Student Name
Xudong Ma

Student ID
822009

Supervisor
Assoc. Prof. Trevor Cohn

Project Title
Close Neighbors search over vector space representations of massive corpora: An application to low-resource NMT

Credit Points
25

Project Type
Conventional Research Project

Subject Code
COMP90055

THE UNIVERSITY OF MELBOURNE

Close Neighbors search over vector space representations of massive corpora: An application to low-resource NMT

Author:
Xudong Ma

Supervisor:
Assoc. Prof. Trevor Chon

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

in the
School of Computing and Information Systems

June 2019

Declaration of Authorship

I, Xudong Ma, declare that this thesis titled, “Close Neighbors search over vector space representations of massive corpora: An application to low-resource NMT” and the work presented in it are my own. I confirm that:

- this thesis does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person where due reference is not made in the text.
- the thesis is 5333 words in length (excluding text in images, table, bibliographies and appendices)

Signed: 马旭东

Date: 05/06/2019

THE UNIVERSITY OF MELBOURNE

Abstract

School of Computing and Information Systems

Master of Information Technology

by Xudong Ma

Neural machine translation (NMT) has been developed a lot in recent years. A well-performed NMT model needs a large amount of parallel data to do the training. However, parallel dataset is hard to obtain for some low-resource languages, such as Nepali and Sinhala. Unfortunately, there are only a few researches on the topic of low-resource NMT at present. Therefore, a research on this topic concerning parallel data harvesting from monolingual datasets is proposed in this report. At the beginning, a sentence embedding model for the pair of source and target language is trained and such model is expected to embed parallel sentences into similar vectors. After that, the sentence embedding model is used to harvest similar sentences from a pair of monolingual datasets in order to expand the parallel dataset. Finally, the expanded parallel dataset is used to train the NMT model. Experiments have been done on two language pairs (German-English and Nepali-English). The result shows that the method proposed in this project is useful on German-English language pair and the BLEU score shows an increasing trend as the size of the expanded parallel dataset increases.

Acknowledgements

Firstly, I would like to express my heartfelt thanks to my supervisor Assoc. Prof. Trevor Cohn from the School of Computing and Information Systems at the University of Melbourne. Trevor provides me with a lot of help throughout the whole semester. He always answered my questions patiently and gave me some guidance, no matter how stupid my questions were.

Secondly, I would like to thank Dr. Matthias Petri from the school of computer and information systems at the university of Melbourne. His model gave me great inspiration as the starting point of the project and aroused my keen interest in the whole project.

Finally, I would like to express my gratitude to my roommates because they help me grow under the pressure. We often encourage and help each other in our study and life. When I was in a bad mood or under too much pressure, my roommates would chat with me, which made me more efficient to complete my study tasks. Thanks

Author Xudong Ma

Contents

List of Figures

List of Tables

Abbreviations

NMT	**N**eural **M**achine **T**ranslation
BLUE	**BiL**ingual **E**valuation **U**nderstudy
OpenNMT	**O**pen-source **N**eural **M**achine **T**ranslation
GRU	**G**ated **R**ecurrent **U**nit
NLP	**N**atural **L**anguage **P**rocessing
RNN	**R**ecurrent **N**eural **N**etworks
ACL	**A**ssociation for **C**omputational **L**inguistics
KNN	**K** **N**earest **N**eighbour
DE	German
EN	**EN**glish
NE	**NE**pali
BF	**B**efore **F**iltering
AF	**A**fter **F**iltering
SE	**S**entence **E**mbedding
LR	**L**earning **R**ate
NNS	**N**umber of **N**egative **S**amples
BS	**B**atch **S**ize
ML	**M**argin of **L**oss function
TD	**T**raining **D**ataset
DD	**D**evelopment **D**ataset
PPS	**P**erfectly **P**arallel **S**entences
AR	**A**verage **R**ank
AS	**A**verage cosine **S**imilarity

Chapter 1

Introduction

1.1 Research Background

According to the statistics of Linguistic Society of America, there are more than 6800 languages in the world and about a quarter of them are spoken by fewer than 1000 people [Anderson, 2010]. This makes it unrealistic to rely all language translation on manual work. Therefore, machine translation becomes more and more popular and demanded in today's world. Generally, there are two main machine translation methods which are Statistical Machine Translation (SMT) and Neural Machine Translation (NMT) [Sutskever et al., 2014]. Among them, some models of NMT are the state-of-art on doing machine translation for many language pairs. For this reason, a NMT based model is applied to do a sentence to sentence translation for the translation part of this project.

In order to train a well performed NMT model, a large amount of parallel sentences need to be applied as the training dataset. However, there are not sufficient parallel sentence pairs for some low-resource languages, such as Nepali, Sinhala, Afrikaans, et al. At present, only a few researches have been done on the topic of low-resource language NMT. In 2016, Zoph, B. and Yuret, D. proposed a method for this topic [Zoph et al., 2016]. They do a transfer learning style NMT on low-resource language pairs. Firstly, they train a model on a high-resource language pair and then retrain this trained model on low-resource languages pairs. In order to evaluate the performance of their method, they applied Bilingual Evaluation Understudy (BLEU) of which the main idea is to compare the differences between the output of the NMT model and the result of manual translation [Papineni et al., 2002]. A higher BLEU score means the machine translation results is more similar to manual translation and the machine translation performs better. Finally, their method improves the BLEU score by 1.3 on average. Although they have made some improvements on the NMT of low-resource languages, the problem of the lack of parallel data for low-resource language pairs is still there. This makes the performance of their method unstable because their method needs a high similarity between the parent language used to pre-train the model and the child language which is a low-resource language.

1.2 The Contribution of this Project

Different from the methods proposed by Zoph, B. and Yuret, D, the main idea of this project is to improve the problem of the lack of parallel data for low-resource language pairs fundamentally so as to train a better NMT model by using a larger training dataset finally. Although it is hard to obtain parallel data for many language pairs, monolingual resources are always abundant. Therefore, this project takes monolingual corpus into account and harvests parallel data from it. Two language pairs, German-English and Nepali-English, are chosen to do experiments for this project. On one hand, Nepali is a low-resource language which is the focus of this project. On the other hand, the language pair of German and English is used as a control group. Meanwhile, the translation performance on German and English language pair can be treated as a limit of this method to some extent because high quality German and English parallel dataset is easy to obtain. And such high quality parallel dataset is more likely to make the sentence embedding model more accurate. Logically, the accuracy of the sentence embedding model decides the quality of training dataset used to train the NMT model so as to decide the machine translation performance.

1.3 Report Structure

Chapter 1 is mainly used to introduce the background of this research. Following that, the main methodology of this project is discussed in chapter 2. Afterwards, chapter 3 shows the experiments part of this project which includes the obtaining of the datasets, the training of models and the evaluation of the results. Finally, a summary for this work and a prospect of the future works are discussed in the conclusion part.

Chapter 2

Methodology

2.1 Word Embedding

Generally, the first step of doing NMT is to do word embedding which means representing words into high dimensional vector space. Neural network based word embedding method is used for this project because it takes the semantic meanings of words into account. After doing such word embedding, words with similar semantic meanings, such as “hi” and “hello”, would have similar vector representations. Besides, the distances between similar word pairs would be alike, such as the distance between “China” and “Beijing” would be very similar to the distance between “Australia” and “Sydney” as they both represent the relationship between Country and the capital of it.

It is obvious that neural network based word embedding needs a large amount of monolingual dataset to train a model for each language in order to get good vector representations for words. However, the huge amount of training dataset means this process is time consuming and needs to do a lot experiments before getting a well-performed model. Therefore, it is hard to train a good word embedding model for each language by ourselves. Fortunately, there are already some well-performed pre-trained wording embedding models available, such as word2vec, fastText, Glove et al. Among them, fastText is chosen for this project because it covers about 157 languages which include all the languages (German, English, Nepali) used in this project. After doing word embedding using the pre-trained model from fastText, each word inside the datasets would be represented as a vector of 300 dimension.

2.2 Sentence Embedding

After doing word embedding, the next step of this project is to do sentence embedding. This is a process of representing sentences into vector space. Similarly to word embedding, the target of sentence embedding is to make sentences with similar meanings have similar vector representations. In order to complete this task, a model based on Gated Recurrent Unit (GRU) [Dey and Salemt, 2017] network is designed and coded by Dr. Matthias Petri from the Natural Language Processing (NLP) group of the School of Computing and Information Systems at the University of Melbourne. After doing sentence Embedding using Matthias's model, each sentence inside the datasets is represented as a 300 dimensional vector.

2.2.1 Gated Recurrent Unit (GRU) network

Recurrent Neural Networks (RNN) [Cho et al., 2014] is the state of art to capture the sequence relationships between words inside a sentence. However, gradient vanishing or gradient exploding may cause the failure of the training of RNN [Bengio et al., 1994]. Gradient clipping can be used to improve the problem of gradient exploding. In comparison, gradient vanishing is much more difficult to handle.

GRU is proposed as a variant of traditional RNN in 2014 and it can improve the problem of gradient vanishing to some extent [Cho et al., 2014]. Similarly to Long Short Term Memory (LSTM) network, which introduces three gates [Gers et al., 1999], GRU network introduces two gates, the reset gate and the update gate, to make the network enable to forget some parts of the inputs $\boldsymbol{x}_t$ and the accumulated hidden states $\boldsymbol{h}_{t-1}$ so as to improve the problem of gradient exploding. More importantly, such forgetting mechanism is also more in line with semantics. The structure of a GRU cell is demonstrated as figure 2.1 and the computation process of time step t of GRU can be simplified as follows [Heck and Salem, 2017]:

$$\boldsymbol{z}_t = sigm(\boldsymbol{W}_{xz}\boldsymbol{x}_t + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{r}_t = sigm(\boldsymbol{W}_{xr}\boldsymbol{x}_t + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1})$$

$$\widetilde{\boldsymbol{h}}_t = tanh(\boldsymbol{W}_{xh}\boldsymbol{x}_t + \boldsymbol{r}_t \odot \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{h}_t = (\mathbf{1} - \boldsymbol{z}_t) \odot \widetilde{\boldsymbol{h}}_t + \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1})$$

Among these formulas, $\odot$ demonstrates element-wise multiplication. $\boldsymbol{x}_t$ and $\boldsymbol{h}_t$ are the input vector and the accumulated hidden state vector at moment t, respectively. All $\boldsymbol{W}$ are different argument matrices which should be adjusted during the process of the training of the model. Additionally, "sigm" and "tanh" represent activation functions ***sigmoid*** and ***tanh*** separately. $\boldsymbol{z}_t$ represents the update gate and $\boldsymbol{r}_t$ represents the reset gate. Meanwhile, $\widetilde{\boldsymbol{h}}_t$ represents a pre-hidden state vector of the time step t. $\mathbf{1}$ is a unit vector.

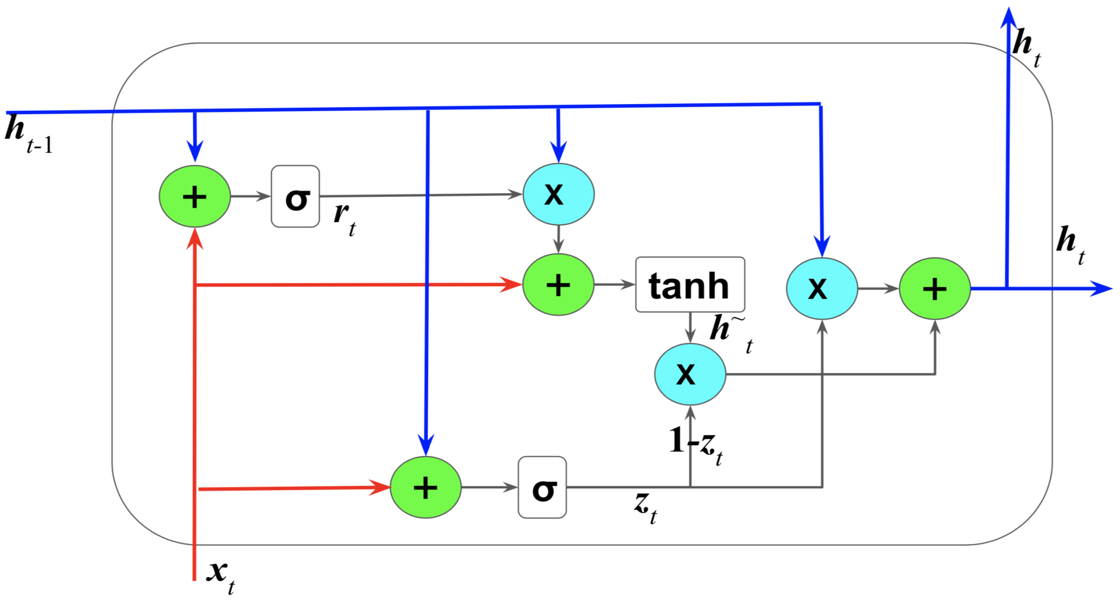

FIGURE 2.1: The structure of a GRU cell

2.2.2 Bidirectional GRU

Unidirectional GRU performs well on obtaining the front-to-back sequence information of a sentence but it cannot capture the back-to-front sequence information properly which is also important in the real world. For example, in the sentence "I was a student ten years ago", "was" is used rather than "is" because "ten years ago" appears at the end of the sentence. Therefore, bidirectional rather than unidirectional GRU networks are used to do sentence embedding for this project.

2.2.3 The Sentence Embedding Model

As mentioned before, this project applies Dr. Petri's model to do sentence embedding which is based on bidirectional GRU networks. For each source language sentence x, it is provided with some corresponding sentences y_i to form a group of data used for training the model. A perfectly parallel target language sentence for x is set to be the first corresponding sentence y_1. Additionally, some random chosen target language sentences are selected from a monolingual dataset as negative samples. These negative samples are the rest of y_i except y_1. For example, if the number of negative samples is 20, negative samples would be y_2 to y_{21}. For each source sentence x there will be 21 target sentences, y_1 to y_{21}, correspond to it. All these 22 sentences, x and y_1 to y_{21}, are treated as a group of training data. On one hand, x is embedded using a GRU, but before passing the GRU network it would also be done a linear variation in order to simulate the process of decreasing the differences between word embedding for different languages. On the other hand, y_1 to y_{21} are embedded using another GRU. After doing sentence embedding, the cosine similarities between x and each sentence from y_1 to y_{21} are calculated. Logically, y_1 is expected to be the most similar sentence from y_1 to y_{21}. As for the loss function, it is designed based on the similarity scores obtained by y_1 to y_{21}. Basically, MarginRankingLoss of pytorch is applied as the core of this loss function. MarginRankingLoss can be presented as the formula 2.1 [Rao and McMahan, 2019]:

$$loss(s_1, s_2, p) = max(0, -p \times (s_1 - s_2) + margin)$$

where s_1 and s_2 are two scalars obtained from the training process. p equals 1 or -1. When s_1 is bigger than s_2, p equals 1. Otherwise, p equals -1. $margin$ is the minimum distances between s_1 and s_2 . It is clear that the loss would be 0 when s_1 - s_2 > $margin$. Otherwise, there will be a loss for the pair of s_1 and s_2. Besides, the value of the loss is proportional to the differences between s_1 and s_2.

As mentioned before, for each group of training data which contains a source sentence x and 21 target sentences y_1 to y_{21}. We set s_1 to be the cosine similarity score between x and y_1. The similarity scores between x and y_2 to y_{21} are set to be s_2 separately. Finally, all these 21 loss values are summed together to represent the total loss value of the current group of training data. The structure of the word embedding model can be demonstrated as figure 2.2 . In this figure, Nepali is set to be the source language and English is set to be the target language as an example.

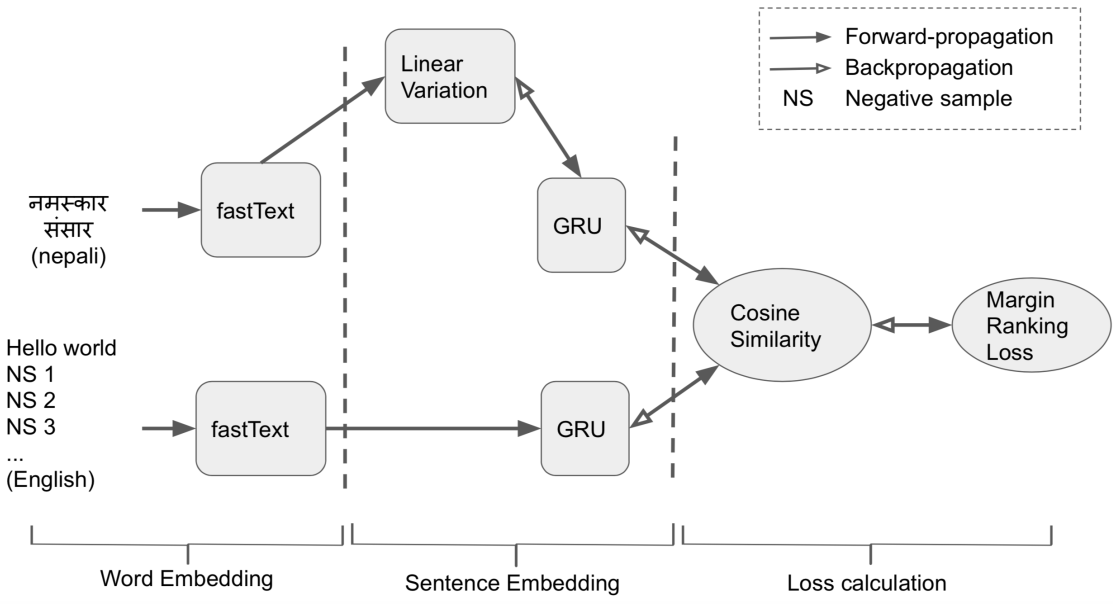

FIGURE 2.2: The structure of the sentence embedding model

2.3 Parallel Data Harvesting

The sentence embedding model is expected to represent a pair of parallel sentences of source and target languages into similar vectors which means the cosine similarity of their vector representations would be close to 1. Then a pair of monolingual datasets for the source language and the target language are applied to harvest parallel sentences. For each sentence inside the source monolingual dataset, sentences inside the target language dataset are retrieved and their cosine similarities with the source sentence would be calculated. Finally, the first K sentences which gets a higher similarity score than the threshold would be selected to form K parallel sentences with the source sentence. The reason for setting a threshold rather than harvesting the nearest neighbour for each source sentence is that nearest neighbour searching leads to a time complexity of O(n). Meanwhile, in order to find a quite similar target sentence for each source sentence, a target monolingual dataset of ten times as big as the source monolingual dataset would be applied so that n is a huge number normally, which makes this task almost impossible to complete. Moreover, the threshold is set according to the average value of all the similarity scores of all sentence pairs inside the development parallel dataset.

Although the newly harvesting parallel sentences may not be parallel perfectly, they would still have similar meanings theoretically which means they are parallel in some way. For example, we would like to get parallel sentences such as “Ich mag Apfel” (German), which means “I like apples”, and “I love apples”.

2.4 Translation

As for machine translation, a model from Open-source Neural Machine Translation (OpenNMT) is applied for this project because this model performs well on many language pairs. Firstly, the parallel dataset used to train the sentence embedding model, assuming it is $train_1$, is used to train a translation model ($model_1$). After that $model_1$ is applied to translate the source sentences in the test dataset ($test_1$) into target language. On the other hand, the newly harvesting parallel dataset obtained from the previous part is mixed together with $train_1$ and the mixture is assumed to be $train_2$. It is clear that $train_2$ contains more parallel sentences than $train_1$. The training set $train_2$ is applied to train another OpenNMT translation model ($model_2$) for doing translation. Similarly, $train_2$ would be used to do translation for the source sentences inside the same test set, $test_1$ as well. Finally, the translation results from $train_1$ and $train_2$ would be compared.

2.5 Evaluation Strategy

Evaluations are introduced for two part of this project. On one hand, the result of the sentence embedding model is evaluated as a milestone. At this stage, a pair of development parallel datasets are used. It is clear that for each source sentence ($source_1$) there is a perfectly parallel target sentence ($target_1$) corresponding to it. Besides, for each $source_1$, all the sentences inside the target language dataset, including $target_1$, are embedded using Petri’s model and then are used to calculate a group of cosine similarity values with embedded $source_1$. After that, all these similarity values are ranked and the rank between embedded $source_1$ and embedded $target_1$ is grabbed particularly. Finally, all the grabbed rank values are averaged to evaluate the performance of the sentence embedding model. On the other hand, the purpose of training the sentence embedding model is to harvest parallel sentences. Therefore, after doing the evaluation for the sentence embedding model, it would be used to harvest parallel dataset. And the newly harvest parallel dataset would be used to train the NMT model. Therefore, the second part of evaluation would be focused on the performance of the translation model. BLEU score is introduced for such evaluation.

Chapter 3

Experiments

3.1 Datasets

The experiments are carried out on two language pairs, German-English and Nepali-English. The datasets are obtained from Association for Computational Linguistics (ACL) 2019 FOURTH CONFERENCE ON MACHINE TRANSLATION (WMT19). However, there are different sources for them. The information regarding datasets is shown in table 3.1.

Language Pair	Dataset	Source	Number of Sentences	
German-English	Parallel	News commentary	0.28m	
	Monolingual DE	News commentary	38.96m	
	Monolingual EN	News commentary	26.86m	
Nepali-English	Parallel	Global Voices	2892	0.38m
		Wikipedia	5394	
		GNOME	0.37m	
	Monolingual NE	Commoncrawl	1.00m	
	Monolingual EN	Commoncrawl	10.00m	

Table 3.1: The information of datasets

where “DE”, “EN”, “NE” represents German, English and Nepali respectively. All the monolingual datasets derive from news commentary or commoncrawl which is a community who crawls news articles from news sites all over the world. And the reason for choosing such data sources is that sentences from news articles would be less likely to make errors. Meanwhile, they can cover almost all fields in human’s life.

About 80% of the sentences inside the parallel datasets are used as the training set. Almost 20% of the parallel sentences are applied as the test set and a fraction of the sentences are grabbed as the development dataset. In order to improve the performance of the system, two steps of filtering are done on the datasets. Firstly, only those sentence pairs whose lengths are between 5 and 64, both for the source language and for the target language, are kept. Otherwise, they would be removed. After that, a duplication removal is done as the second step of data filtering. The size of the datasets is demonstrated in table 3.2. However, even after doing filtering, the quality of the parallel datasets for Nepali and English are still not as good as that of German and English language pair. Because about 97.83% of the parallel datasets for Nepali and English come from GNOME which includes many fake sentence pairs, such as a pair of parallel sentences with the English sentence The game file to use which even is not a real sentence. However, although it is not high quality sentence, it is still kept inside the training dataset because a strict filtering would make the size of the dataset shrink rapidly and will result the number of parallel sentences inside Nepali and English parallel dataset decreases to a figure smaller than 10k. It is clear that such a mini size of dataset is extremely small for training a good performance NLP model.

Language Pair	Dataset	Sentences BF	Sentences AF
German-English	Training	0.22m	0.21m
	Development	4000	3854
	Test	60.00k	58.11k
Nepali-English	Training	0.30m	19.75k
	Development	2095	136
	Test	76.00k	4937

Table 3.2: The size of datasets

where "BF" and "AF" represent before and after filtering, respectively.

3.2 Model Training

3.2.1 Sentence Embedding Model Training

For German and English, the sentence embedding model is trained in two steps. The parameters for these two steps are set similarly except the value of the learning rate and the number of learning epochs. As for Nepali and English, the training process is designed similarly with that of German and English. The training parameters are shown in table 3.3.

Language Pair	Training	SE Dimension	NNS	BS	ML	Epochs	LR
German-English	First Step	300	20	100	0.5	20	0.0002
	Second Step	300	20	100	0.5	10	0.0001
Nepali-English	First Step	300	50	50	0.5	20	0.0001
	Second Step	300	50	50	0.5	10	0.00005

TABLE 3.3: Training parameters for sentence embedding model

where "SE" means sentence embedding. "LR" means learning rate. "NNS" means the number of negative samples. "BS" represents batch size and "ML" represents the margin value for the loss function.

3.2.2 Translation Model Training

As mentioned before, for each language pair, the pre-filtered parallel dataset is used to train the translation model from OpenNMT and this model is treated as a baseline. Afterwards, an expanded training dataset is used to train another translation model of OpenNMT. Finally, these two models are applied to do translation task on a same test dataset and the translation results are compared based on their translation BLEU score. In order to train the dataset sufficiently, the numbers of training steps are set differently until the translation accuracy defined by OpenNMT, which is an output of translation model, is stable at around 70%.

3.3 Evaluation

There are two steps of evaluations for this project. The first step is the evaluation of the sentence embedding model provided by Matthias Petri. As mentioned before, the sentence embedding model is evaluated based on an average ranks value. Meanwhile, in order to add more evidence for the evaluation, I also calculate the percentage of target sentences which get a rank of exact 1, which means it is the most similar sentence with the source sentence. Additionally, the average similarity score of the perfectly parallel sentences is also computed in order to provide guidance for setting the value of similarity threshold and the value of K which represents how many target sentences are planed to be harvested for each source sentence. The results are shown in table 3.4.

Language Pair	TD	DD	Percentage of PPS	AR	AS
German-English	0.21m	3854	91.36%	5.86	0.82
Nepali-English	19.75k	136	17.71%	30.23	0.56

Table 3.4: Performance of the sentences embedding model

where "TD" represents the size of the training dataset and "DD" means the size of the development dataset. "PPS" means perfectly parallel sentences. "AR" represents the average value of ranks and "AS" means the average value of cosine similarity score.

From table 3.4, it is clear that the sentence embedding model performs quite well on the development parallel dataset of German and English. Most of the perfectly parallel target sentences acquire quite high cosine similarity scores and even the average of them is up to 0.82. Therefore, the similarity threshold for harvesting parallel sentences for German and English is set to 0.82 and only the first target sentence which gets a higher similarity score than the threshold would be returned, which means K is set to 1 here. On the other hand, the sentence embedding model does not perform so good on Nepali and English. As shown in table 3.4, the average similarity score for perfectly parallel Nepali-English sentence pairs is only 0.56 which is far away from 1. Therefore, the similarity threshold is set to 0.56 and K is assigned a higher value 3, in order to improve the probability of harvesting real similarly parallel sentences.

The second step of evaluation is applied on the translation performance. As discussed before, the filtered but unexpanded parallel training dataset is used to train a model as the baseline. Besides, the parallel sentence harvesting algorithm of this project is used to double and triple the baseline training dataset respectively. All training datasets are trained sufficiently using the translation model from OpenNMT and the result is shown in table 3.5.

From table 3.5, it is clear that the parallel dataset harvesting algorithm of this project performs well on German and English. The double expanded training dataset improves the BLEU score by 2.54 and the triple expanded training dataset improves the BLEU score by 4.74. It is observed that there is an increasing trend of the translation BLEU

Language Pair	Threshold	K	Dataset	Training	Test	BLEU
German-English	0.82	1	Baseline	0.21m	58.11k	28.71
			Double expanded	0.43m		31.25
			Triple expanded	0.63m		33.45
Nepali-English	0.56	3	Baseline	19.75k	4937	8.63
			Double expanded	39.50k		8.13
			Triple expanded	59.35k		7.33

Table 3.5: Translation performance of different size of training datasets

score as the size of the expanded training increases for German and English language pair. However, the method of this project performs badly on Nepali and English which even decreases the translation BLEU score. Besides, there is a decreasing trend of the BLEU value as the size of the expanded training training increases for Nepali and English. In conclusion, this results is reasonable because it conforms to the performances of the sentence embedding model on these two language pairs.

Chapter 4

Conclusion

4.1 Summary

As shown from the results of the previous part, Petri's sentence embedding model performs quite well on the language pair German-English and it can be used to harvest high quality parallel dataset for German and English. However, it cannot embed Nepali-English properly, which results the expanded parallel dataset performs not so good on training the translation model. I believe the reason is that the amount of training dataset for sentence embedding model of Nepali and English is too small to train a well-performed model because it is almost 10 times smaller than the dataset for training German-English model. This results the trained sentence embedding model for Nepali and English cannot calculate high quality sentence similarities and makes the newly harvesting parallel sentences too noisy to use. As a result, such noisy parallel dataset makes the translation BLEU score smaller instead of becoming bigger.

4.2 Future works

4.2.1 Neighbours Search Algorithm

In this project, a threshold is set and the first K target sentences which get high similarities scores with the source sentence than the threshold are selected. However, K nearest neighbours (KNN) may be a better choice. The main reason for not using it is that its time complexity is unacceptable. However, there are some efficient KNN searching algorithms to improve the problem of time consuming. For example, a smarter approach by Y. A. Malkov [Malkov and Yashunin, 2018] for the calculation of KNN and this approach can decrease the time complexity of computing KNN from O(n) to O(log n). Therefore, this method can be applied in the future.

4.2.2 Parallel Data Harvesting Method

On the other hand, I only harvest a similar target sentence y from a target monolingual dataset for each source sentence x from a source monolingual dataset in this project. However, Professor Trevor also suggests several different ways to harvest parallel sentences. For example, for each sentence pair (x, y) from the parallel dataset, we can harvest similar sentences x' and y' for x, where x' and y' are in the source and target language, respectively. Then new pairs (x', y) , (x, y') and (x', y') arise. Meanwhile, the same operations can also be done on y. Additionally, prototype editing supported by K. Guu [Guu et al.,2018] can also be applied. For each x, we find its nearest neighbor in the target language y'. Then a monolingual NMT model for (y', y) can be trained. Superficially, although the number of parallel pairs used to train NMT remains the same, all the monolingual resources are still taken into count.

Bibliography

[1] Stephen R Anderson. How many languages are there in the world. *Linguistic Society of America*, 2010.

[2] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[3] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. *arXiv preprint arXiv:1604.02201*, 2016.

[4] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[5] Rahul Dey and Fathi M Salemt. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1597–1600. IEEE, 2017.

[6] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[7] Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5 (2):157–166, 1994.

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[9] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[10] Joel C Heck and Fathi M Salem. Simplified minimal gated unit variations for recurrent neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1593–1596. IEEE, 2017.

[11] Delip Rao and Brian McMahan. *Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning.* " O'Reilly Media, Inc.", 2019.

[12] Yury A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[13] Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *Transactions of the Association of Computational Linguistics*, 6:437–450, 2018.