 'numbers of candidates per entry' and 'generation count' is 'maximum number of sentences per candidate' defined in Table 4.3.

Models	list length	generation count	size	test score
base	-	-	0kB	30.00
out1	20	10	1,168kB	26.20
out2	10	10	591kB	27.02
out3	10	20	1,082kB	27.48
in	20	10	320kB	27.05
in	-	-	1,229kB	36.51
out+orig1	20	10	2,397kB	35.43
out+orig2	10	10	1,820kB	35.76
out+orig3	10	20	2,311kB	35.89
in+orig	20	10	1,549kB	35.31

Data Generation

When generating data using the same method with the previous Section, we changed the two parameters in Table 4.3, 'numbers of candidates per entry', which is the length of synonym list given by Word2Vec for each entry of IDD, and 'maximum number of sentences per candidate', which retrict too many outputs for single replacement pair. The settings of theses parameters and the sizes of generation results are listed together with the translation results in Table 4.8.

Results

The base model trained in the previous experiments were used again. Parameters of fine-tuning process also remained the same as in Table 4.6.

The results are listed in Table 4.8.

Unfortunately, all the experimental models worked worse than the model 'orig', which indicated that adding the generated data to the in-domain data brought bad influence to the in-domain translation tests.

The models tuned only by generation data, 'out1', 'out2', 'out3' and 'in' still performed worser than the base model, which indicated the issues we discussed in the last Section were not solved.

The models tuned by a mixture of generated data and in-domain data, 'out+orig1', 'out+orig2', 'out+orig3' and 'in+orig' worked better than the base model, but not as well as 'orig', which probably meant that the generated data were not consistent enough to act as training data.

Furthermore, by comparing 'out+orig1', 'out+orig2' and 'out+orig3', we can find out that the two parameters set here did not show apparent difference. 'out+orig1' performed better than 'in+orig', though little, showing that it was not necessary to use a large in-domain dataset as GOC.

4.2.2 Discussion

We assumed that if we enlarge the size of IDD, we could get better results as generated data would be enlarged accordingly. However, the results did not support our assumption.

The experiment this time even showed worse results than the previous one, as model 'orig' was the best performing model, which indicated the generated data were worse than the original in-domain data, so that adding them into the in-domain data made the scores decrease.

As we have obtained better results in the previous section, we consider the newly added entries in the IDD did not work as we expected. One possible reason is the way we established the IDD. In the previous experiment, we only chose the word that were recognized as unknown words by the base model. However, in order to enlarge the IDD, we had to chose also some of the domain-specialized words even if they were already recognized by the base model. Since we were not able to obtain a suitable dictionary for experiments, we had to continue research using the same IDD.

There were more things to do about raising the quality of generated data. To define 'good' generated data, we believes that consistency and variety are important. High consistency means the generated sentences make sense, which basically focuses on the imported word. If the imported word has the same part of speech of the replaced one, is near in meaning, the sentences will seem consistent. Variety ensures that the data do not contain a lot of simple repetitions, which we have proved not beneficial in Chapter 3.

To improve consistency, we may change the threshold when obtaining the synonym list from Word2Vec model. A high threshold ensures that only the words really similar to the entry word are given. However, it also decreases the size of the generated data to set a higher threshold.

To improve variety, we may set up a count for each original sentence to make sure the sentences are picked in a more dispersive way. We can maintain a list of counts to ensure each sentence in GOC should be picked up to a certain times. When the count of a sentence reaches the upper limit, it will be skipped in the following search.

Furthermore, choosing shorter sentences in the replacement step may bring improvements as NMT system is weak to long sentences compared to shorter ones. Thus, a threshold of the length of original sentence to be picked may work in the step of searching sentences for replacement. However, a low threshold of length may also cause a lower rate of replacement success, resulting in the size of generated data.

To apply the possible solutions listed above, we conducted further experiments.

4.3 Experiment: Adjusting Generation Parameters

4.3.1 Experiment

Defining Parameters

As is discussed in the previous Section, we decided to examine if adjusting the following parameters would improve the generation quality.

- *min similarity*:
  The minimum value of similarity given by Word2Vec between a source-entry and an acceptable source-candidate.
- *max usage count*:
  The maximum times that a sentence in GOC can be used for generation.
- *max length*:
  The maximum length of a sentence in GOC available for generation.

Generation

By adding the parameters, we revised our algorithm for generation. The revised algorithm is described in the following pseudo codes.

By setting different values to these parameters, we conducted extra experiments of generation. The values of parameters are listed together with the translation results in Table 4.8.

Results

Experiments were conducted using the same base model, GOC (only in-domain), parameters of fine-tuning and test data.

The results are listed in Table 4.9.

Pseudo Codes 3 Pairing and replacement between entries of the IDD and their synonyms in GOC using parameters.

Input: *IDD*: List, *GOC*: List, *maxCountPerEntry*: Integer,
minSimilarity: Float,
maxUsageCount: Integer, *maxLength*: Integer
Output: *dataGenerated*: List

```
1: dataGenerated ← new List
2: GOCUsageCountDictionary ← new Dictionary
3: for sentencePair ∈ GOC do
4:     GOCUsageCountDictionary[sentencePair] ←0
5: end for
6: for entryPair ∈ IDD do
7:     sourceCandidateList ← Word2Vec.GetSynonyms(entryPair.source)
8:     for sourceCandidate ∈ sourceCandidateList do
9:         if Word2Vec.GetSimilarity(entryPair.source, sourceCandidate)
       < minSimilarity then
10:            continue
11:        end if
12:        count ← 0
13:        sourceCandidate.target ← Translate(sourceCandidate)
14:        for sentencePair ∈ GOC do
15:            if
       GOCUsageCountDictionary[sentencePair] == maxUsageCount
       or sentencePair.source.length>maxLength then
16:                continue
17:            end if
18:            if sentencePair.source contains sourceCandidate and
       sentencePair.target contains sourceCandidate.target  then
19:                sentencePairGen.source ←
       sentencePair.source.Replace(sourceCandidate, entryPair.source)
20:                sentencePairGen.target ←
       sentencePair.target.Replace(sourceCandidate.target, entryPair.target)
21:                dataGenerated adds sentencePairGen
22:                count ← count + 1
23:                GOCUsageCountDictionary[sentencePair] ←
       GOCUsageCountDictionary[sentencePair] + 1
24:            end if
25:            if count == maxCountPerEntry then
26:                break
27:            end if
28:        end for
29:    end for
30: end for
31: return dataGenerated
```

TABLE 4.9: BLEU Scores of models fine-tuned by mixtures of in-domain data and generated data using different parameters. Model name definition is the same as in Table 4.7.

Models	min similarity	max usage count	max length	size	test score
base	-	-	-	0kB	30.00
orig	-	-	-	1,229kB	36.51
in+orig0	0	∞	∞	1,549kB	35.31
in+orig1	0	∞	100	1,492kB	35.74
in+orig2	0	∞	70	1,320kB	36.00
in+orig3	0.6	∞	∞	1,533kB	35.80
in+orig4	0.7	∞	∞	1,412kB	35.37
in+orig5	0	1	∞	1,296kB	34.90
in+orig6	0	3	∞	1,396kB	35.45

'in+orig0' took no parameters of all the 3 defined above. By comparing the scores of other models with 'in+orig0', we can find out the influence brought by the added parameters.

'in+orig1' and 'in+orig2' took the parameter 'max length', which restricted the maximum length of a sentence in GOC to be picked in the pairing step of generation processes. we can find out that 'in+orig2' performed the best with a value of 70, following by the 'in+orig1' model, still better than 'in+orig0'.

'in+orig3' and 'in+orig4' took the parameter 'min similarity', which got rid of source-candidates that were not really similar with the source-entry, resulting in higher consistency of the generated data. we can find out that 'in+orig3' performed the best with a value of 0.6, following by the 'in+orig4' model nearly the same with 'in+orig0'.

'in+orig5' and 'in+orig6' took the parameter 'max usage count', which ensured each sentence in GOC would not be picked a lot of times, resulting in better variety of the generated data. we can find out that 'in+orig6' performed slightly better than 'in+orig0' with a value of 3. But 'in+orig5' performed worse, possibly because of the very small size of generated results.

4.3.2 Discussion

From the results given above, we can find out the parameters set here have influence on the fine-tuning effect.

Unfortunately, we did not have time to conduct experiments trying more values of the parameters or a combination of different parameters as NMT training costs a lot of time.

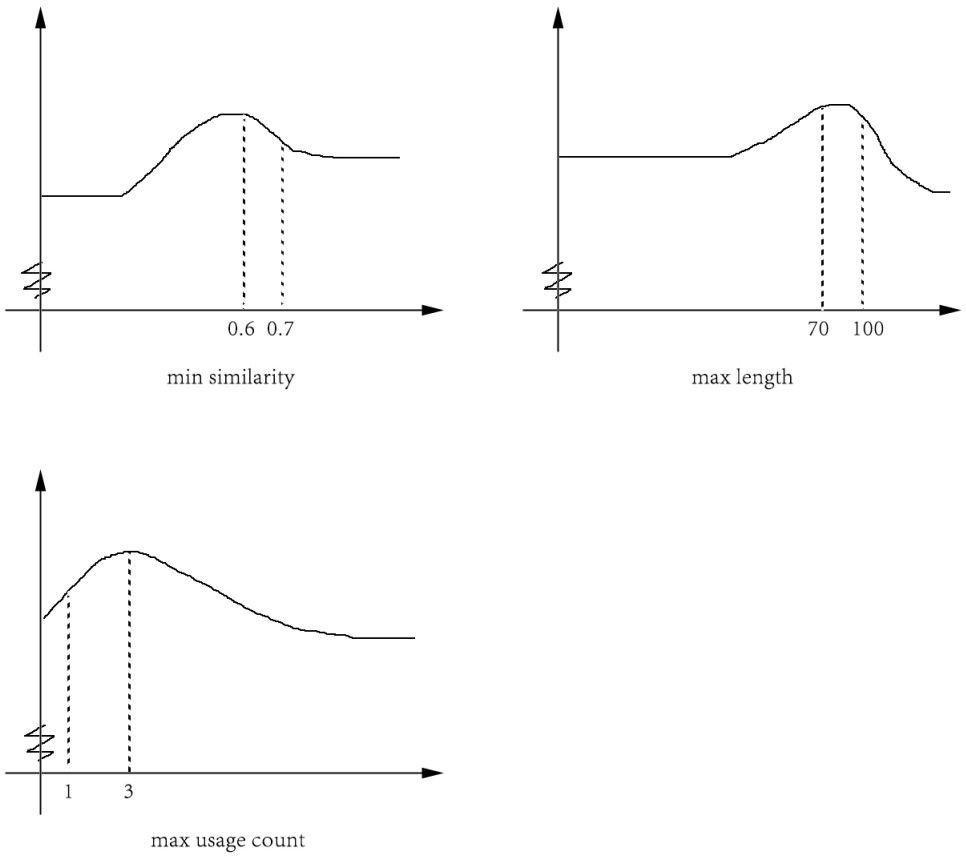

FIGURE 4.3: Inferred relationships between BLEU scores of 'in+orig' types of models and parameters including min similarity, max usage count and max length considering the situation in which the generated data using the proposed method performs better than the original in-domain data.

However, from the results, we can infer that the parameters influence the result in a certain way. Figure 4.3 shows the inferred graphs about the parameters and BLEU scores using the 'in+orig' types of models. This figure and the following discussion on the parameters are based on the situation where 'in+orig' types of models perform better than 'orig' model, just like what we have seen in the first experiment in this Chapter.

For min similarity, a suitable value makes the generated sentences more consistent, resulting in higher scores finally. A very high value, such as setting it to 0.9 disables most of outputs, actually doing nothing.

For max length, it is apparent that a suitable value is able to make generated sentences shorter, resulting in a better translation effect. However, to

setting a very low value also disables most of the generation outputs.

For max usage count, setting it to 1 showed an unsatisfying result in the experiments, but the value 3 worked well. This indicated that repetition is not always unfavorable. Setting it to a very large value does not restrict the repetition at all, resulting in a bad variety in the generated data though.

These parameters show a common trend that there is a peak point to make the fine-tuning result the best. As they basically restrict outputs in different ways, we can expect that combining all the peak point together makes a better score than singly applying one of them.

Furthermore, concerned with the reproducibility of the trends described above, we consider that max length and min similarity should not be influenced by the domain or dataset used in NMT. Instead, language pairs may count a lot. Thus, it is possible to establish default values of them for each language pairs for other users of this generation method.

Besides, we consider that the best max usage count may differ according to the dataset used. If further experiments cannot find a best value, we can use a starting default value and a searching algorithm to find a suitable value.

Although the experiments still did not show a reproducible improvement based on the original in-domain data, we believe there are more points to improve the generation algorithm. For example, we can often recognize similar but not exactly the same sentences in a corpus. They share most of the words as well as the way of word arrangement, but differ in numbers or domain specific words. For these sentences, a simple count to check if the same sentence has been used is not effective in improving data variety. Thus, while picking sentences for replacement, an extra step which checks if the sentence is too similar to any sentences that have been picked, using Word2Vec models.

Chapter 5

Discussion

We conducted experiments of repetition and reordering as methods of data editing. Furthermore, we proposed a generation method using in-domain dictionary and synonyms given by a Word2Vec model. What we want to examine is what makes a dataset 'good' for fine-tuning and according to the results of the different experiments, we are able to summarize as following.

About the appearances of repetition in the corpus for tuning, though the size of dataset is increased, there is no new words and sentences added.

In the experiment of simple repetition, we saw a reduction on the scores after the process. It was quite unexpected because it looked like just an increase of training epoch numbers. However, partial repetition showed an improvement on the fine-tuning effect. It is still not clear why there is apparent difference between simple repetition and proposed partial repetition methods, but we can tell that repetition is not always unfavorable.

In the last part of Chapter 4, the parameter 'max usage count' applied to the experiment was also about repetition. If it is not set, or set to a large value, the generation process tends to create a set of similar sentences when a pairing is completed because similar sentences may occur several times in the GOC, containing the same source-candidate, which actually act as repetitions. We assumed that to set this parameter to 1 would bring the best effect, but the results showed that a value of 3 worked better than 1. Although the generated size counts a lot basically, and a small value makes the outputs less, which seems to account for the reason, as we were getting results basically worse than the original in-domain data, to add nothing should be the best. Thus, the result that the model of value 3 performed better than the model of value 1 possibly means that repetition appeared in the generation process brings good influence to the result.

It is widely believed that shorter sentences are more preferable than longer ones for NMT, with a quite strong trend. In our experiments, this rule was

shown to be right also for fine-tuning. The method of partial repetition contains decompositions of long sentences, which showed good effect on fine-tuning. Besides, in the experiment of generation, when a parameter of max length was applied to the generation process, the scores increased. Although We have shown the feasibility and effect of the partial repetition process, the process costs too much because it requires manual work. In some language pairs with similar sentence composition and ordering, we believe that it is possible to establish automatic decomposition algorithms, such as Spanish and Portuguese. This remains future work.

About the order of the tuning data, we have obtained results in Chapter 3 to show that sorting the data with sentence length increased the scores, no matter in an ascending order or a descending order. We have not understand the reason of the phenomenon.

However, the conclusion here seems to lack reproducibility. When conducting the experiments in Chapter 4, we tried to sort the generated data according to the sentence length, but both of the orders showed a very unsatisfying influence on the results. As the size of the data here was bigger than the one in Chapter 3 for tens of times, we consider that the process may be effective only for a really small dataset, which in fact lacks practicality. Since sorting is a common step for training in many NMT toolkits, and it is believe to prevent wasted calculation in padding shorter sentences to the longest one in a mini-batch (Morishita et al., 2017), we consider that future work is still needed and meaningful for this topic.

Chapter 6

Conclusion

In this study, we have proposed several methods concerning data editing and generation for fine-tuning in NMT.

In Chapter 3, we have discussed 2 techniques, partial repetition and reordering, to conduct certain editing to the data for fine-tuning. Both of them have shown improvements in translation of test data. The best NMT model in the experiments (using partial repetition and ascending reordering) outperforms the simple fine-tuning process by +3.96 BLEU points on test data, which shows the effect of the proposed methods. Though we have conducted the experiments by manual work, we believe that it is possible to establish automatic algorithm for language pairs with more similar sentence order and compositions such as Spanish-Portuguese. This remains future work.

In Chapter 4, we have discussed the proposed method to generate bilingual sentence pairs for fine-tuning, using an in-domain dictionary and a set of original bilingual data. When given an in-domain data as the GOC, the method has proved to succeed in improving the fine-tuning effect by +1.25 BLEU points. In further experiments, though there is no results shown better than the simple fine-tuned model, we have seen influences of the 3 parameters applied to the generation process.

Considering ideal data composition for fine-tuning in NMT, we have discussed about the influences of appearance of repetition, ordering and sentence length. For future work, a set of default parameters can be established to conduct automatic editing and generation using the methods proposed in this study.

Bibliography

Amazon Elastic Compute Cloud (Amazon EC2). Amazon.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.

Cho, Kyunghyun et al. (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *arXiv preprint arXiv:1406.1078*.

Chu, Chenhui, Raj Dabre, and Sadao Kurohashi (2017). "An Empirical Comparison of Simple Domain Adaptation Methods for Neural Machine Translation". In: *arXiv preprint arXiv:1701.03214*.

Google Cloud Translation API. Google.

Graham, Neubig, Mori Shinsuke, and Sasada Tetsuro. *KyTea*.

Hutchins, John (1994). "Machine translation: History and general principles". In: *The encyclopedia of languages and linguistics* 5, pp. 2322–2332.

Klein, Guillaume et al. (2017). "Opennmt: Open-source toolkit for neural machine translation". In: *arXiv preprint arXiv:1701.02810*.

Kusuri-no-Shiori. RAD-AR.

Luong, Minh-Thang and Christopher D Manning (2015). "Stanford neural machine translation systems for spoken language domains". In: *Proceedings of the International Workshop on Spoken Language Translation*.

Luong, Minh-Thang, Hieu Pham, and Christopher D Manning (2015). "Effective approaches to attention-based neural machine translation". In: *arXiv preprint arXiv:1508.04025*.

Mikolov, Tomas et al. (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Morishita, Makoto et al. (2017). "An empirical study of mini-batch creation strategies for neural machine translation". In: *arXiv preprint arXiv:1706.05765*.

NTCIR Patent Machine Translation (NTCIR PatentMT). NTCIR.

Papineni, Kishore et al. (2002). "BLEU: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pp. 311–318.

Rehurek, Radim and Petr Sojka (2010). "Software framework for topic modelling with large corpora". In: *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*, pp. 3104–3112.

Wu, Yonghui et al. (2016). "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144*.

Zhang, Jiajun and Chengqing Zong (2016). "Bridging neural machine translation and bilingual dictionaries". In: *arXiv preprint arXiv:1610.07272*.