



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

박사학위논문

문맥 임베딩을 이용한 베트남-한국어
신경망기계번역의 성능 향상

**ENHANCING THE PERFORMANCE OF
VIETNAMESE-KOREAN
NEURAL MACHINE TRANSLATION USING
CONTEXTUAL EMBEDDING**

울산대학교 대학원
전기전자컴퓨터공학과
VU VAN HAI

Doctor of Philosophy

**ENHANCING THE PERFORMANCE OF
VIETNAMESE-KOREAN
NEURAL MACHINE TRANSLATION USING
CONTEXTUAL EMBEDDING**

The Graduate School of the University of Ulsan
Department of Electrical, Electronic and Computer Engineering

Vu Van Hai

문맥 임베딩을 이용한 베트남-한국어
신경망기계번역의 성능 향상

**ENHANCING THE PERFORMANCE OF
VIETNAMESE-KOREAN
NEURAL MACHINE TRANSLATION USING
CONTEXTUAL EMBEDDING**

지도교수 옥철영

이 논문을 공학박사학위 논문으로 제출함

2022년 06월

울산대학교 대학원
전기전자컴퓨터공학과
Vu Van Hai

**ENHANCING THE PERFORMANCE OF
VIETNAMESE-KOREAN
NEURAL MACHINE TRANSLATION USING
CONTEXTUAL EMBEDDING**

Supervisor: Prof. Cheol-Young Ock

A Dissertation

**Submitted to
the Graduate School of the University of Ulsan
In Partial Fulfillment of the Requirements
for the Degree of**

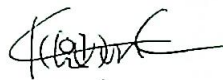


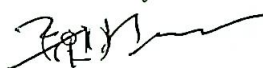

Doctor of Philosophy

by

Vu Van Hai

**Department of Electrical, Electronic
and Computer Engineering
Ulsan, Korea
Jun 2022**

Vu Van Hai 의 공학박사 학위논문 을 인준함

심사위원장	권영근	
심사위원장	옥철영	
심사위원장	정진호	
심사위원장	류범모	
심사위원장	김영길	

울산대학교 대학원
2022년 06월

**Enhancing the performance of
Vietnamese-Korean Neural Machine Translation
using Contextual Embedding**

by

Vu Van Hai

Submitted to the Department of Electrical, Electronic and Computer Engineering
on June 2022, in partial fulfilment of the requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

Since deep learning was introduced, a series of achievements have been published in the field of automatic machine translation (MT). However, Vietnamese-Korean MT systems face many challenges because of a lack of data. In this research, we built the open extensive Vietnamese-Korean parallel corpora for training MT models consisting of over 412 thousand sentence pairs.

Besides, the problem of multiple meanings of words depending on their contexts leads to difficulty to understand the meaning of the corpus for MT. This dissertation discusses a method of applying a linguistic annotation named Part-of-Speech (POS) tagging to Vietnamese sentences to improve the performance of Vietnamese-Korean MT systems. The experimental results indicate that tagging POS in Vietnamese sentences can improve the quality of Vietnamese-Korean Neural MT (NMT) in terms of the Bi-Lingual Evaluation Understudy (BLEU) and Translation Error Rate (TER) score. After applying

POS tagging to the Vietnamese corpus, our Vietnamese-Korean MT system improved by 1.07 BLEU points and 2.96 TER scores, respectively.

In addition, in recent years, a state-of-the-art context-based embedding model called BERT introduced by Google has appeared in the MT models in different ways to boost the accuracy of MT systems. The BERT model for Vietnamese has been built up and significantly improved in natural language processing (NLP) tasks such as POS, NER, dependency parsing, and natural language inference. This dissertation discusses a method for applying POS tagging that is also developed based on the BERT model to Vietnamese sentences to improve the performance of Vietnamese-Korean MT systems. Moreover, our research experiment injected the Vietnamese BERT into the NMT model where the BERT model for Vietnamese is concurrently connected to both encoder layers and decoder layers in the NMT model. MT results show that using the contextual embedding model significantly enhances the performance of Vietnamese-Korean MT by 2.78 BLEU points at the sentence-level and 3.01 BLEU points at the document-level, respectively.

Thesis Supervisor: Cheol-Young Ock

Title: Professor

Acknowledgements

I first want to thank my parents for the wonderful things they have done for me. And, I would like to thank my wife for her sacrifices, encouragement and delicious meals.

I especially would like to express my deep gratitude to my advisor, Professor Ock, Cheol-Young, for his guidance, support and concern for me and my family during our time living in Korea. Living and working in a new country is always a challenge; thanks to his enthusiastic help and his generosity, my family quickly adapted to life in Korea and we had great experiences in this beautiful country.

I would also like to thank other members of my graduate committee Professor Kwon, Young-Keun, Doctor Kim, Young-Kil, Professor Ryu, Pum-Mo, and Professor Chung, Jin-Ho for their concerns, advice, and suggestions on my research.

I am thankful to all members of the Korean Language Processing Laboratory at the University of Ulsan for their kindness and friendship. They helped me a lot to be familiar with life in Korea and shared interesting things about life and research. I would like to thank Doctor Nguyen Quang Phuoc and Doctor Vo Anh Dung for their great help when I live and study in Korea.

My research was supported by BK 21 PLUS Project. Without that financial support, I can hardly complete this research. I do acknowledge the finance, academic, and technique support from the staff of the University of Ulsan.

Contents

Abstract	i
Acknowledgements	iii
Contents	iv
List of figures	viii
List of tables	ix
Introduction	1
1.1. Motivation	1
1.2. Challenges and solutions	3
1.3. Related works	4
1.3.1. Building machine translation systems.....	4
1.3.2. Applying the annotation to parallel corpus	6
1.3.3. Building and applying contextual embedding model	7
1.3.4. Building parallel corpus for machine translation.....	8
1.4. Dissertation Outline.....	11
Backgrounds	12
2.1. Word embedding	12
2.1.1. General word embedding.....	12
2.1.2. Classical word embedding.....	13
2.1.3. Neural word embedding.....	14
2.2. Neural machine translation	16

2.2.1. Recurrent neural network.....	17
2.2.1. Encoder	18
2.2.2. Decoder	19
2.2.3. Attention Mechanism.....	20
2.3. Transformer model	23
2.3.1. The encoder of the transformer.....	24
2.3.2. The decoder of the transformer.....	27
2.4. Linguistic annotation	31
2.4.1. Part of Speech	32
2.4.2. Named Entity Recognition.....	33
2.4.3. Morphological Analysis.....	33
2.4.4. Word Sense Disambiguation	34
2.5. Summary.....	36
Pre-trained BERT model and Vietnamese Pre-trained BERT variant	38
3.1. Architecture of the BERT model.....	38
3.2. Pre-trained BERT	40
3.3. Pre-trained BERT variants.....	44
3.3.1. ALBERT	44
3.3.2. RoBERTa.....	45
3.3.3. Multilingual BERT	46

3.4. Vietnamese Pre-trained BERT	47
3.5. Summary.....	47
Vietnamese Part of Speech.....	49
4.1. Tagging POS using Bi-directional LSTM model	49
4.1.1. Bidirectional LSTM.....	49
4.1.2. Bi-LSTM-CRFs for POS Tagging.....	53
4.2. Tagging POS using Pre-trained BERT model.....	54
4.3. Summary.....	57
Vietnamese-Korean Parallel Corpus.....	58
5.1. Parallel Corpus Acquisition.....	58
5.2. The Parallel Corpora Analysis.....	62
5.2.1. Applying Morphological Analysis and Word-sense Annotation for Korean sentences	62
5.2.2. Applying Word Segmentation and POS for Vietnamese sentences	64
5.3. The detail of Vietnamese-Korean parallel corpora	64
5.4. Summary.....	66
Vietnamese-Korean Machine Translation.....	68
6.1. Models of Vietnamese-Korean Machine Translation.....	68
6.1.1. Machine translation with NMT model.....	69
6.1.2. Combination of BERT and NMT	69
6.1.3. SMT and BPE	71

6.2. Result Evaluations	72
6.3. Discussion.....	73
6.3.1. Comparision with SMT and BPE	73
6.3.2. Impact of Vietnamese POS tagging.....	74
6.3.3. Impact of Vietnamese BERT on Neural Machine Translation.....	75
6.4. Summary.....	76
Conclusions.....	78
7.1. Achievements	78
7.2. Future work	79
Appendix A.....	81
Appendix B.....	83
Bibliography.....	84

List of figures

Figure 1. Vietnam FDI 2020.....	2
Figure 2. Word2Vec architecture.....	15
Figure 3. Structure of a recurrent neuron.....	17
Figure 4. Encoder structure.....	18
Figure 5. Decoder structure.....	20
Figure 6. Attention mechanism structure.....	21
Figure 7. A simple demonstration of the encoder.....	24
Figure 8. Encoder x with the add and norm component.	26
Figure 9. A simple demonstration of the decoder.....	27
Figure 10. A simple demonstration of the decoder.....	29
Figure 11. The combination of encoder and decoder in transformer model.....	31
Figure 12. A simple demonstration of the BERT model. L and m are the number of the encoder and the length of the input sentence, respectively.....	39
Figure 13. Generating representation of a sentence with BERT-base.....	40
Figure 14. The representation of the input.....	41
Figure 15. The strategy of NSP.....	43
Figure 16. An LSTM cell.....	50
Figure 17. Structure of Bi-LSTM.	53
Figure 18. Bi-LSTM-CRFs for POS.....	54
Figure 19. Illustration of POS tagging with BERT.....	55
Figure 20. The acquisition process of the UPC.	58
Figure 21. Sentence length distributions of the Vietnamese-Korean parallel corpus.....	66
Figure 22. A demonstration of translation of a Vietnamese sentence to a Korean sentence with the PhoBERT model.	69

List of tables

Table 1. Top 5 foreigners with Korean nationality.....	1
Table 2. A brief comparison of NMT, RBMT and SMT.....	23
Table 3. Some examples of word senses	35
Table 4. Comparison the result of BERT and some other systems by F1.	44
Table 5. Accuracy of the POS tagging using M-BERT.....	46
Table 6. Performance of some Vietnamese POS tagging systems by F1 score	56
Table 7. Form of Vietnamese sentences after applying POS tagging.....	56
Table 8. Performance of some Vietnamese NER tagging systems by F1 score	57
Table 9. Morphological analysis and word-sense annotated sentence.	62
Table 10. The variation in the number of token and vocabulary in UPC after Morph. Ana.	63
Table 11. The result of the Morph. Ana. and WSD research.....	64
Table 12. The detail of the Vietnamese-Korean parallel corpus.	65
Table 13. The results of the translation system.....	73

Chapter 1

Introduction

1.1. Motivation

Recently, the relationship between South Korea and Vietnam has improved significantly in various fields, from the economy to politics, to culture. According to the statistics of the Korean Statistical Information Service¹ in 2020, the number of people of Vietnamese holding Korean nationality ranks second among foreigners holding Korean nationality is shown below.

Table 1. Top 5 foreigners with Korean nationality.

Country	China	Vietnam	Thailand	Japan	Uzbekistan
Number of people	247,330	211,243	181,386	26,515	65,205

In the opposite direction, according to the statistics in February 2020, there are also about 170,000 Korean living and working in Vietnam. In recent years, Korea has always been one of the countries with the highest FDI in Vietnam. Even in 2019, South Korea is

¹ https://kosis.kr/statHtml/statHtml.do?orgId=111&tblId=DT_1B040A6

the leading country on this list with 7.92 billion dollars, according to the statistics 2019 of the ministry of planning and investment Vietnam as Figure 1.

From February 2021, Korean has become the preferred language for training in high schools in Vietnam. Besides, the demand for Vietnamese to learn Korean and for Koreans to learn Vietnamese is also increasing. However, translation machines like Google Translate or Papago have not yet provided Vietnamese-Korean translations as expected of users.

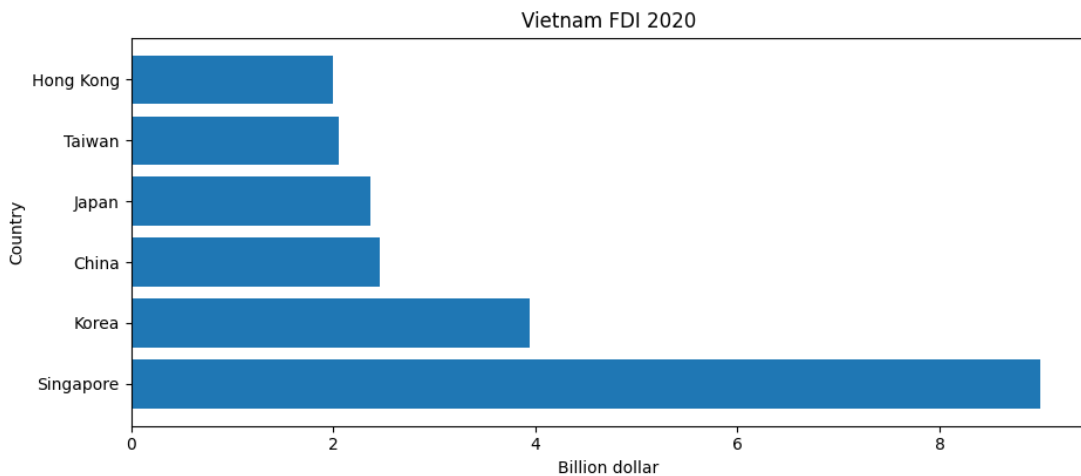


Figure 1. Vietnam FDI 2020

In addition, in recent years, research on MT is also very popular. However, these studies are mainly related to popular languages such as English, Chinese, German, and French. Besides, research on Vietnamese-Korean MT is still very limited for various reasons such as lack of data, nonoptimal translation model or the semantic complexity in both Vietnamese and Korean, etc.

For the above reasons, we believe that it is necessary to contribute to the Vietnamese-Korean MT research community.

1.2. Challenges and solutions

Firstly, to build a good MT system, first of all, it is necessary to have a large enough and good enough dataset to evaluate the quality of the MT with many different input sentences. Unfortunately, the datasets available are not large enough or mostly for commercial purposes. Our research had sifted through and collected data from a variety of sources. Then, we process the data through continuous steps to produce a dataset consisting of over 412 thousand Vietnamese-Korean sentence pairs. It is quite limited to build a real MT system but it is large enough and good enough for research compared to the other available datasets that are used in well-known studies for popular languages.

Secondly, in Vietnamese, a word can have different meanings depending on its role in the sentence. Many words have different meanings or diverse grammatical categories depending on their context. For example, the word "*bàn*" can be a verb (*to meet*) or it can be a noun (*table*) depending on the sentence. To overcome this problem, annotation which is used to add information to the data can be used. Specifically, POS tags thus indicate a word's syntactic function, allowing the MT systems easier to find words with the same meaning and function in the target language. Besides, in Vietnamese, the proper names (person, organization and place names) carry their meanings and if the MT translates these proper names according to their meaning into another language, the meaning of the target sentence becomes more complicated.

Thirdly, recent years have witnessed the birth and strong development of the state-of-the-art context-based embedding model called BERT. The BERT model generates the representations of the input sentence depending on the different contexts. The translator is

provided with clear input and this can also help the MT generate target sentences with more accuracy. To improve the quality of the Vietnamese-Korean MT system, we injected Vietnamese BERT into the NMT model and the translation results were significantly improved through evaluation by BLEU and TER scores.

1.3. Related works

1.3.1. Building machine translation systems

An MT system that can automatically translate text written in one language into another has been a dream from the beginning of artificial intelligence history. Recently, the research of MT has been energized by the emergence of statistical and deep-learning methodologies. Several open-source toolkits were released to benefit the MT community, such as Moses [1], cdec [2], and Phrasal [3], for statistical MT (SMT), and then OpenNMT [4], Nematus [5], and Tensor2Tensor [6] for deep-learning neural MT (NMT), which has, to a large extent, brought the dream to reality.

For SMT, Cho et al. [7] used an SMT phrase table to create a morpho-syntactic filter for solving the problem of the lexical gap. Depending on the lexical choice, they grouped component morphemes of Korean adjectives and Korean verbs. After they used the Moses toolkit for training data and translation from Korean to Vietnamese, they used the BLEU and TER score to evaluate the performance of their translation. The translation quality improved by approximately one point in BLEU scores, and that decreased by over two points in TER scores. Nguyen et al. [8] also indicated that the translation quality improved when analyzed morphologies were used to train a parallel corpus. They built a Vietnamese-Korean MT system using the Moses toolkit. Before training the MT model with 24K

sentence pairs, they analyzed the morphologies of the Korean text. The experimental results showed that analyzing the morphologies improved the MT quality by 3.34 BLEU points. Cho et al. [9] built a Vietnamese-Korean SMT MT based on the Moses toolkit. They selected data, including words, phrases, and sentences inside brackets, quotation marks, and parentheses, that were translated individually. This simple method was effective with the training data for a sentence containing brackets, quotation marks, or parentheses.

For NMT, NMT directly translates the source sentence to the target sentence by using a neural network [10,11]. A series of published toolkits have made it easier for researchers to access NMT such as OpenNMT [4], Sockeye [12], Fairseq [13], which have made it easier for researchers to access NMT. NMT research is mostly performed on many rich-resource language pairs [14–16], while research in low-resource languages is getting more attention [17–19]. The neural network models used in studies are also diverse. For example, Wu et al. have used Gap NMT (GNMT) [20], Gerhring et al. used ConvS2S (Convolutional sequence to sequence learning) [21], Huang et al. used a hybrid translation model [22]; and Vaswani et al. applied the transformer model [23].

In terms of data-level, most of the MT research has been conducted at the sentence-level, but there are various studies performed at the document-level. Voita et al. [24] proposed a context-aware NMT system based on the Transformer architecture. In their research, they used a previous sentence, a next sentence and a random sentence as the context, respectively. Most of their experiments showed the improved performance of NMT in terms of BLEU scores. Besides, Guo and Nguyen [25] incorporated BERT into document-level MT. They used BERT as a context encoder to accomplish document-level

contextual embedding. The experiment result showed that the performance of the MT system can be significantly improved at the document-level.

For Vietnamese-Korean MT, Nguyen et al. [26] used UWordMap to establish a Korean lexical-semantic network in which each sense of every polysemous word is connected to a sense-code that constitutes a network node. After tagging the Korean corpus using UWordMap, Nguyen performed Vietnamese-Korean translation experiments with OpenNMT [4].

1.3.2. Applying the annotation to the parallel corpus

Adding an annotation to the corpus makes the words in the sentence clearer, thereby improving translation quality. Ballier et al. [27] tried to improve the quality of their English-French and English-German MT systems by enriching the linguistic annotation of the input sentences. In this research, they used Combinatory Categorical Grammar (CCG) to add more information to the input sentences.

Modrzejewski et al. [14] incorporated Named Entity Annotation into their English to German and English to Chinese MT systems. Their NEs include location, person and organization and they are annotated with spaCy NER.

Nguyen et al. [28] applied MA and WSD to their Korean sentence to improve their Korean-English and Vietnamese-Korean MT systems. Their experimental results show that the Korean WSD system can significantly improve the translation quality of NMT in terms of BLEU, TER, and DLRATIO metrics.

In this study, we also apply MA and WS to Korean sentences, in addition, we tagged POS for Vietnamese sentences to improve the performance of Vietnamese-Korean MT systems.

1.3.3. Building and applying contextual embedding model

Bidirectional Encoder Representations for Transformers, abbreviated as BERT, is a pre-trained model introduced by Google [29]. BERT has greatly improved the quality of NLP tasks [29–31]. This state-of-the-art context-based embedding model comprises two tasks: masked language modelling (MLM) and next sentence prediction (NSP). There are different variants of BERT by language. ALBERT [32], RoBERTa [33], SpanBERT [34] are for English, FlauBERT [35] for French, PhoBERT [31] for Vietnamese; GottBERT [36] for German; MacBERT for Chinese [37]; and KR-BERT [38] for Korea.

ALBERT stands for A Lite BERT; this model uses cross-layer parameter sharing and factorized embedding layer parameterization techniques to decrease the number of parameters, thus reducing the training time. RoBERTa is also known as the Robustly Optimized BERT pre-training Approach and is one of the most popular variants of the BERT model. To improve the original BERT model, RoBERTa has implemented several changes in pre-training such as using byte-level byte pair encoding (BBPE) and dynamic masking instead of static masking in the MLM task. PhoBERT is a pre-trained BERT model for the Vietnamese language and this model was built based on the RoBERTa model. PhoBERT was trained on a huge Vietnamese corpus including a 50GB Vietnamese dataset.

Besides, the application of the contextual embedding model is also popular in downstream tasks. Devlin et al. [29] released BERT and they used BERT for some

downstream tasks such as question answering, NER and sentence classification. Nguyen et al. [31] used Vietnamese contextual embedding for POS tagging, NER, and dependency parsing. Zhu et al. [30] applied English contextual embedding to translate English sentences to some other languages such as Deutsch, Estonian, Chinese, and France. Clinchant et al. [39] also used BERT for NMT and their experiments are performed in English-German and English-Russian parallel corpora.

In this research, we apply the Transformer model for Vietnamese-Korean MT, in which the MT model combines the pre-trained contextual embedding model Vietnamese BERT (PhoBERT) and NMT. Before being included in the MT systems, the Vietnamese-Korean bilingual corpus is prepped in a series of pre-treatment steps to improve the quality of the MT. In addition, we tagged POS to Vietnamese sentences. This POS tagging was also developed based on a pre-trained contextual embedding model.

1.3.4. Building parallel corpus for machine translation

Besides the MT methodologies, the parallel corpus is another very crucial component of MT systems. An excellent MT system needs a large parallel corpus to ensure high accuracy during translation model training. The manually compiled parallel corpora, which consume a lot of time and budget to establish, are commonly used for economic purposes. There are some automatically collected parallel corpora available for research [40–42] but only for popular languages. Since parallel corpora are essential resources for not only MT but, also, a variety of natural language processing duties (i.e., semantic resources production and multilingual information extraction), many groups have built parallel

corpora, which are in use today. Some of which include Europarl² [40], consisting of English and 20 European languages; JRC-Acquis³[41], containing 20 official European Union languages; and the United Nations corpus⁴ [42], with Arabic, French, Spanish, French, Russian, Chinese, and English. These corpora are freely provided for research purposes.

For the Korean-English parallel corpus, several research groups have generated parallel corpora to train their MT systems. Lee et al. [43] built a corpus with 46,185 sentence pairs by manually collecting texts from travel guidebooks. Hong et al. [44] collected about 300,000 sentence pairs from bilingual news broadcasting websites. Chung and Gildea [10] also collected the Korean-English alignment sentences from websites and got approximately 60,000 sentence pairs. These collected parallel corpora are not public, and their sizes are inefficient to train high-quality MT systems. There are a few available Korean-English parallel corpora. The Sejong parallel corpora [11] contains about 60,000 Korean-English sentence pairs collected from diverse sources such as novels, transcribed speech documents, and government documents. KAIST corpus⁵ also contains 60,000 Korean-English sentence pairs, but the collection resources are not mentioned. News Commentary corpus⁶ [45], which was crawled from the CNN and Yahoo websites contains approximately 97,000 Korean-English sentence pairs. These parallel corpora are publicly available; however, their sizes are too small to train MT systems. In another study, the open

² <http://www.statmt.org/europarl>

³ https://wt-public.emm4u.eu/Acquis/JRC-Acquis.3.0/doc/README_Acquis-Communautaire-corpus_JRC.html

⁴ <http://uncorpora.org>

⁵ http://semanticweb.kaist.ac.kr/home/index.php/KAIST_Corpus

⁶ <https://github.com/jungyeul/korean-parallel-corpora>

parallel corpora OPUS [46] was proposed with a huge number of sentence pairs for multiple languages, including Korean-English pairs. The Korean-English dataset was obtained from technical documents (i.e., Ubuntu and GNOME) and movie subtitles, but the number of sentence pairs of this dataset is limited and contains numerous noises.

For the Vietnamese-Korean language pair, Nanyang Technological University NTU-MC [12] introduced multilingual corpora including Korean and Vietnamese sentences. Nguyen et al. [8] built a Vietnamese-Korean parallel corpus for their own MT systems. However, both datasets are extremely small, with only 15,000 and 24,000 sentence pairs, respectively. The computational linguistics centre (University of Science, Ho Chi Minh City) provided a Vietnamese-Korean bilingual corpus with 500,000 sentence pairs for commercial purposes [47].

In this research, we propose parallel corpora called UPC, consisting of a large parallel open corpus for training Vietnamese-Korean MT models. “U” stands for Ulsan (i.e., University of Ulsan, our affiliation), and PC represents parallel corpora. The data is collected for many different audiences, and the topics focus on issues related to everyday life, such as economics, education, religion, etc. The sources used to extract these parallel corpora are mainly articles from multilingual magazines and example sentences from online dictionaries. Up to 412 thousand sentence pairs in Vietnamese-Korean were obtained. These datasets are large enough to train quality MT systems and are available for download at <https://github.com/haivv/UPC>.

1.4. Dissertation Outline

This dissertation is composed of seven chapters. The first chapter is about the motivation, challenges and solutions of the dissertation. Besides, in this chapter, I mentioned some studies that are related to this dissertation. The remaining dissertation is organized as follows:

Chapter 2 presents the main background knowledge related to the dissertation including word embedding, NMT, Transformer model, and linguistic annotation.

Chapter 3 illustrates an overview of the pre-trained BERT model and some pre-trained BERT variants including the Vietnamese BERT model.

Chapter 4 mentions the Vietnamese POS task and some of the approaches to tag POS for Vietnamese sentences.

Chapter 5 describes the parallel corpus for Vietnamese-Korean MT and the process in which this dataset was generated.

Chapter 6 shows in detail the process of performing and evaluating the accuracy of experiments to demonstrate the benefit of POS tagging and the contextual embedding in Vietnamese-Korean MT.

Chapter 7 demonstrates the achievements of this dissertation and shows some other research that we plan to perform in the future.

Chapter 2

Backgrounds

2.1. Word embedding

2.1.1. General word embedding

A word embedding is a learned representation for text where words that have the same meaning have a similar representation. This is an important step in NLP because the computer cannot understand natural language. For the computer to read the word, it is necessary to represent the text as vectors in the real number space. Word embedding is a language modelling and feature learning technique to map words into vectors of real numbers using neural networks, probabilistic models, or dimension reduction on the word co-occurrence matrix. Word embedding methods can be divided into Classic Word Embedding Method and Neural Embedding (Text Vectorization by Neural Network Method).

2.1.2. Classical word embedding

Bag of Words (BoW)

This is the most commonly used traditional vector representation. Each word or n-gram of words will be described as a vector with dimensions equal to the number of words in the dictionary. At the position corresponding to the position of that word in the bag of words, the element in that vector will be marked as 1. The remaining positions will be marked as 0. The BoW method is often used in text classification problems. In which, the frequency of each word or n-gram of words will be considered as a feature in the classification text.

The disadvantage of the BoW is that we cannot determine the true meaning of each word and the words associated with them. In the BoW method, the same words are weighted equally. This method does not consider the frequency of occurrence of words or word scenes.

Term Frequency–Inverse Document Frequency (TF-IDF)

This is a statistical method, which reflects the importance of each word or n-gram of words to the text on the entire input document. TF-IDF represents the weight of each word according to the text context. The value of TF-IDF proportional increases to the number of occurrences of the word in the text and the number of documents containing that word in the entire document set. This method makes TF-IDF more categorical than the BOW method. However, the TF-IDF method can only show the weights of different words in the text, but it cannot represent the meaning of words.

2.1.3. Neural word embedding

Some common types of word embedding are based on neural networks including Word2Vec, Glove, ELMo and BERT.

Word2Vec

Word2Vec was introduced in 2013 by Google and this embedding method is contextual independent word embeddings. This outputs only one embedding for each word that is including all the different senses of the word into one vector. Word2Vec is word-based model and this takes words as input and generates word embeddings.

Word2vec is a prediction-based embedding algorithm. This model represents the vector of words through the surrounding context words to improve the ability to predict the meaning of words. There are two ways to build a Word2vec model to represent the dispersion of words in the space of vector:

- **Continuous Bag-of-Words model (CBOW):** The CBOW predicts words based on the context. This algorithm takes the context of the surrounding word as the input and tries to predict the word corresponding to the context.
- **Continuous Skip-Gram model:** This method is similar to the CBOW model, but it predicts the surrounding word of the current word.

The CBOW algorithm takes less time to train than the Continuous Skip-gram model.

The architecture of Word2Vec is demonstrated in figure 2:

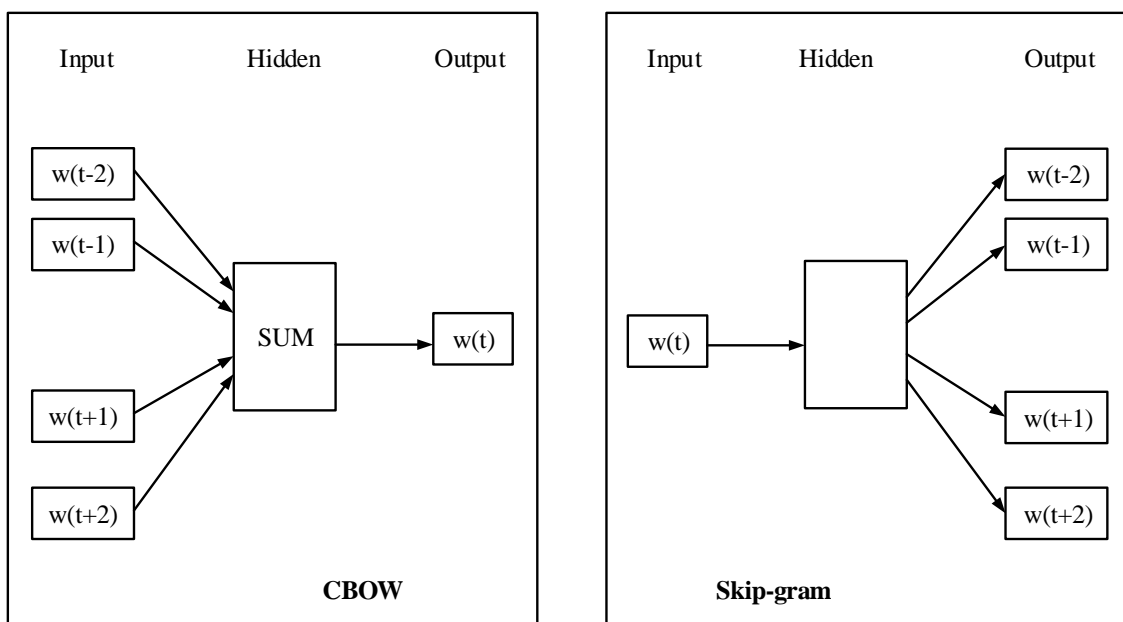


Figure 2. Word2Vec architecture.

In the preceding figure, $w(t)$ is target word; $w(t-2)$, $w(t-1)$, $w(t+1)$, and $w(t+2)$ are surrounding words of the target word.

Global Vectors for Word Representation

Global Vectors for Word Representation (GloVe) was created in 2013 by a team at Stanford University [48]. Similar to the word2vec method, GloVe is also the contextual independent word embeddings and word-based models (word-count base model).

Word2vec uses a context window to generate training sets for neural networks and GloVe uses it to generate the Co-occurrence Matrix. The GloVe is more "global" than Word2vec because GloVe calculates word probabilities based on the entire data set. The algorithms of the Word2Vec method only consider the context surrounding the target word, but this does not consider the full-text context.

Embeddings from Language Model

Embeddings from Language Model (ELMo) is a deep contextualized word representation using Char-based CNN or Bidirectional LSTM model. With the same word, ELMo uses a bidirectional language model (biLM) to capture context-dependent aspects of word meaning and generate; this model learns both words (e.g., syntax and semantics) and linguistic context. Instead of using a fixed embedding for each word (like the GloVe model), ELMo looks at the entire sentence before assigning each word with its embedding. Elmo uses bi-directional LSTM in training; therefore, its language model can understand the next word as well as the previous word in the sentence. The ELMo contextualized embedding is generated through grouping together the hidden states and initial embedding.

2.2. Neural machine translation

NMT integrates neural language models and a traditional statistical MT system into one system. In the 1990s, neural networks has been used to train a translation model. However, because of hardware limitations, NMT was abandoned for almost two decades. Hardware advances have allowed the performance of NMT to be improved significantly. In 2014, Sutskever et al. [49] and Cho et al. [50] proposed a sequence-to-sequence framework for NMT models. After using an RNN to encode the input sequence into a fixed-length vector representation, their method used another RNN to decode the target sequence from that vector.

2.2.1. Recurrent neural network

RNN is an artificial neural network designed for processing data types in the form of sequential sequences. In an RNN, the hidden state at each time step is calculated based on the input data at the corresponding time step and the information obtained from the previous time step. Figure 3 shows the recurrent neuron, in which one neuron receives an input, generates an output and then sends the output back to itself. At time step t , each neuron N receives both input x_t and the output from the previous step y_{t-1} .

The output y_t is computed by

$$y_t = f(x_t^T W_x + y_{t-1}^T W_y + b), \quad (1)$$

where W_x and W_y are two matrix weights for the input x_t and the output of the previous time step y_{t-1} , and f is the active function; and b is a vector of n neurons containing each neuron's bias term.

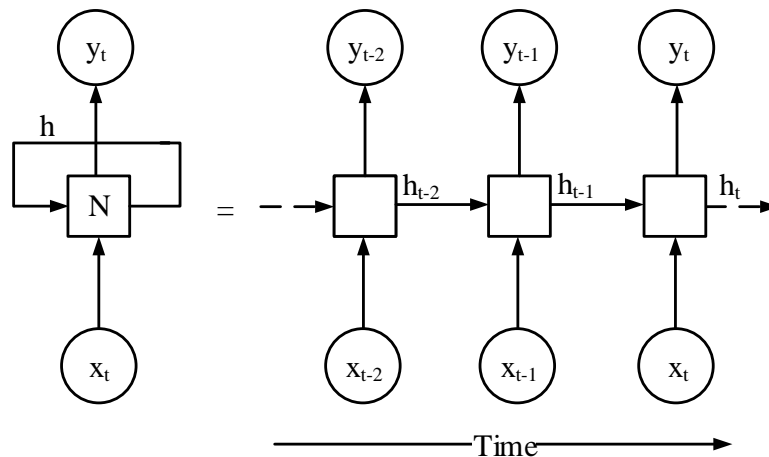


Figure 3. Structure of a recurrent neuron.

The h_t shortcut for the hidden state at state t is calculated based on the current input and the hidden state at the previous time step as:

$$h_t = f(h_{t-1}, x_t). \quad (2)$$

2.2.1. Encoder

The encoder is a bi-directional RNN, including a forwarding RNN and a backward RNN, as shown in Figure 4. The first task of the encoder is to transform the input sentence into a sequence of word vectors (word embedding). Then, the encoder processes those vectors using a bi-directional RNN.

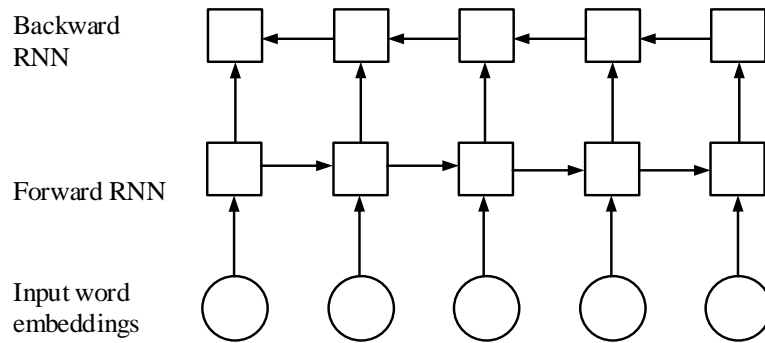


Figure 4. Encoder structure.

Mathematically, the source sentence is a sequence of the form $x = (x_1, x_2, \dots, x_n)$, where n is the length of the sentence. In RNNs, the word by word of the sentence is fed to the network. First, the word x_1 is passed as input, next the word x_2 is passed, and so on. Therefore, the network understands the sentence completely. The hidden states of the forwarding RNN (\vec{h}) and backward RNN (\overleftarrow{h}) are calculated by

$$\vec{h}_i = f(\vec{h}_{i-1}, \vec{E}x_i) \quad (3)$$

$$\overleftarrow{h}_i = f(\overleftarrow{h}_{i+1}, \vec{E}x_i). \quad (4)$$

In equations (9), and (10), f is a tanh function (a typical feed-forward neural network layer) - $f(x) = \tanh(Ax + B)$. \bar{E} is a word-embedding matrix of the source language.

The source annotations (h_1, h_2, \dots, h_n) are a concatenation of the forward and backward hidden states as:

$$h_i = (\overrightarrow{h_i}, \overleftarrow{h_i}) \quad (5)$$

2.2.2. Decoder

The decoder, shown in Figure 5, is a forwarding RNN used to predict the target sentence $y = (y_1, y_2, \dots, y_m)$, where m is the length of the sentence. A sequence of the hidden state s_j is computed from the previously hidden state s_{j-1} , the previous target word Ey_{j-1} , and the input context vector c_j using the following equation:

$$s_j = f(s_{j-1}, Ey_{j-1}, c_j). \quad (6)$$

The prediction vector p_j is based on the input context c_j , the decoder hidden state s_{j-1} , and the embedding of the previous output word Ey_{j-1} .

$$p_j = \text{softmax}(W(Us_{j-1} + Ey_{j-1} + Cc_j)) \quad (7)$$

In equations (12), and (13), E is the word-embedding matrix of the target language, and W , U , and C are weight matrices.

The output word y_j will be selected to have the highest value in p_j before using its embedded Ep_j for the following stage.

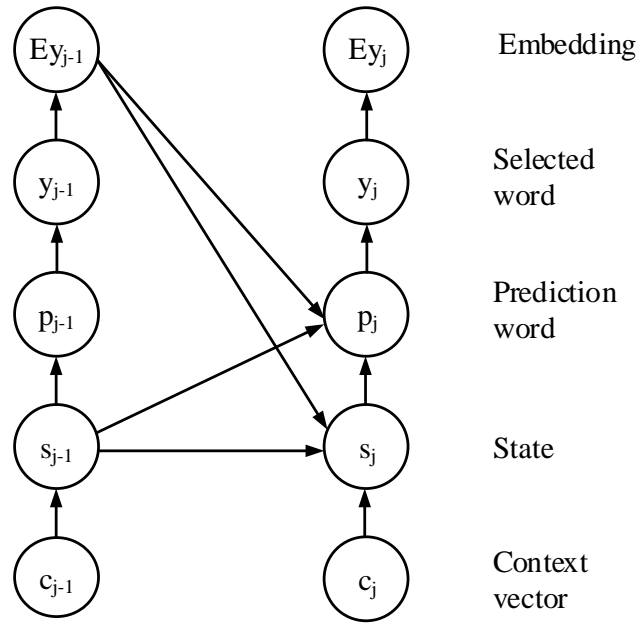


Figure 5. Decoder structure

2.2.3. Attention Mechanism

In 2015, to increase the accuracy of information capture in Encoder-Decoder models, the Attention mechanism was introduced. The main idea of the Attention mechanism is to create a shortcut that directly associates each word in the target language with the corresponding word of interest in the source language (Bahdanau et al., 2015 & Luong et al., 2015).

Attention is a mechanism that helps the model focus on the corresponding important parts of the information by creating an alignment model that computes alignment scores to re-align the relevance of the hidden states at the encoder to the decoder outputs at the corresponding time step. With NMT, the attention mechanism helps the model understand the relevance of the input word and the next predictor word at the decoder.

The attention mechanism is a combination of input word representation $(\vec{h}_t, \overleftarrow{h}_t)$ (which is generated in the encoder step) and the context state c_j (produced from the previously hidden state of the decoder s_{j-1}). The structure of the attention mechanism is visualized in Figure 6.

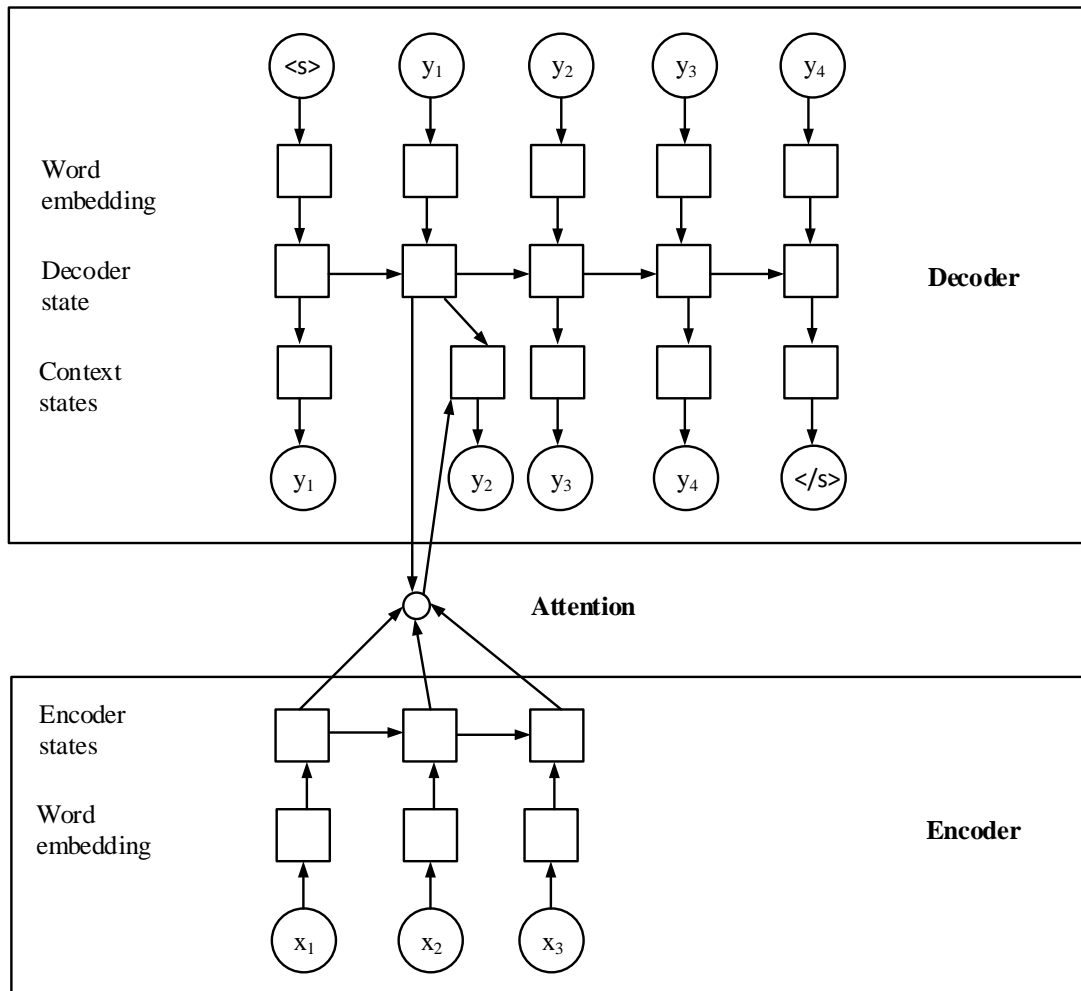


Figure 6. Attention mechanism structure.

The attention value is computed as:

$$e_{ij} = \frac{\exp(a(s_{j-1}, h_i))}{\sum_k \exp(a(s_{j-1}, h_k))}, \quad (8)$$

where \mathbf{a} is the association between the decoder state and each input word. This association is calculated by:

$$a(s_{j-1}, h_i) = W^{aT} s_{j-1} + U^{aT} h_i + b^a, \quad (9)$$

where W^a and U^a are weight vectors, and b^a is the bias value.

At that time, the value of the context vector c_j is computed by the following equation:

$$c_j = \sum_i e_{ij} h_i. \quad (10)$$

$\langle s \rangle$ and $\langle /s \rangle$ signals the start and end of the decoder.

The NMT method has developed strongly and it has shown superior performance compared to the previous methods (RBMT and SMT). A brief comparison of NMT and RBMT and SMT is described in Table 2.

Table 2. A brief comparison of NMT, RBMT and SMT.

Name Feature	Rule-Based Machine Translation	Statistical Machine Translation	Neural Machine Translation
Abbreviation	RBMT	SMT	NMT
Overview	Based on dictionary and knowledge of grammar	Based on a statistical model. Parameters are acquired from the analysis of bilingual text corpus.	Based on the neural network. Each neurone in the network is a mathematical function that processes data.
Advantage	<ul style="list-style-type: none"> • Fast (training and translation) • No bilingual texts are required • Total control. 	<ul style="list-style-type: none"> • Translations are relatively natural sentences (better than RBMT) 	Can generate natural target sentences
Disadvantage	Translation is lack of fluency	Translation is lack of fluency	Takes a lot of time to train

2.3. Transformer model

Transformer [23,51] is a popular deep learning model that has regularly substituted RNN or LSTM for NLP tasks. In the language translation task, the transformer model contains an encoder-decoder architecture. For example, to convert a sentence from Vietnamese to Korean, the encoder learns the representation of the given Vietnamese sentence and feeds this representation to the decoder. Then, the decoder will generate the corresponding sentence in the Korean language.

2.3.1. The encoder of the transformer

A transformer contains a stack of L number of encoders. Each encoder block involves two sublayers including a multi-head attention mechanism and a feed-forward network as shown in Figure 7. Each encoder sends its output to the next encoder and the last encoder generates a matching representation of the corresponding input sentence.

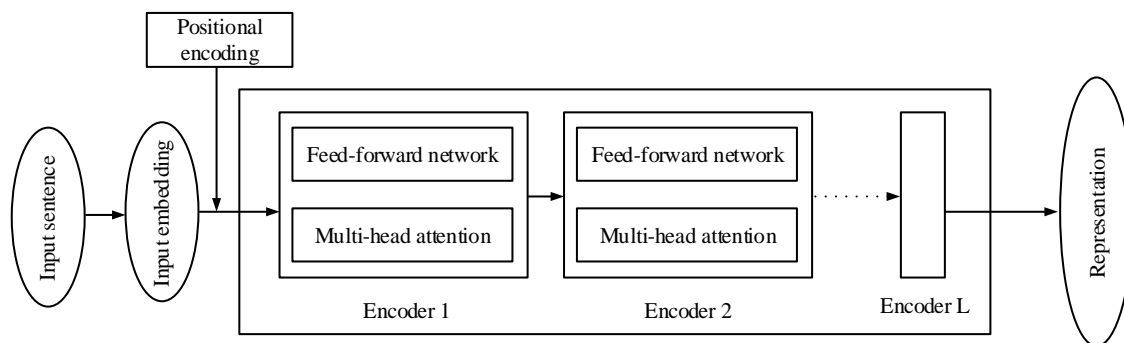


Figure 7. A simple demonstration of the encoder.

In RNNs, the word by word of the input sentence is fed into the network. But, in the transformer model, all words of the input sentence are fed into the network. This method helps in learning the long-term dependency and this also helps in decreasing the training time. Firstly, the input sentence is converted to input embedding. This embedding matrix is fed to the first encoder block (encoder 1). In addition, all words in the input sentence are fed into the transformer network, thus causing the transformer cannot understand the order of words in the sentence. To understand the sentence, a positional encoding is added to provide information about the position of each word before feeding it into the transformer model. The positional encoding matrix is added to the input embedding matrix, and then all of them are fed as input to the encoder of the transformer.

Encoder 1 gets the input and puts it into a multi-head attention layer, which generates the attention matrix as output. Then, the attention matrix is sent to the feed-forward network as the input of this layer, and this returns the encoder representation as output. In the next step, the encoder representation of encoder 1 is fed to encoder 2 as input and encoder 2 performs similar steps as encoder 1. This process ends when the final encoder blocks (encoder L) are computed, and the representation of the given input sentence is generated.

Mathematically, suppose n is the number of single attention matrix Z and the multi-head attention mechanism is the combination of all the attention matrices (attention heads) multiplied by a new weight matrix, W_0 , as follows:

$$\text{multi-head attention} = \text{Concatenate}(Z_1, Z_2, \dots, Z_n)W_0 \quad (11)$$

Each attention matrix Z_x can be computed as:

$$Z_x = \text{softmax}\left(\frac{Q_x K_x^T}{\sqrt{d_k}}\right) V_x \quad (12)$$

where Q_x , K_x , and V_x are respectively the query, key, and value matrix of input word x ; d_k represents the dimension of the key vector. Instead of computing a single attention head, multiple attention heads will be calculated. As equation (2), a single attention head is computed by the self-attention mechanism according to the following. First of all, the dot product between the query matrix Q and the key matrix K^T is computed. Then, the value of QK^T is divided by the square root of the dimension of the key vector. Next, the values

obtained in the previous step are fed into the soft-max function to normalize the scores and achieve the score matrix. Finally, the attention matrix Z is computed by multiplying the score matrix by the value matrix V .

The feed-forward network in the encoder of the transformer contains two dense layers with Rectified Linear Unit (ReLU) activation function. In the feed-forward network, the parameters are different from the encoder blocks, but this is the same in the different positions of the sentence.

Besides feed-forward network and multi-head attention, the add and norm is also an important component in the encoder. The add and norm component is used to connect the input and output of each feed-forward network and multi-head attention layer. This is used in order to avoid the values in each layer from changing heavily, therefore the transformer can train faster. The demonstration of an encoder x with the add and norm component is shown as follows:

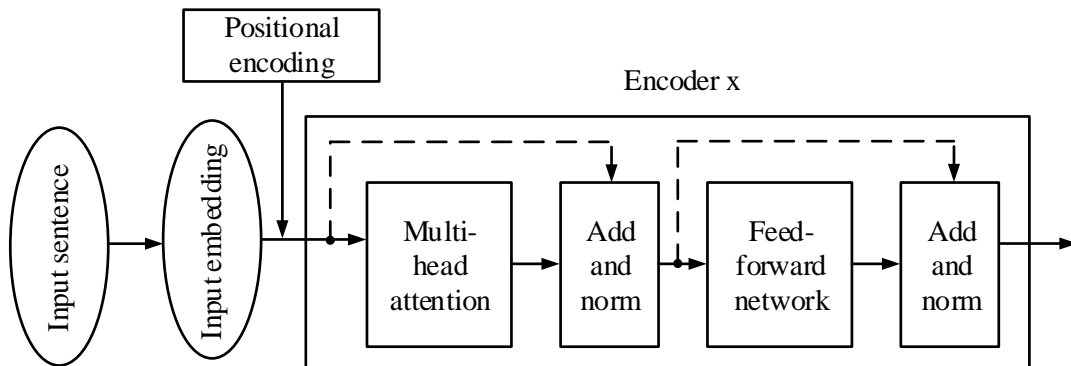


Figure 8. Encoder x with the add and norm component.

The encoder representation is obtained from the final encoder and this is fed into the decoder of the transformer. The decoder takes the encoder representation to generate the target sentence.

2.3.2. The decoder of the transformer

The decoder of the transformer takes the embedding representation of the encoder as an input and returns the target sentence as output. Similar to the encoder, the decoder also contains a stack of N sub-decoders. A decoder has two inputs including the output of the previous decoder and the representation of the encoder, as in the following figure:

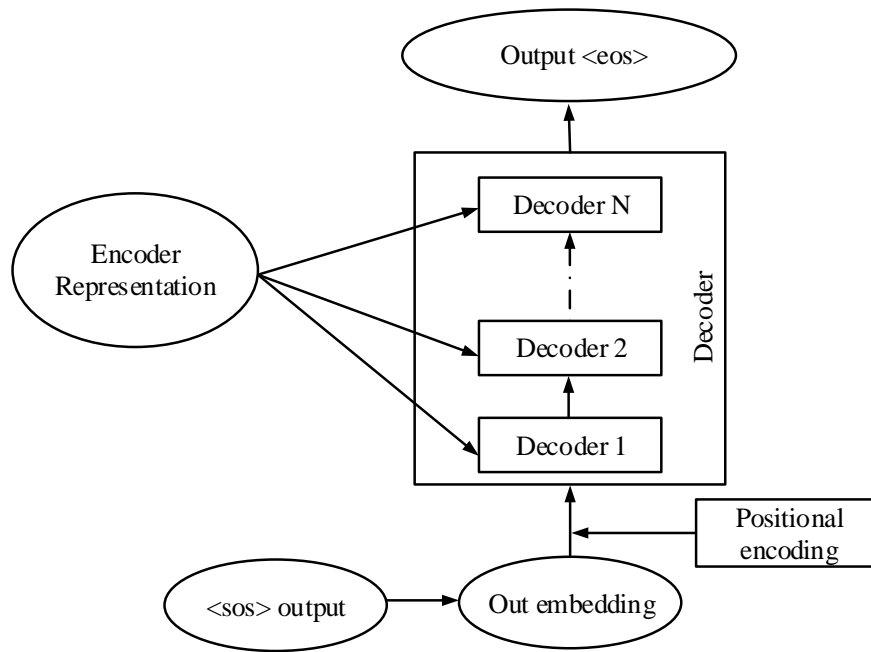


Figure 9. A simple demonstration of the decoder.

During training processing, the *< sos >* and *< eos >* tokens respectively indicate the beginning and the ending of the sentence. As in the preceding figure, the input of the decoder is converted into an embedding matrix (output embedding matrix); and this is also

added with a positional encoding. Then, this input and the representation of the encoder are fed into the decoder to generate the output of the decoder.

Suppose the output of decoder is a sentence of the form $y = (y_1, y_2, \dots, y_m)$; m is the length of sentence. At first time step $t = 0$, the *< sos > output* is *< sos >* and the *output < eos >* is y_1 ; at time step $t = 1$, the *< sos > output* is *< sos >* y_1 and the *output < eos >* is $y_1 y_2$; at the last time step when full output is generated, the token *< eos >* is created and the output of the decoder is $y_1 y_2 \dots y_n \text{ < eos >}$.

There are three sublayers on each single decoder block including masked multi-head attention, multi-head attention and feed-forward network. In addition, the output of each sublayer is connected to an add and norm component; similar to the encoder component, the positional encoder layer is also added to the decoder component as shown in the following diagram:

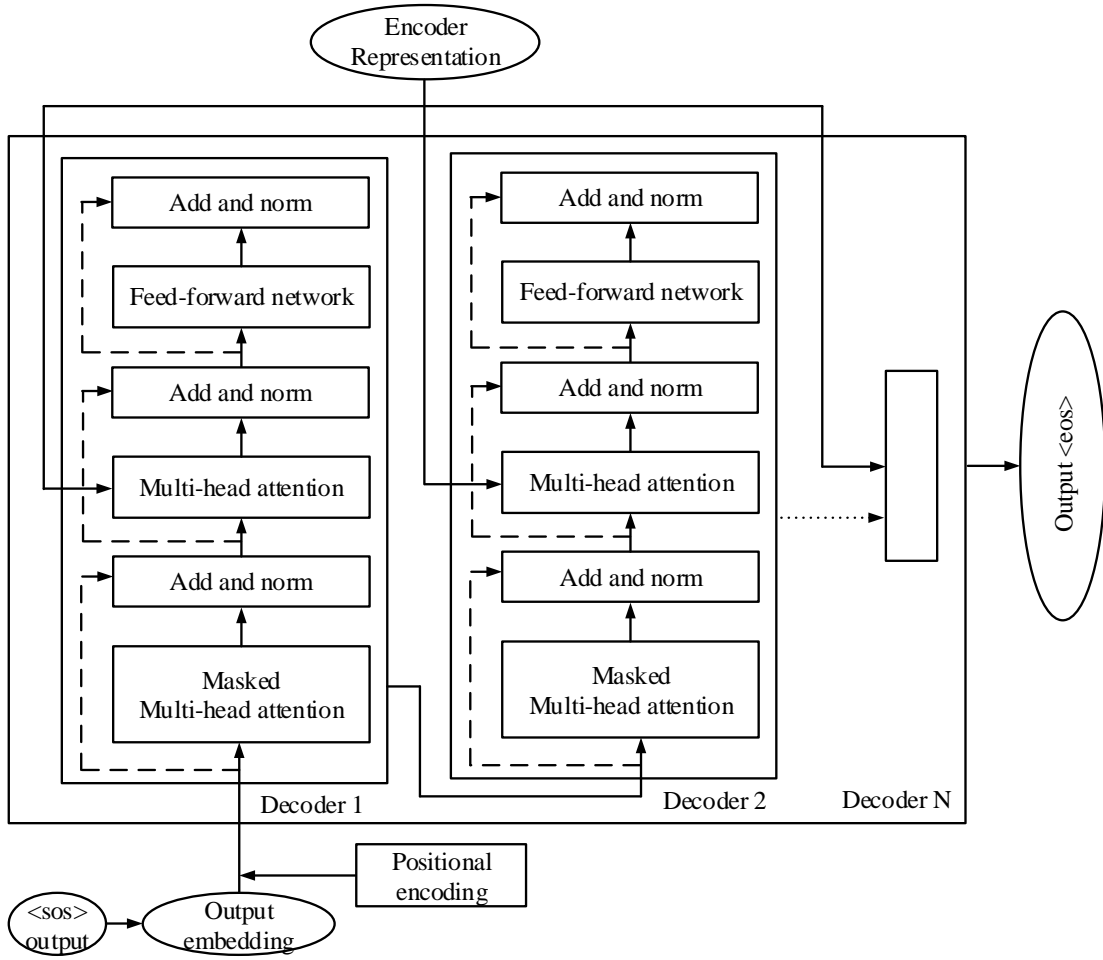


Figure 10. A simple demonstration of the decoder.

The masked multi-head attention is the first layer in the decoder; this works similar to the multi-head attention mechanism in the encoder. For head i , the attention matrix Z_i is computed as follows:

$$Z_i = \text{softmax} \left(\frac{Q_i K_i^T}{\sqrt{d_k}} \right) V_i \quad (13)$$

where Q_i, K_i, V_i are the query, key, and value of the input matrix; d_k is the dimension of the key vector. The final attention matrix M is generated by concatenation of all attention matrices and a new weight matrix W_0 as the following equation:

$$M = \text{Concatenate}(Z_1, Z_2, \dots, Z_h)W_0 \quad (14)$$

The result of the final attention in the masked multi-head attention layer is fed to the multi-head attention layer. As in Figure 11, each multi-head attention layer has two inputs including the representation of the encoder (R) and the output of the masked multi-head attention (M). The key and the value of this layer are calculated similarly to the method used in the encoder layer but with a small difference. The weight matrix of the encoder representation (R) is multiplied by the weight matrices of the query and the key, respectively.

The demonstration of an encoder and a decoder is described in Figure 11 (x and y are the numbers of encoder and decoder layers, respectively).

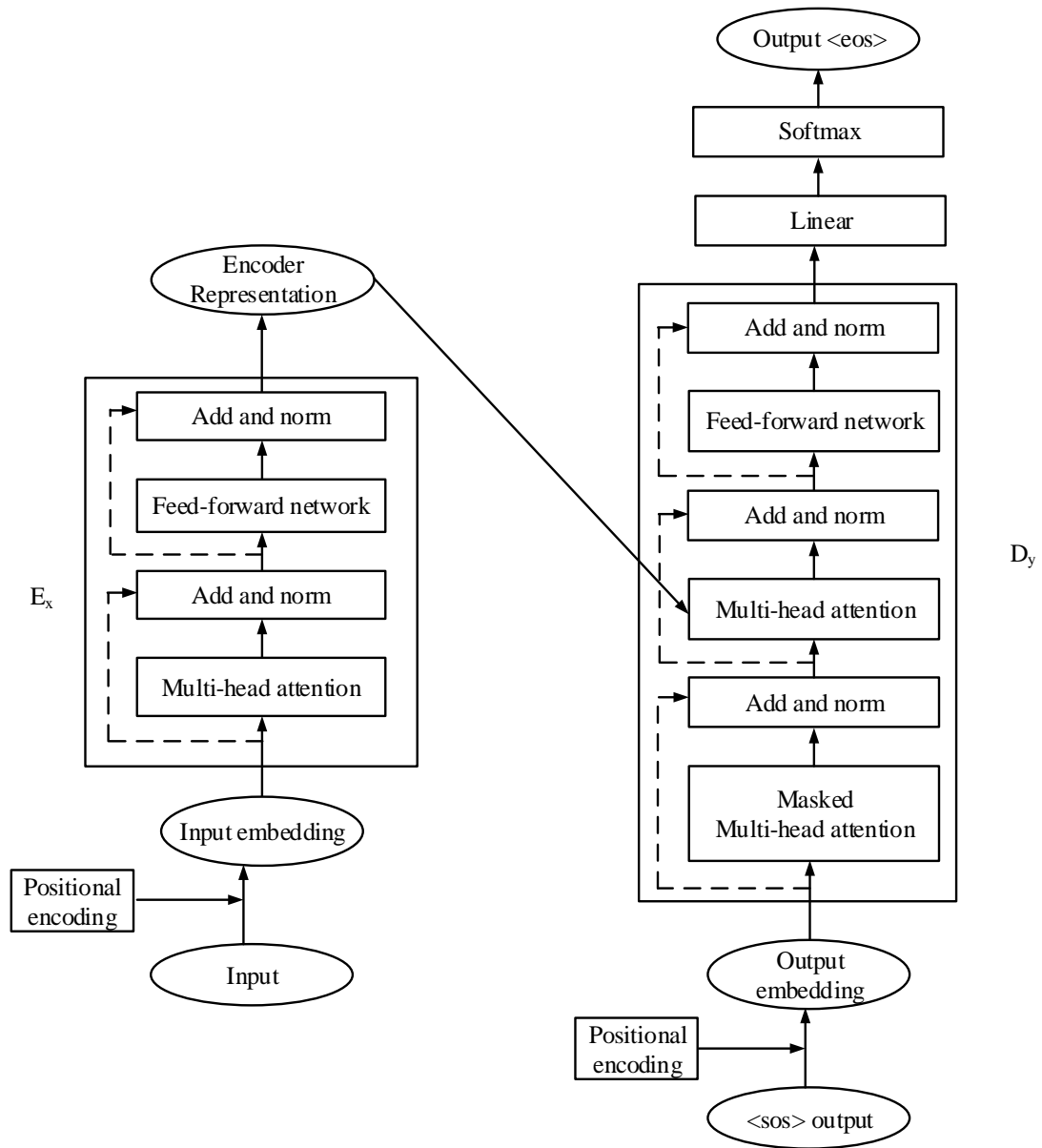


Figure 11. The combination of encoder and decoder in transformer model.

2.4. Linguistic annotation

Some recent studies have shown the benefit of linguistic annotation for machine translation by combining linguistic annotation with input sentences. The typical linguistic features such as POS, morphological properties of words, WSD, syntactic dependencies,

et al. can be added to input sentences. In this research, we present three types of annotation including POS, MA and WSD, which we have applied to boost the quality of our MT system.

2.4.1. Part of Speech

Part of Speech is abbreviated as POS. In corpus linguistics, POS is the process of marking a word in the text as corresponding to a certain type of word based on its definition and grammatical context. Currently, there are many different POS tagging techniques such as:

- **Lexical Based Methods:** label each word POS according to the word form that occurs most frequently in the data set.
- **Rule-Based Methods:** Label the POS based on a defined rule. For example, in English, words that end in "*ed*" or "*ing*" are often assigned a verb. Rule-Based Methods can be combined with Lexical Based Methods to label words that are in the train set but not in the test set.
- **Probabilistic Methods:** this method labels the POS based on the probability of a particular sequence of labels. Conditional Random Fields (CRFs) and Hidden Markov Models (HMMs) algorithms are the two most popular algorithms of this method.
- **Deep Learning Methods:** using neural networks to label POS.

2.4.2. Named Entity Recognition

Named Entity Recognition or NER is one of the basic tasks in Natural Language Processing. The main role of this task is to recognize phrases in the text and classify them into predefined groups such as names of people, organizations, places, times, product types, brands, and so on. The results of the NER task can be handled for more complex problems such as Chatbot, Question Answering, or Searching.

The problem of entity recognition has also been posted for a long time, so there are many solutions such as Rule-based, Statistical Learning, and Deep Learning.

2.4.3. Morphological Analysis

Morphology is the study of the structures, shapes or forms of animate and inanimate objects such as crystals, words, or organisms. In linguistics, morphology analysis (Morph. Ana.) is the study of the internal structure of words [52]. This identifies analysis and describes the structure of words and parts of words such as root words, prefixes, and suffixes.

There are three principal approaches to morphology as follows:

- **Morpheme-based morphology:** A morpheme is the smallest meaningful lexical item in a language. In this method, word forms are analyzed as arrangements of morphemes. For example, the word “*independently*” is parsed into the morphemes “*in*”, “*depend*”, “*ent*”, and “*ly*”. Morpheme-base morphology takes advantage of an item-end-arrangement approach.

- **Lexeme-based morphology:** Instead of analyzing a word form as a set of morphemes arranged in sequence, Lexeme-based morphology applies rules to alter a word form to produce a new one. This normally makes use of an item-and-process approach.
- **Word-based morphology:** this normally takes advantage of a word-and-paradigm approach. Paradigm standards for what constitutes legitimate contributions to a field and this is a distinct set of concepts including theories, research methods, and postulates. Instead of using rules to combine morphemes into word forms or to generate word forms from the root word, word-based morphology expresses generalizations that hold between the forms of inflectional paradigms.

2.4.4. Word Sense Disambiguation

Word sense disambiguation (WSD) is the issue of determining which "sense" (meaning) of a word in a particular context. WSD is a natural classification problem with a given word and its possible senses. For example, a word can have several senses as Table 3 and the task of WSD is to determine the meaning of a given word depending on its context (neighbouring words).

There are four regular approaches to WSD including:

- **Dictionary and knowledge-based methods:** This method is based on dictionaries, thesauri, and lexical knowledge bases, without using any corpus evidence.

Table 3. Some examples of word senses

Word	POS	Sense No.	Senses
hit ⁷	Verb	01	to move your hand or an object onto the surface of something
		02	to succeed in reaching or achieving something:
		03	to arrive at a place, position, or state
	Noun	04	a thing or person that is very popular or successful
		05	the act of hitting someone or something, or an occasion when someone or something is hit
foot ⁸	Noun	01	the part of the body at the bottom of the leg on which a person or animal stands
		02	the bottom or lower end of a space or object
		03	a unit of measurement

- **Supervised methods:** This method uses sense-annotated corpora to train the model. However, a sense-tagged corpus for training is expensive to create.
- **Semi-supervised or minimally-supervised methods:** This approach is based on a small annotated corpus as seed data in the bootstrapping process. The seed data is used to train an original classifier based on the supervised method. Then, this classifier is used on the untagged portion of the corpus to extract a larger training set.

⁷ <https://dictionary.cambridge.org/dictionary/english/hit>

⁸ <https://dictionary.cambridge.org/dictionary/english/foot>

- **Unsupervised methods:** This approach uses raw unannotated corpora to train the model and this method is the greatest challenge for WSD studies.

2.5. Summary

This chapter describes the background knowledge that is applied in this thesis including word embedding, NMT, transformer model and linguistic annotation.

This chapter is started by looking at some word embedding methods consisting of classical word embedding and neural word embedding. Currently, some neural word embedding methods BERT is a hot topic and it is applied in many different fields such as text classification, question-answering, MT, and more.

In the next part of the chapter, the NMT model with attention mechanism is mentioned. The NMT is a sequence-to-sequence encoder-decoder model with attention. The encoder is a bi-directional RNN, this used a context-free model (word2vec) to convert the input sentence into an input vector; then by using a bi-directional RNN, the encoder generates the corresponding representation of the input sentence. The decoder is also an RNN; this takes the representation of the input sentence, and the previous hidden state, the output word prediction, and engenders a new output word prediction. The attention mechanism creates a shortcut that directly links the words in the target language with the corresponding words.

Then, the structure of the transformer model is described. This is also an encoder-decoder with attention architecture. The encoder contains some sublayers such as multi-head attention and a feed-forward network. The decoder contains three sublayers including

the masked multi-head attention, encoder-decoder attention, and the feedforward network. The self-attention mechanism relates a word to all the words in the sentence to better understand the word.

Section 2.4 mentioned some linguistic annotations including POS tagging, NER, morphological analysis, and WSD. Linguistic annotations add meaning to words so they can contribute to improving the quality of MT.

Chapter 3

Pre-trained BERT model and Vietnamese Pre-trained BERT variant

Bi-direction Encoder Representation from Transformer (BERT) is the state-of-the-art text embedding model developed by Google. BERT is a pre-trained model and it has made a breakthrough in the field of natural language processing.

3.1. Architecture of the BERT model

BERT model is built based on the encoder of the Transformer architecture. Unlike other free-based embedding models such as word2vec, BERT is a context-based embedding model. Therefore, a word in different contexts will be created by BERT with dynamic vector embedding. In the BERT model, the input sentence is fed to the encoder. The encoder uses a multi-head attention layer to understand the context of each word and returns the representation of each given word based on context, as in Figure 12.

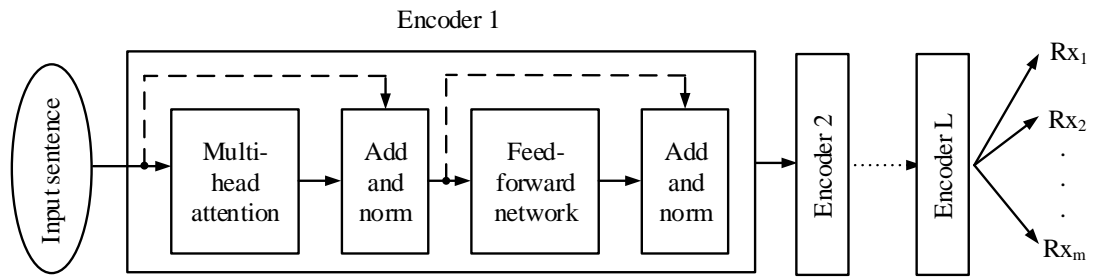


Figure 12. A simple demonstration of the BERT model. L and m are the number of the encoder and the length of the input sentence, respectively.

Based on the number of encoder layers (L), the attention-head (A), and the hidden unit (H), BERT is divided into two main types, BERT-base and BERT-large. In the BERT-base model, $L = 12$, $A = 12$, $H = 768$, and there are 110 million parameters. In the BERT-large model, $L = 24$, $A = 16$, $H = 1024$, and there are 340 million parameters. For example, the BERT-base generates the representation of the input sentence “I am studying at University of Ulsan” like Figure 13.

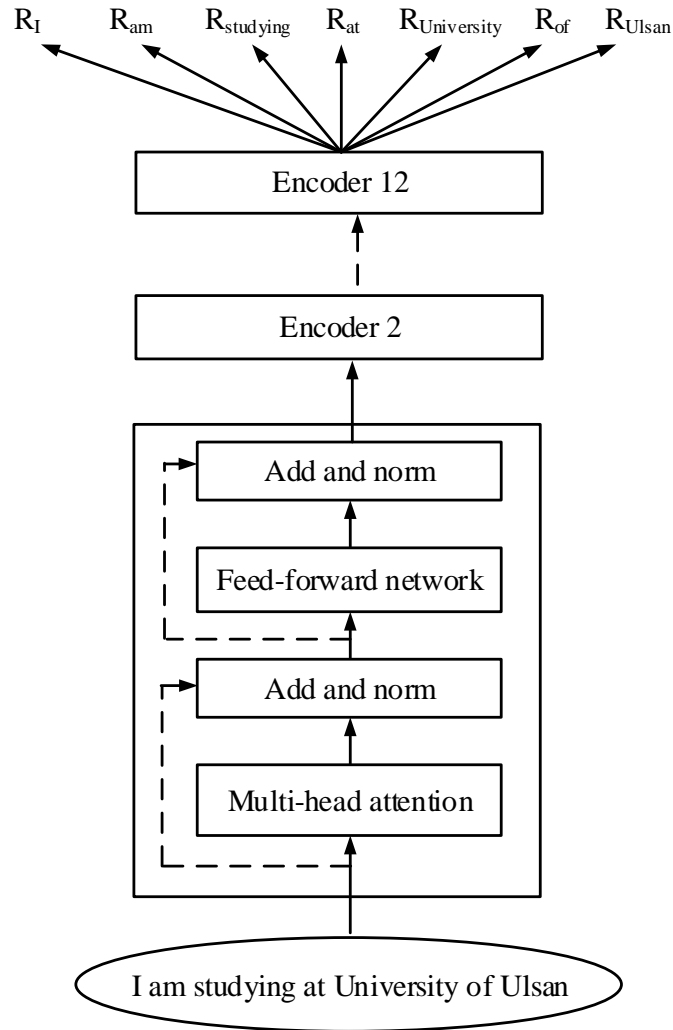


Figure 13. Generating representation of a sentence with BERT-base

3.2. Pre-trained BERT

To train BERT from scratch, we need a huge dataset and a strong computer hardware system causing. This restricts the use of BERT; therefore, using BERT on different tasks with trained weights is an optimal choice. We can also adjust (fine-tune) the weights of BERT for the new task.

To pre-train the BERT model, first, the input is converted to embedding using three embedding layers including token, segment, and position embedding. Suppose that the input includes two sentences, $s_1 = (x_1, x_2, x_3, x_4)$ and $s_2 = (z_1, z_2, z_3)$. Before feeding to the BERT, the input sentence is converted into embeddings, as shown in Figure 14. The token $[CLS]$ is added to the beginning of the first sentence and the token $[SEP]$ is added at the end of each sentence. The token in s_1 and s_2 are mapped to the embedding E_A and E_B , respectively. E_0 - E_9 represent the position of each token in the final embedding of the input. All the embeddings are combined and the final representation is fed into the BERT.

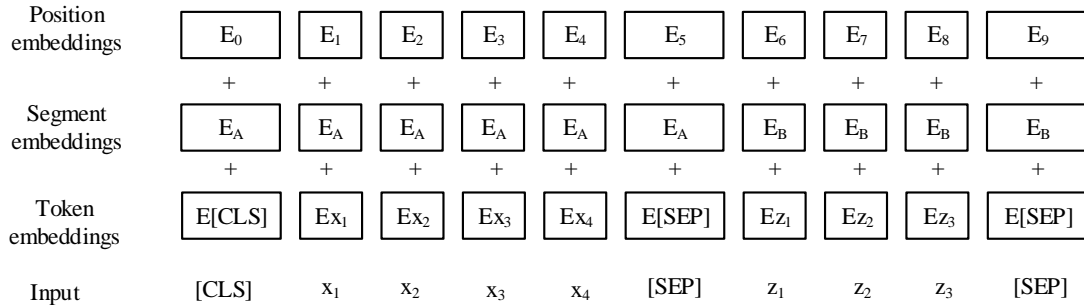


Figure 14. The representation of the input.

In the next step, the WordPiece tokenizer is used to split the word into sub-words. The tokenizer algorithm checks if a word is not in the current vocabulary then it splits the word into sub-words until reaching individual characters. The BERT vocabulary is established with 30K token and if a word is outside this vocabulary, the split will be performed. For example, if the word “*understanding*” is not in the BERT vocabulary, this will be split into the sub-words “*under*”, “*stand*”, and “*ing*”. Since “*under*”, “*stand*”, and “*ing*” are in the BERT vocabulary, the spitting process will finish. By this step, the out-of-vocabulary problem is significantly improved.

Then, the BERT model is pre-trained on two strategies including **masked language modelling (MLM)** and **next sentence prediction (NSP)**. BERT is an auto-encoding language model. To make a prediction, BERT reads the input sentence in both directions (left to right, and in the reverse direction). Due to reading the sentence from both directions, the BERT model gives better results than others.

In the strategy of MLM, 15% of the given input is randomly masked, and the network is trained to predict the masked tokens. Using BERT, the sentiment analysis is converted to the representation of the masked token, R[mask], and then this is fed to the feedforward network with a soft-max activation. This network returns the probability of all the work in the vocabulary that is replaced by the masked words. The word that has the highest probability is the result of the masked work.

In the strategy of NSP, two given sentences are fed into BERT, and it will forecast whether or not the second sentence is the successor to the first. The implementation process of the NSP task is similar to the process of creating masked words in the MLM strategy. However, in the NSP task, the output is a binary class including the isNext or notNext label as in the diagram below (suppose that s_1 is followed by s_2).

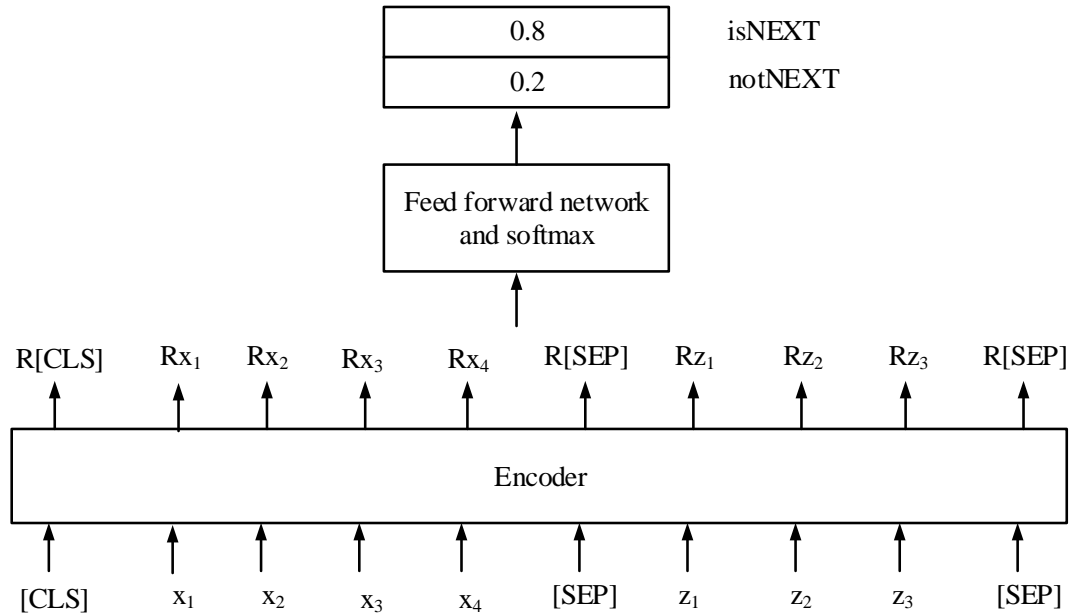


Figure 15. The strategy of NSP.

In this strategy, label *isNext* is defined to denote that sentence s_1 is followed by sentence s_2 and label *notNext* is defined to denote that sentence s_1 is not followed by sentence s_2 . Two constant sentences from the dataset are labeled as *isNEXT* and the class *notNext* is generated by taking one sentence from the dataset and pairing it with a random sentence in the dataset.

The result of applying BERT to some NLP tasks (MNLI: Multi-Genre Natural Language Inference, QQP: Quora Question Pairs, QNLI: Question Natural Language Inference, SST-2: Stanford Sentiment Treebank, CoLA: Corpus of Linguistic Acceptability) is unsurpassed other methods as Table 4.

Table 4. Comparison the result of BERT and some other systems by F1 [53].

System	MNLI	QQP	QNLI	SST-2	CoLA	Average
Pre-OpenAI SOTA	80.6	66.1	82.3	93.2	35.0	71.4
BiLSTM+ELMo+Attn	76.4	64.8	79.9	90.4	36.0	69.5
OpenAI GPT	82.1	70.3	88.1	91.3	45.4	75.4
BERT Base	84.6	71.2	90.1	93.5	52.1	78.3
BERTLarge	86.7	72.1	91.1	94.9	60.5	81.1

3.3. Pre-trained BERT variants

Since the BERT model was introduced in 2018, many variants of BERT have been published. The different variants of BERT are not only built on different data, but they also have architectural variations. Some typical versions of BERT include ALBERT, RoBERT, and ELECTRA.

3.3.1. ALBERT

ALBERT is short for A Lite version of the BERT model and this is an interesting variant of BERT. In the ALBERT model, the author included cross-layer parameter sharing and factorized embedding parameterization to reduce the training time.

The BERT-base contains 110 million parameters and this is challenging for the training process. ALBERT is pre-trained with various configurations compared with BERT. The number of encoder layers (L), and the embedding vector size (H) of ALBERT is similar to the configuration of BERT, but the number of parameters in ALBERT-base and ALBERT-large reduced to 12 million and 18 million, respectively.

In ALBERT, with cross-layer parameter sharing, the parameters of the first encoder layer are learned, and then this parameter is shared with all the other encoder layers. There are some options for cross-layer parameter sharing including all-shared (sharing the value of all sublayers of the first layer with the other encoder layers), shared feed-forward network (sharing the parameter of the feed-forward network with feed forward network of all other encoders), and shared attention (sharing the parameter of the multi-head attention of the first encoder to all other multi-head attention of the all other encoders). This leads to the number of the parameter in ALBERT being significantly reduced.

3.3.2. RoBERTa

RoBERTa stands for Robustly Optimized BERT pre-training Approach and this is also a popular variant of BERT. In deference from BERT, RoBERTa uses dynamic masking instead of static masking and it trained the model on huge data sizes. In the MLM task of the BERT model, 15% of the tokens of the given sentence are masked, then all the tokens are fed to the network. But, in the MLM task of RoBERTa, the given sentence is duplicated 10 times, and then 15% of the tokens of each duplicated sentence are masked. This leads to a significant increase in the size of the training set. RoBERTa is trained for 40 epochs and each duplicated sentence is used in only four epochs.

RoBERTa training process uses only MLM task and SNP task is not used. Besides, while BERT uses a WordPiece tokenizer, RoBERTa uses BBPE as a tokenizer. BBPE uses a byte-level sequence instead of a character-level sequence that is used in the byte pair encoding (BPE) tokenizer.

3.3.3. Multilingual BERT

BERT, RoBERT and most other variants of BERT generate a representation for English text. Multilingual BERT (M-BERT) is generated by Google to output representation for 104 different languages (not just English). M-BERT is based on MLM and SNP tasks and it used corpus from Wikipedia to train the model. The corpus used to train M-BERT was converted to lowercase and it was tokenized by WordPiece.

M-BERT is trained with data of 104 different languages and it contains 110K shared WordPiece vocabulary across all these 104 languages. The accuracy of applying M-BERT to NLP tasks usually gives good results in similarly structured languages. For instance, the fine-tuned M-BERT model for Bulgarian is better than the fine-tuned M-BERT model for Japanese because English and Bulgarian have a similar order of subject, verb, and object as following table.

Table 5. Accuracy of the POS tagging using M-BERT.

Test language Training language	English	Bulgarian	Japanese
English	96.8	87.1	49.4
Bulgarian	82.2	98.9	51.6
Japanese	57.4	67.2	96.5

3.4. Vietnamese Pre-trained BERT

BERT and its variants have had great success in many NLP tasks, but this success has been largely limited to the English language. In the Vietnamese language, 85% of Vietnamese words are compound words. However, M-BERT models that train different languages including Vietnamese used syllable-level data to train the model, leading to the low quality of the model and demanding to development of a pre-trained monolingual BERT model for Vietnamese.

VinAI (a research group of VinGroup – the largest corporation in Vietnam) has published PhoBERT to resolve the above problem. They used the Vietnamese Wikipedia corpus at the word level (e.g. đất_nước (country), thủ_đô (capital city), dân_chủ (democracy), etc) to train their BERT model. PhoBERT is based on RoBERTa and it has two versions including PhoBERT-base (150M parameters) and PhoBERT-large (350M parameters). The procedure of pre-training PhoBERT took 8 weeks with 4 GPUs.

3.5. Summary

The beginning of this chapter presented the basic idea of the BERT model. Unlike context-free models such as word2vec or Glove, the BERT model can understand the contextual meaning of the words and generate embeddings according to their context.

The pre-trained BERT model used two strategies including MLM and NSP. In MLM, 15% of the tokens are masked and the BERT model is trained to predict the masked tokens. In the NSP task, the BERT model is trained to allocate whether the second sentence is a

follow-on sentence from the first sentence or not. The pre-trained BERT model consists of the BERT-base and the BERT-large model.

The following of this chapter showed some different variants of BERT such as ALBERT, RoBERTa, Multilingual BERT and Vietnamese pre-trained BERT. The ALBERT uses cross-layer parameter sharing and factorized embedding parameterization to reduce the number of parameters. The RoBERTa uses only the MLM task for training and this need a large batch size for training. The multilingual BERT is trained with the data containing 104 different languages and this is used to compute the representation of different languages. The Vietnamese pre-trained BERT is one of the variants of BERT, this is trained with the Vietnamese dataset at the word level.

Chapter 4

Vietnamese Part of Speech

POS tagging is a challenging sequence-labeling problem. A POS defines a category of words that have similar grammar (verbs, nouns, adjectives, etc.). Most traditional sequence label tagging has used supervised learning techniques such as CRFs [54], hidden Markov models [55], or maximum entropy Markov models [56]. However, model sequence label tagging has used neural network architectures instead, because of their effectiveness [57–59]. This dissertation presents 2 methods of tagging POS including using the Bi-LSTM model and the pre-trained BERT model.

4.1. Tagging POS using Bi-directional LSTM model

4.1.1. Bidirectional LSTM

LSTM was introduced by Hochreiter & Schmidhuber [60], and its special ability is learning from long-term dependencies. LSTM is designed to solve problems of long-term dependencies in RNNs that are affected by the vanishing gradient problem. RNN can hardly remember information from long-distance steps, so the first element in the input

sequence has not influenced the results of predictive computations for the output sequence in the following steps. An LSTM network contains LSTM cells and each LSTM cell is demonstrated in Figure 16. The idea of LSTM is to add more states to the cell internal state. An LSTM cell consists of three gates to filter input and output information of the cell including forgetting gate f_t , input gate i_t , and output gate o_t . The remember vector is usually called the forget gate and this decides which information is ignored. The save vector is usually called the input gate and this determines which information should enter the cell state. The focus vector is usually called the output gate and this decides which information is the output of the cell state.

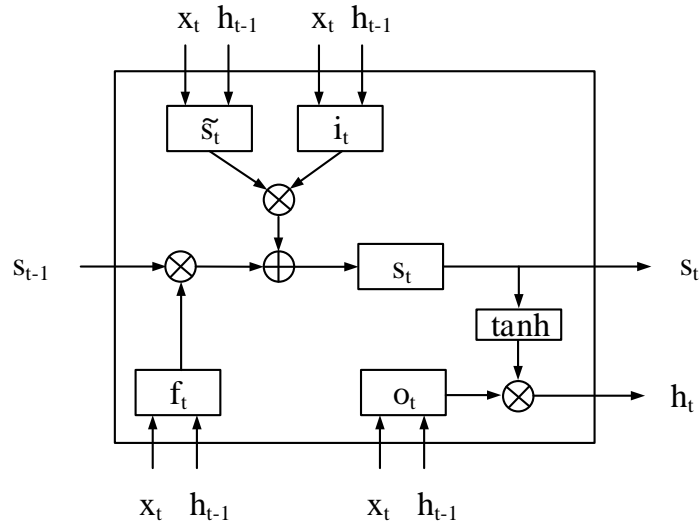


Figure 16. An LSTM cell.

x_t is the input vector at time step t ; $W_{f,x}$, $W_{f,h}$, $W_{\tilde{s},x}$, $W_{\tilde{s},h}$, $W_{i,x}$, $W_{i,h}$, $W_{o,x}$, and $W_{o,h}$ are weight matrices in each LSTM cell; b_f , $b_{\tilde{s}}$, b_i , and b_o are bias vectors; f_t , i_t , and o_t are active values of the forget gate, the input gate, and the output gate, respectively; s_t and \tilde{s}_t

are cell internal state and candidate value state vector, respectively; h_t is the output value of the LSMT cell.

During forward propagation, the cell internal state s_t and the output value h_t are calculated as follows:

In the first step, the LSTM cell decides which information should be removed from the cell internal state in the previous time step s_{t-1} . The activation value f_t of the forget gate at the time step is calculated based on the present input value, the output h_{t-1} of the LSTM cell at the previous time step, and the bias b_t of the forget gate. The sigmoid function transforms all activation values into a range of values between 0 (completely forgetting) and 1 (completely remembering).

$$f_t = \sigma(W_{f,x} * x_t + W_{f,h} * h_{t-1} + b_f) \quad (15)$$

In the second step, the LSTM cell decides which information should be added to the cell internal state s_t . This step consists of two computations for \tilde{s}_t and f_t . The candidate value \tilde{s}_t represents the potential information that needs to be added to the internal state and it is calculated as follows:

$$\tilde{s}_t = \tanh(W_{\tilde{s},x} * x_t + W_{\tilde{s},h} * h_{t-1} + b_{\tilde{s}}) \quad (16)$$

Besides, the activation value i_t is calculated by

$$i_t = \sigma(W_{i,x} * x_t + W_{i,h} * h_{t-1} + b_i) \quad (17)$$

In the thirist step, the value of the cell internal state s_t is calculated based on the calculation results obtained from the previous steps using the following formula:

$$s_t = f_t \otimes s_{t-1} + i_t \otimes \tilde{s}_t \quad (18)$$

where \otimes is Hadamard product⁹.

In the final step, the output value h_t of the LSTM cell is calculated by:

$$o_t = \sigma(W_{o,x} * x_t + W_{o,h} * h_{t-1} + b_o) \quad (19)$$

$$h_t = o_t \otimes \tanh(s_t)$$

In the Bi-LSTM model, the output at step t depends on both the front elements and the behind elements. For example, to forecast missing words in a sentence, it is necessary to consider both the previous parts and the next parts of the sentence. Therefore, we can consider the model as the overlap of two LSTM networks facing each other. At this time, the output is calculated based on the hidden states of both LSTM networks. The Bi-LSTM structure is shown in Figure 17.

⁹ [https://en.wikipedia.org/wiki/Hadamard_product_\(matrices\)](https://en.wikipedia.org/wiki/Hadamard_product_(matrices))

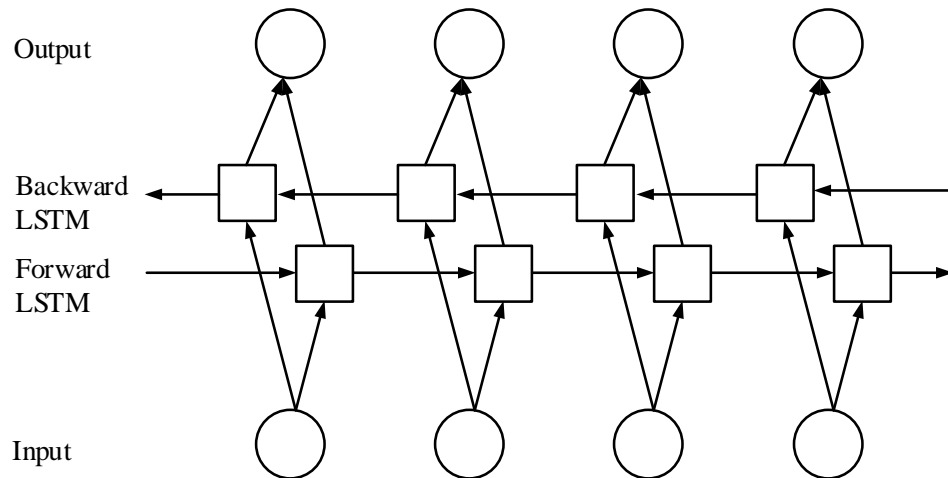


Figure 17. Structure of Bi-LSTM.

4.1.2. Bi-LSTM-CRFs for POS Tagging

Figure 18 illustrates the structure of Bi-LSTM-CRFs for POS tagging. CRFs are a probability model often applied to the predictive structures in sample identity and machine learning. In the combination of Bi-LSTM and CRF, a sequence input that has passed the Bi-LSTM becomes the input for the CRF layer. Then, the CRF layer predicts the POS output sequence that best corresponds to the input string. Word embedding of each word is fed into the Bi-LSTM model to extract useful information about the semantics, the morphology of the word and the context surrounding the word. Next, the CRF layer will process the above information as features to make predictions about the POS label for each word. The combination of the Bi-LSTM and CRF takes advantage of the strengths of both models including the intelligent feature extraction of LSTM and the powerful ability of CRF to predict the label sequence.

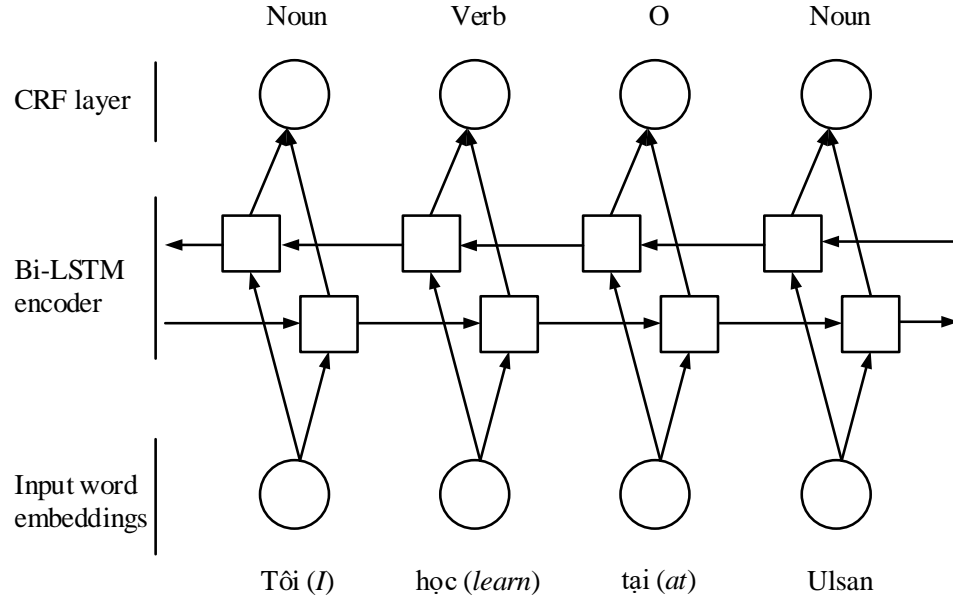


Figure 18. Bi-LSTM-CRFs for POS.

4.2. Tagging POS using Pre-trained BERT model

For Vietnamese sentences, we base on the PhoBERT model [31] to tag POS. This model is trained on 20GB of Vietnamese text that is collected from Wikipedia. Before training the model, all word is segmented to generate word in word-level.

In this model, the PhoBERT base model is applied in the encoding layer to generate the embedding of each token of the given input. Assume that the given input is a sequence of the form $x = (x_1, x_2, \dots, x_n)$, where n is the length of the sequence; the embedding e_i of the word x_i is calculated by:

$$e_i = \text{PhoBERT}_{\text{base}}(x_{1:n}, i) \quad (20)$$

The contextualized word embedding e_i is fed into a feed-forward network and then, by a soft-max predictor, the POS of each word in the given input will be predicted as the following:

$$p_i = \text{softmax}(FFN(e_i)) \quad (21)$$

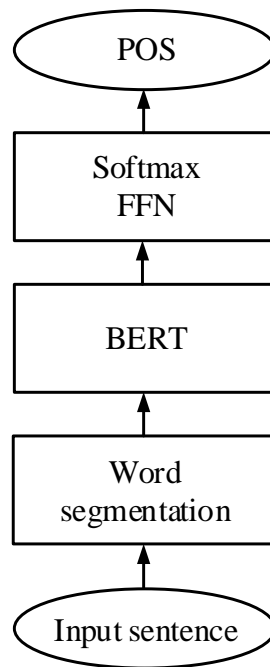


Figure 19. Illustration of POS tagging with BERT.

The experimental exercise is based on the benchmark datasets of the VLSP 2013 POS tagging corpus. The result of this method is superior when compared to other research on POS for Vietnamese, as in Table 6.

Table 6. Performance of some Vietnamese POS tagging systems by F1 score [31].

POS tagging systems	Accuracy
BiLSTM-CNN-CRF	95.4
VnCoreNLP-POS	95.9
jPTDP-v2	95.7
jointWPD	96.0
PhoBERT _{base}	96.7

The transformation in form of Vietnamese sentences in the bilingual corpus is demonstrated as shown in the following table:

Table 7. Form of Vietnamese sentences after applying POS tagging.

Form	Meaning
Initial	anh ấy làm việc tại trường đại học FPT (he works at the University of FPT)
WSeg.	anh_ấy làm_việc tại trường đại_học FPT
WSeg. and POS	anh_ấy N làm_việc V tại E trường N đại_học N FPT Ny

N, V, E, and Ny are the tagged POS indicating the noun, verb, preposition, and abbreviation of the noun, respectively.

Similar to the POS tagging task, NER tagging also uses a linear prediction layer on top of the BERT architecture. In the NER tagging using the pre-trained Vietnamese BERT model, the dataset from VLSP 2016 NER task is used to identify personal names, locations, and organizations. The performance of the NER task using the pre-trained model is best compared to some other systems as shown in Table 8.

Table 8. Performance of some Vietnamese NER tagging systems by F1 score [31].

NER tagging systems	Accuracy
BiLSTM-CNN-CRF	88.3
VnCoreNLP-NER	88.6
BiLSTM-CNN-CRF + ETNKP	91.1
XLM-R	92.0
PhoBERT _{base}	93.6

4.3. Summary

This chapter presented two approaches to tag POS for Vietnamese sentences by using Bi-LSTM and a pre-trained model called PhoBERT.

LSTM is an improved network of RNN to solve the problem of remembering long steps of RNN. The Bi-LSTM model contains a forward LSTM and a backward LSTM. Each LSTM cell consists of three gates to filter information including forget gate, input gate, and output gate. The Bi-LSTM network combines with the CRF to create a Bi-LSTM-CRFs structure to tag POS for the input sequence. The CRF layer receives output from the Bi-LSTM network and predicts the POS corresponding with the input word by using a probability model.

In terms of tagging POS for Vietnamese sentences by the pre-trained BERT model, we used PhoBERT as an embedding method. The POS tagging is predicted by a Softmax FFN layer. Currently, tagging POS for Vietnamese using the pre-trained BERT model is the best method.

Chapter 5

Vietnamese-Korean Parallel Corpus

5.1. Parallel Corpus Acquisition

This research introduces the Vietnamese-Korean parallel corpora for use in training MT models. Vietnamese-Korean parallel corpus is built with building process for both are similar, as shown in the following figure:

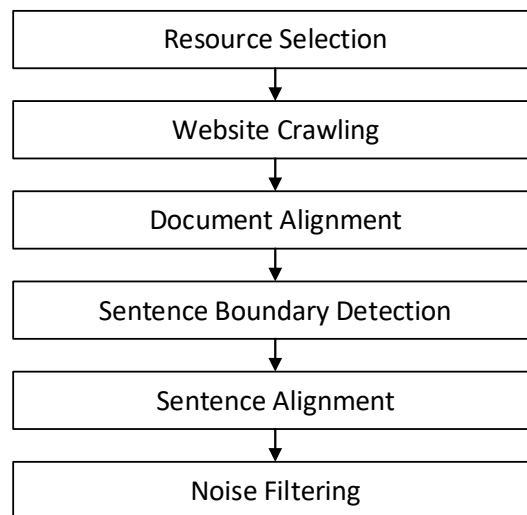


Figure 20. The acquisition process of the UPC.

The initial step of the parallel corpora collection is to select resources that contain Korean and Vietnamese sentences. Since the resources determine the quality of the parallel corpora, we had to select resources that have good alignment and literal translations. After carefully verifying the contents, the following resources were selected to build the parallel corpora.

- “National Institute of Korean Language’s Learner Dictionary¹⁰ (NIKLLD)”, is an online dictionary containing definition statements of words in Korean and ten other languages including Vietnamese.
- “Watchtowers and Awake!¹¹” where “Watchtowers” is a monthly online magazine and “Awake!” is a bi-weekly online magazine.
- “Books and Brochures for Bible Study¹²”, contains multilingual books in PDF format.
- “Danuri portal of information about Korean life”¹³, contains a guide for living in Korea (HTML format) and a series of multicultural magazines – Rainbow¹⁴ (PDF format).
- Online Korean learning website¹⁵ that contains Korean, and Vietnamese texts available in HTML format.

In the next step, we downloaded the PDF files and other contents from the websites. Except for NIKLLD where we directly received word definition statements from the

¹⁰ <https://krdict.korean.go.kr/eng/mainAction>

¹¹ <https://www.jw.org/en/publications/magazines>

¹² <https://www.jw.org/en/publications/books>

¹³ <https://www.liveinkorea.kr>

¹⁴ <https://www.liveinkorea.kr/app/rainbow/webzineList.do>

¹⁵ <http://hanquoclythu.com/>

organization in charge. For the rest of the selected resources, we generated URLs for each website and made a web spider crawl the contents and PDF files of these websites. The “National Institute of Korean Language’s Learner Dictionary”, contains the definition statements of Korean and Vietnamese words aligned at the sentence level, which we directly received. Hence, we did not need to crawl texts nor align at the document or sentence level. The “Watchtower and Awake!” magazines are available on websites (i.e., HTML format). We first generated an URL based on the structure of the magazine over time. Then, we used these URLs to feed an automatic crawler that was developed based on the “requests” library of Python to download each URL and we used the “Beautiful Soup” library to parse the HTML format and extract the content in texts. Likewise, the automatic crawler was used to extract the content of “Books and Brochures for Bible Study” and the “Danuri portal of information about Korean life”. However, their URLs were generated according to their URL structures.

In the document alignment stage, each URL of the magazine is an issue that contains several articles or topics. In this step, we separated topics by topic and matched them in each language pair Vietnamese-Korean. Then, we removed topics that only exist in one language. After which the document pairs were correctly aligned.

Data crawling from the website is in the form “*<tag> content </tag>*”. For a website written in standard HTML, the text is usually contained in pairs of “*<article> </article>*” or “*<p> </p>*” tags. Therefore, we use python to get the content between these tags pairs. We execute the process of sentence boundary detection and sentence alignment when obtaining the parallel corpora. After a sentence (boundary) in Korean was detected, the corresponding sentence in English or Vietnamese was examined and alignment was made

between them. The boundary detection was done based on the period “.”, the question mark “?”, or the exclamation mark “!” for texts extracted from the magazines. For the data collected from “National Institute of Korean Language’s Learner Dictionary”, the sentence pairs were aligned by humans. For the rest, each paragraph pair in the language pairs is aligned. Then, we count the number of sentences of each paragraph, and pairs of paragraphs with an equal number of sentences are retained for processing in the next step, thus, aligning sentence pairs correctly.

The data crawled from websites contains numerous noises (e.g. HTML tags and special symbols). For an instant, the character “&” is represented by “&” tag or the character “©” is represented by “©” tag. These noises expand the size of the training dataset; therefore, it is necessary to remove them from the parallel corpora. Besides, according to the data that we have gathered from various sources, sentences with more than 50 words are very rare. Training sentences that are too long is wasteful and lengthens the training process because the system will need to process many unnecessary calculations. Therefore, we removed sentences with over 50 words from the parallel corpora. We also removed duplicated sentences, which sometimes occur as a result of collecting data from many resources. These corpora were re-corrected by the splitting of sentences and stored one sentence per line on a disk file. All data is encrypted in utf-8.

5.2. The Parallel Corpora Analysis

5.2.1. Applying Morphological Analysis and Word-sense Annotation for Korean sentences

The Korean text in the parallel corpus after applying Morph. Ana. and word-sense annotation system (UTagger) is shown in Table 9. In the example sentence, MA splits tokens into sub-tokens and adds the corresponding morphemes to these sub-tokens. Besides, the WSD process attaches the corresponding sense-codes to the tokens. For example, the token “*nun-e*” is split into two tokens (i.e. “*nun*” and “*e*”); and the homograph “*nun*” occurs two times with two disparate meanings “*snow*” and “*eye*” that are represented by sense-code “04” and “01”, respectively. After applying UTagger, the homograph “*nun*” was transformed into “*nun_04*” and “*nun_01*” according to its meaning. Morphemes existing in the same *eojeol* are separated by the plus symbol “+”.

Table 9. Morphological analysis and word-sense annotated sentence.

	눈에 미끄러져서 눈을 다쳤다.
Initial form	<i>nun-e mi-kkeu-leo-jyeo-seo nun-eul da-chyeoss-da.</i> (I slipped over the snow and my eyes are injured)
Form after applying UTagger	눈_04/NNG + 예/JKB 미끄러지/VV + 어서/EC 눈_01/NNG + 을/JKO 다치_01/VV + 었/EP + 다/EF + ./SF <i>nun_04 + e/JBK mi-kkeu-leo-ji/VV + eo-seo/EC nun_01/NNG + eul/JKO da-chi_01/VV + eoss-da/EF ./SF</i>

The token size of Korean texts in the parallel corpus increases due to the morphemic segmentation while the vocabulary size is reduced by the recovery of initial forms. UTagger annotated different sense-codes to the same form of words, thus causing the expansion of the number of Korean vocabulary in the parallel corpus. Table 10 shows an

example of the increase in the token size and reduction in the vocabulary size of Korean texts after undergoing Morph. Ana.

Table 10. The variation in the number of token and vocabulary in UPC after Morph. Ana.

No.	Initial		Morphological Analysis		
	Token / Voc.	Meaning	Form	Token	Voc.
1	<i>jib-e-seo</i>	at home	<i>jib</i> /NNG <i>e-seo</i> /JKB	$\frac{jib}{e-seo}$	jib
2	<i>jib-e</i>	at home	<i>jib</i> /NNG <i>e</i> /JKB	$\frac{jib}{e}$	hag-gyo
3	<i>hag-gyo-e-seo</i>	at school	<i>hag-gyo</i> /NNG <i>e-seo</i> /JKB	$\frac{hag-gyo}{e-seo}$	ga-ge
4	<i>hag-gyo-e</i>	at school	<i>hag-gyo</i> /NNG <i>e</i> /JKB	$\frac{hag-gyo}{e}$	e-seo
5	<i>ga-ge-e-seo</i>	at store	<i>ga-ge</i> /NNG <i>e-seo</i> /JKB	$\frac{ga-ge}{e-seo}$	e
6	<i>ga-ge-e</i>	at store	<i>ga-ge</i> /NNG <i>e</i> /JKB	<i>ga-ge</i>	

Compared with some other studies, UTagger showed outstanding results in both Morph. Ana. and WSD tasks as shown in Table 11.

Table 11. The result of the Morph. Ana. and WSD research.

Research	MA	WSD
Phrase-based Statistical Model [3]	96.35	
Bi-Long Short-Term Memory [61]	96.20	
Bidirectional Recurrent Neural Network [62]		96.20
Statistical-based [63]		96.42
UTagger	98.20	96.52

5.2.2. Applying Word Segmentation and POS for Vietnamese sentences

In the Vietnamese sentences, we first used the tool of Nguyen [64] to segment words. The task of word segmentation (WSeg.) is dividing the written text into meaningful units. Then, we applied the tool [65] for POS tagging. The method of applying POS to the Vietnamese corpus changed the form of the sentences, as shown in Table 7 (Form of Vietnamese sentences after applying POS tags).

5.3. The detail of Vietnamese-Korean parallel corpora

After a series of data collection and analysis, we obtained 412,317 sentence pairs for the Vietnamese-Korean parallel corpora with 4,782,063 tokens and 5,958,096 tokens in Korean text and Vietnamese text, respectively. Table 12 shows the detail of the Vietnamese-Korean parallel corpus in UPC.

Table 12. The detail of the Vietnamese-Korean parallel corpus.

		Number of sentences	Average length	Number of tokens	Number of vocabularies
Korean	Initial	412,317	11.6	4,782,063	389,752
	Morph. Ana. and WSD		20.1	8,287,635	68,719
Vietnamese	Initial		14.5	5,958,096	40,090
	WSeg. and POS		11.2	4,609,163	49,208

In Korean texts, after the morphological analysis and WSD processes, the number of tokens expanded from 4,782,063 to 8,287,635 while the number of vocabularies reduced from 389,752 to 68,719. The sentence length distributions of the Korean-Vietnamese parallel corpus is demonstrated as Figure 21. In original form, the average length of Korean sentences is 11.6 tokens and most of them have a length from 5 to 14 tokens, while the average length of Vietnamese sentences is 14.5 tokens and most Vietnamese sentences have a length from 8 to 18 tokens. Applying WSeg. for Vietnamese sentences reduces the average length and the number of tokens but this increases the number of vocabularies. The use of POS tagging does not change the average length, the number of tokens and the number of vocabularies of the Vietnamese sentences.

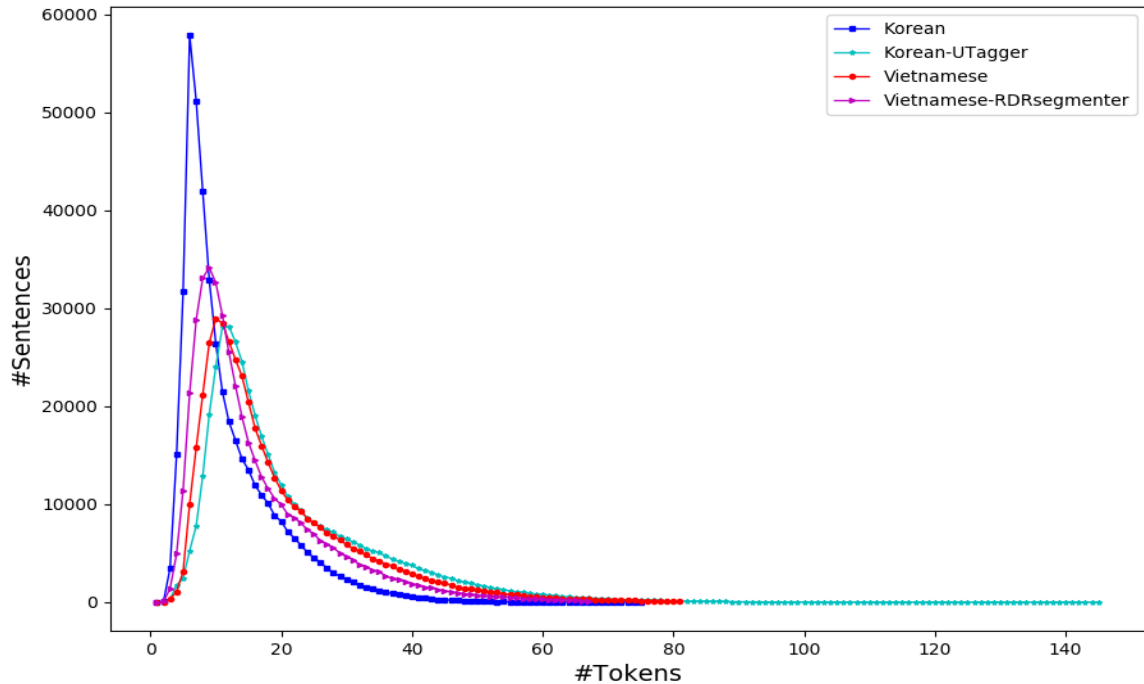


Figure 21. Sentence length distributions of the Vietnamese-Korean parallel corpus.

5.4. Summary

This chapter describes a parallel corpus for the Vietnamese-Korean MT systems. This corpus is collected from various sources such as NIKLLD, books, and online websites. We have performed various processing steps including resource selection, website crawling, document alignment, sentence boundary detection, sentence alignment and noise filtering to obtain more than 412 thousand sentence pairs.

In order to make the parallel corpus more suitable for MT, we performed a series of processing steps in both Korean and Vietnamese. On the Korean side, we applied Morph. Ana. to split one word into multiple sub-tokens and applied WSD to distinguish the sense of a word in different contexts. On the Vietnamese side, WSeg. and POS tagging is added.

After data collection and processing, a Vietnamese-Korean parallel corpus consisting of 412,317 sentence pairs was established. The Vietnamese sentences have an average length of 14.5 tokens and the Korean sentences have an average length of 20.1 tokens.

Chapter 6

Vietnamese-Korean Machine Translation

6.1. Models of Vietnamese-Korean Machine Translation

In this dissertation, experiments are mainly performed on NMT platforms. Besides, SMT and NMT with BPE are also performed to compare with our methods. In the detail, our MT systems translate Vietnamese sentences to Korean sentences in some different ways as the following:

- *Baseline*: building NMT system using the Vietnamese sentences with WSeg. and the Korean sentences with Morph. Ana. and WSD.
- *SMT*: building SMT system uses the corpus as in the Baseline system.
- *NMT with BPE*: building the NMT system uses the corpus in which text of both Vietnamese and Korean was tokenized into sub-word by BPE [66].
- *POS*: building NMT system uses Vietnamese sentence with WSeg. and POS tagging and Korean sentences as in the Baseline system.

- *VietBERT-NMT*: building NMT system with the Vietnamese BERT model (PhoBERT) uses the dataset as in the Baseline system.
- *D-NMT*: building NMT system with the Vietnamese BERT model (PhoBERT) uses the dataset as in the Baseline system, but the inputs of BERT in the NMT model are a pair of a sentence and its preceding sentence in the corpus (document-level).

6.1.1. Machine translation with NMT model

Our Vietnamese-Korean NMT systems are implemented on the OpenNMT-py framework PyTorch version [4]. The encoder uses word2vec to convert a sequence of the source language into embedding vectors. Each embedding vector in the encoder represents one word of input sequence x . This encoding is typically done using an LSTM network. The decoding process is analogous to encoding and takes the sequence of previously generated target words, and then converts them to the corresponding embedding vectors. This embedding vector provides a context-free representation, this is based on previous words. The attention component matches encoder hidden states and decoder hidden states. The detail encoder-decoder with attention is mentioned in section 2.2.

6.1.2. Combination of BERT and NMT

The foundation of the NMT with the BERT model still uses the sequence-to-sequence model with an attention approach. In addition, BERT is used to extract the representations for an input sentence. Then, through the attention mechanism, representations are fused with the layers of both the encoder and decoder in the MT model. In this paper, we inherit the research of Zhu et al. [30] to use BERT to extract representation for an input sentence. Then, these representations are fused with both

encoder and decoder layers in the NMT model based on attention layers as shown in Figure 22.

Suppose that $x = (x_1, x_2, \dots, x_n)$ and $y = (y_1, y_2, \dots, y_m)$ are respectively source sentence and target sentence; n and m are the length of the source sentence and target sentence. l_x and l_y are the number of units (sub-word or word) in each sentence x and y , respectively; both encoder and decoder contain L layers; R_B and R_E are the output of the BERT and the encoder, respectively.

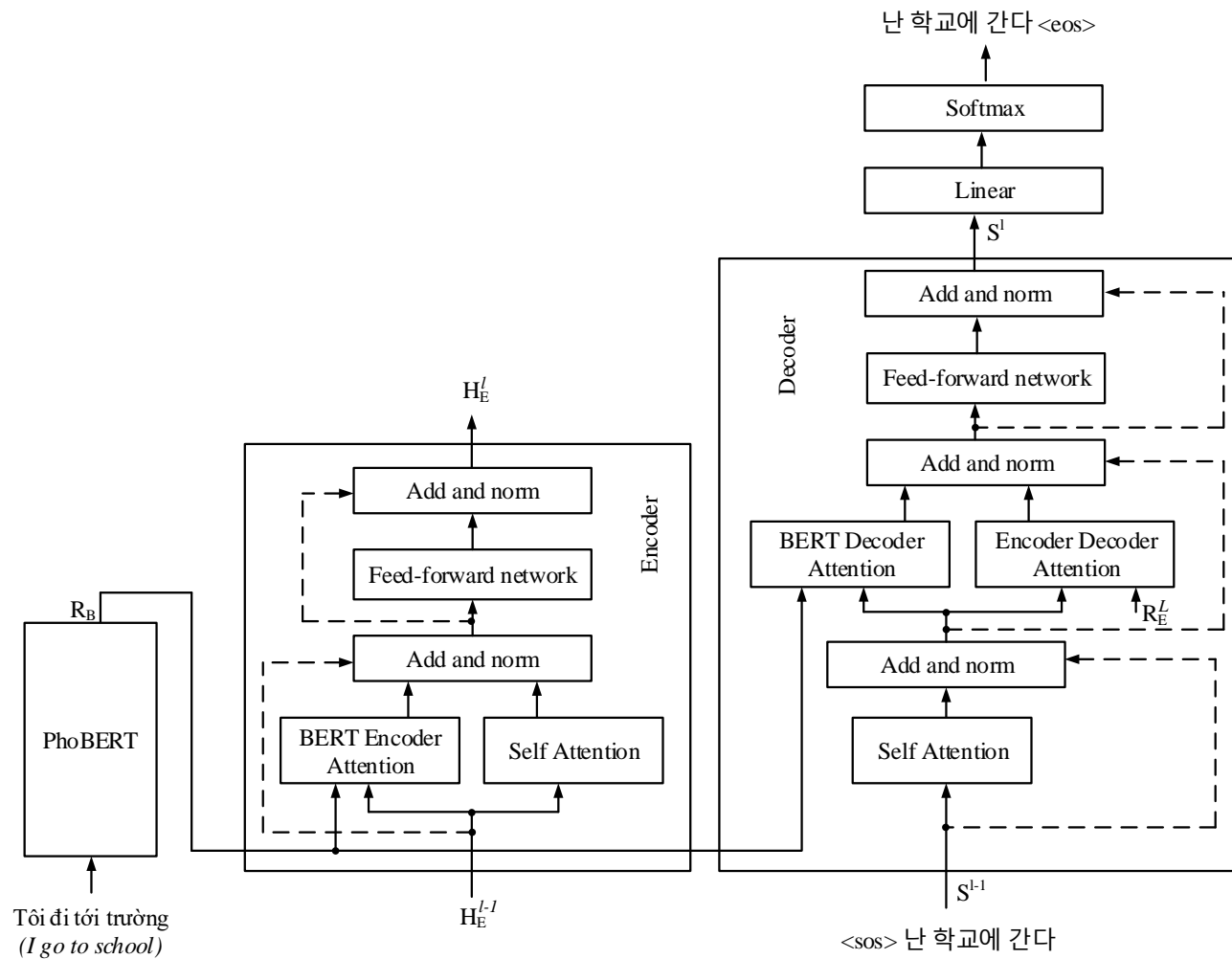


Figure 22. A demonstration of translation of a Vietnamese sentence to a Korean sentence with the PhoBERT model.

First, BERT encodes a given input x into representation R_B , and this is fed into the encoder. The output of the i -th encoder layer ($i \in l_x$) is calculated by:

$$\tilde{r}_i^l = \frac{1}{2} \left(\text{attn}_S(r_i^{l-1}, R_E^{l-1}, R_E^{l-1}) + \text{attn}_B(r_i^{l-1}, R_B, R_B) \right) \quad (22)$$

where attn_S and attn_B are attention layers; and they are computed with different parameters as follows:

$$\text{attn}(q, K, V) = \sum_{i=1}^{|K|} \exp \left((W_q q)^T (W_k k_i) \right) \quad (23)$$

where q , K and V respectively denote query, key, and value; $|K| = |V|$; q is a dimensional vector; W_q and W_k are the parameters to be learned. Then, the output of the encoder R_E^l is calculated by:

$$R_E^l = (\text{FFN}(r_1^l), \text{FFN}(r_2^l), \dots, \text{FFN}(r_{l_x}^l)) \quad (24)$$

where the value of a feed-forward network of any input x (e.g. $\text{FFN}(x)$) is calculated by:

$$\text{FFN}(x) = W_2 \max(W_1 x + b_1, 0) + b_2 \quad (25)$$

where W_1 , W_2 , b_1 , b_2 are the parameters that are learned.

In the last step, the l -th decoder layer ($l \in l_y$) is calculated by

$$\begin{aligned} \hat{s}_t^l &= \text{attn}_S(s_t^{l-1}, s_{<t+1}^{l-1}, s_{t+1}^{l-1}); \\ \tilde{s}_t^l &= \frac{1}{2} \left(\text{attn}_B(\hat{s}_t^l, H_B, H_B) + \text{attn}_E(\hat{s}_t^l, H_E^l, H_E^l) \right), s_t^l = \text{FFN}(\tilde{s}_t^l) \end{aligned} \quad (26)$$

where s_t^l is the hidden state of the l -th layer in the previous time step t (i.e., $s_{<t}^l = (s_1^l, s_2^l, \dots, s_{t-1}^l)$); $attn_S$, $attn_B$ and $attn_E$ are respectively self-attention model, BERT-decoder attention model, and encoder-decoder attention model. The output \hat{y}_t is generated by mapping s_t^l via a linear transformation and soft-max function. When the last token in the sentence is achieved, the decoder process will be completed

This research carried out a series of bi-directional translation experiments between Korean and Vietnamese to assess the effects of POS tagging and the pre-trained BERT model on the performance of NMT

Before putting into MT models, all words in sentences are split into sub-word by BPE [66]. For the transformer model, the dropout ratio is set to 0.3; the embedding dimension, FNN dimension, and the number of layers in the model are set to 512, 1024 and 6, respectively.

6.1.3. SMT and BPE

Statistical machine translation

We performed the Vietnamese-Korean SMT experiment based on the open-source software Moses [1]. This toolkit is a statistical MT engine that can be applied to train statistical models to interpret the source string to the target string. We have downloaded and compiled Vietnamese-Korean SMT based on Linux version 3.16.

Byte pair encoding

BPE [66] is an approach that segments a word into sub-words. This method followed the NMT architecture but the input text is segmented into subword units to handle the problem of out-of-vocabulary in the word-level NMT model. In our NMT system, before the parallel corpus is put into the translation model, all Vietnamese and Korean sentences are applied BPE [66].

6.2. Result Evaluations

In this dissertation, the two most popular evaluation methods were used to evaluate the quality of MT systems including BLEU [67] and TER [68].

BLEU

BLEU is the most common method for assessing the accuracy of MT systems and this indicates the similarity between the candidate text and the reference texts. This method counts the number of matching n-grams of candidate and reference. The value of BLEU ranges from 0 to 100. A result of 0 means that the translation is very poor and a result of 100 means that the translation is perfect. The BLEU method is considered to be fast, low in computational cost, and has a high similarity to human evaluations.

TER

The TER is used to measure the amount of correction that a translator needs to produce a perfect translation. This is based on the edit distance between the candidate and the reference. TER scores range from 0 (perfect match with the reference translation) to 1

(complete mismatch). The accuracy of our Vietnamese-Korean MT systems is shown in the table below.

Table 13. The results of the translation system

Systems	BLEU	TER (%)
Baseline	25.44	65.03
SMT	24.22	66.98
BPE	22.67	68.57
POS	26.51	62.07
VietBERT-NMT	28.22	58.12
D-NMT	28.45	57.54

6.3. Discussion

In general, applying the pre-trained contextual embedding model to NMT at the document-level gave the best results, followed by the method of VietBERT-NMT at sentence-level and applying POS tagging. Both methods of SMT and BPE showed lower results than the baseline system.

6.3.1. Comparison with SMT and BPE

Table 13 shows that the accuracy of the SMT method is less than the accuracy of the NMT method by 1.22 BLEU points and 1.95 TER points, respectively. The NMT model is based on a neural network and this can generate a more natural target sentence than the

SMT model. The performance of NMT has also been shown to be superior to that of SMT in numerous other studies.

Besides, the BPE method shows high efficiency when translating from English to other languages, but our Vietnamese-Korean MT experiment based on this method gives not too good results when compared with other methods. The reason for this lower result is that we have applied word segmentation to convert Vietnamese from syllable-level to word-level, the encoder will have better input embedding and the result of the NMT system significantly improve. The method of BPE split word into sub-tokens, this method reverse the method of applying word segmentation. This method is better than the systems using original corpus (without word segmentation) but it is not good when compared with the method of applying word segmentation to the corpus.

6.3.2. Impact of Vietnamese POS tagging

In Vietnamese, various words have different meanings depending on their word type. For example, the word "*bàn*" can be a verb (to discuss) or it can be a noun (discussion or table) depending on their word type. To overcome this problem, annotation POS is used to add information to Vietnamese sentences. The word "*bàn*" is transferred into "*bàn/V*" or "*bàn/N*". The POS tagging indicates the syntactic function of words, thus leading to different representations of the word "*bàn*" and each representation will map to another word in the target language. Therefore, the MT systems easier to find words with the corresponding meaning in the target language. As a result, POS makes the MT more accurately. Specifically, after applying POS tagging for Vietnamese sentences, our

Vietnamese-Korean NMT system increased its BLEU score by 1.07 points and 2.96 TER points compared with the baseline systems.

6.3.3. Impact of Vietnamese BERT on Neural Machine Translation

In the VietBERT+NMT system, the Vietnamese BERT model was used to extract representation for a Vietnamese sentence. Different from context-free models such as word2vec, BERT can recognize the meaning of words according to the context and generate the corresponding embedding according to different contexts. For example, when generating the representation of the word “*em bé*” (baby) and the word “*nó*” (it) in the sentence “*em bé thích ăn kem vì nó ngon*” (the baby eats ice cream because it is delicious), context-free models generate a different representation for the word “*em bé*” and the word “*nó*” because they are different words. In contrast, contextual embedding models learn that the word “*nó*” is “*em bé*” in terms of meaning. Therefore, contextual embedding models generate almost similar representations for the word “*em bé*” and the word “*nó*”.

Besides, the output features of the BERT model are fully exploited; therefore, BERT generates a better representation than the context-free word embedding model of the NMT model. In addition, the pre-trained features of BERT were fused with NMT. Thus, the NMT model can take advantage of the features from BERT. Then, based on the attention mechanism, this representation is bridged to each layer of the encoder and decoder of the NMT model. Thanks to clear input, the decoder of the NMT model generates better target sentences. Our experiment also showed that using Vietnamese BERT in

combination with NMT led to a significant improvement in the quality of the Vietnamese-Korean translation system by 2.78 BLEU points and 6.91 TER points.

Furthermore, with the next sentence prediction (NSP) task BERT model can understand the adjacent sentences, so we can use a couple of sentences (the current sentence and the previous sentence) as input for the MT model (document-level). Not only learning the meaning of the current sentence, the MT model learns the meaning of the preceding sentence. Therefore, the MT model can deeper understand the meaning of the sentence and gives a better target sentence. Our implementation also shows that the performance Vietnamese-Korean NMT at the document-level is slightly better than that of the sentence-level.

6.4. Summary

Chapter 6 presents experiments and results of Vietnamese-Korean MT. NMT with Vietnamese sentences has been added POS annotation and MT that combines Vietnamese contextual embedding and NMT has been implemented. Besides, we also perform 2 other MT methods including SMT and BPE to compare with the 2 methods mentioned above.

This chapter also mentions two methods to evaluate the accuracy of MT systems including BLEU and TER. Large BLEU scores represent a good MT system; however, if the TER score is small, the MT system is more accurate.

The last part of this chapter discussed the method of POS and contextual embedding on MT. Like many other languages, a word in Vietnamese can have many meanings depending on its grammatical role in the sentence (verb, noun, adjective, etc). The

application of POS tagging helps the translator to generate different vectors for the same word if they differ in word type.

In another approach, contextual embedding also generates different vectors depending on the context, and they can generate similar vectors of words with similar meanings in the same sentence. Therefore, among the Vietnamese-Korean MT methods mentioned above, the method that combines contextual embedding and NMT gives the best results.

Conclusions

7.1. Achievements

In this study, we have collected and processed the Vietnamese-Korean parallel corpus for MT including more than 412 thousand sentence pairs. To achieve this dataset, we went through a series of steps such as resource selection, crawling data, sentence alignments and noise filtering.

We found that most Vietnamese words exist in form of compound words, however, in the initial data, Vietnamese words exist in a single word form. This leads to difficulty in determining the border of words and reduces the quality of MT. Therefore to increase the quality of MT, we used WSeg. to define the border for the Vietnamese. Besides, there are various words in Vietnamese having different meanings depending on their grammatical role in the sentence. Clarifying the grammatical composition of words in a sentence can also increase the quality of MT. To handle this issue, we have applied POS tags to Vietnamese sentences and the experiments show that POS tagging can increase the quality of MT through 2 evaluation methods including BLEU and TER.

Furthermore, there are numerous words that have different meanings in different contexts, which also leads to difficulty in understanding the meaning of sentences. We have used the best current contextual embedding for Vietnamese called PhoBERT to create different vectors for the same word in different contexts. The application of this pre-trained contextual embedding model significantly improves the quality of MT, besides it is easier to train MT because the MT system does not have to train a huge data from the beginning.

In conclusion, we performed a series of experiments on a data set consisting of more than 412 thousand Vietnamese-Korean sentence pairs that we directly collected and compared the results of these MT with each other. The results indicated that this method of applying contextual embedding for Vietnamese sentences is superior to many other methods that we have done before such as SMT, BPE, or the method of applying POS tags for Vietnamese sentences before putting it in the MT model.

7.2. Future work

Because data related to low-resource languages such as Vietnamese or Korean is very rare. Our current dataset is collected mainly from religious and economic online magazines, the amount of data on other topics related to daily life is limited. The data is enough to test the effectiveness of the MT models, but building an application that can be used in real life is still limited. We plan to collect even more data to build a real-world usable machine translation.

Besides, our experiments are carried out in the direction from Vietnam to Korea. We also plan to incorporate the Korean BERT model to encourage the quality of the Korean-Vietnamese MT systems.

In addition, we plan to do some experiments on Vietnamese-Korean MT on some other models such as GPT-3. Generative Pre-trained Transformer 3 (GPT-3) is emerging as a highly effective model in the fields of natural language processing. GPT-3 can perform a wide range of NLP tasks without finetuning on any particular task. It can translate text, answer questions, reading comprehension, write poetry and even perform calculations.

Furthermore, we also plan to research MT architecture, and we hope that we can propose a new architecture for MT in the near future.

Moreover, in this thesis, we implemented MT experiments with input as a text and output as a corresponding text. In the future, we plan to research and develop a Vietnamese-Korean MT system with voice input or output. We found that building translators with speech input or output will make users more convenient when they use machine translators.

Appendix A

List of acronyms

No.	Acronyms	Detail
1	ALBERT	A Lite version of the BERT model
2	BBPE	Byte-level Byte Pair Encoding
3	BERT	Bi-direction Encoder Representation from Transformer
4	biLM	bidirectional language model
5	BLEU	Bi-Lingual Evaluation Understudy
6	BoW	Bag of Words
7	CBOW	Continuous Bag-of-Words model
8	CCG	Combinatory Categorical Grammar
9	CNN	Convolutional neural network
10	CoLA	Corpus of Linguistic Acceptability
11	ConvS2S	Convolutional sequence to sequence learning
12	CRFs	Conditional Random Fields
13	ELMO	Embeddings from Language Model
14	ELMo	Embeddings from Language Model
15	FFN	Feed Forward Network
16	GloVe	Global Vectors for Word Representation
17	GNMT	Gap Neural Machine Translation
18	GPT-3	Generative Pre-trained Transformer 3
19	HMMs	Hidden Markov Models
20	LSTM	Long short-term memory
21	MA	Morphological Analysis
22	M-BERT	Multilingual BERT
23	MLM	Masked Language Modelling

24	MNLI	Multi-Genre Natural Language Inference
25	MT	Machine Translation
26	NER	Named Entity Recognition
27	NLP	Natural Language Processing
28	NMT	Neural Machine Translation
29	NSP	Next Sentence Prediction
30	POS	Part-of-Speech
31	QNLI	Question Natural Language Inference
32	QQP	Quora Question Pairs
33	RBMT	Rule-based Machine Translation
34	ReLU	Rectified Linear Unit
35	RNN	Recurrent neural network
36	RoBERTa	Robustly Optimized BERT pre-training Approach
37	SMT	Statistical Machine Translation
38	SST-2	Stanford Sentiment Treebank
39	TER	Translation Error Rate
40	TF-IDF	Term Frequency–Inverse Document Frequency
41	UPC	Ulsan Parallel Corpora
42	VinAI	Vin Artificial Intelligence
43	WSD	Word Sense Disambiguation
44	WSeg.	Word Segmentation

Appendix B

Vietnamese POS Tag sets¹⁶

No.	Name	Description	No.	Name	Description
1	N	Noun	19	D	Dirty word
2	Np	Proper noun	20	X	Undetermined group
3	Nc	Classifier noun	21	FW	Foreign words
4	Nu	Noun indicating unit of measure, currency	22	CH	Symbol
5	Nux	Extended Unit Noun	23	Nb	Borrowed Noun
6	M	Numeral	24	Vb	Borrowed Verb
7	Mx	Extended Numeral	25	Ab	Borrowed Adjective
8	L	Quantifier	26	Ny	Abbreviation of Noun
9	V	Verb	27	Vy	Abbreviation of Verb
10	A	Adjective	28	Ay	Abbreviation of Adjective
11	P	Pronoun	29	Npy	Abbreviation of Proper Noun
12	R	Adverb	30	Cy	Abbreviation of Conjunction
13	E	Preposition	31	Dy	Abbreviation of Dirty word
14	C	Conjunction	32	My	Abbreviation of Numeral
15	T	Auxiliary	33	Py	Abbreviation of Pronoun
16	I	Interjection	34	Ry	Abbreviation of Adverb
17	G	Group	35	Gy	Abbreviation of Group
18	MW	Multi-word expression	36	Nby	Abbreviation of Borrowed Noun

¹⁶ <https://viettelgroup.ai/document/part-of-speech-tagging>

Bibliography

1. Koehn, P.; Hoang, H.; Birch, A.; Callison-Burch, C.; Federico, M.; Bertoldi, N.; Cowan, B.; Shen, W.; Moran, C.; Zens, R.; et al. Moses: Open Source Toolkit for Statistical Machine Translation. In Proceedings of the Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions; Association for Computational Linguistics: Prague, Czech Republic, June 2007; pp. 177–180.
2. Dyer, C.; Lopez, A.; Ganitkevitch, J.; Weese, J.; Ture, F.; Blunsom, P.; Setiawan, H.; Eidelman, V.; Resnik, P. Cdec: A Decoder, Alignment, and Learning Framework for Finite-State and Context-Free Translation Models. In Proceedings of the Proceedings of the ACL 2010 System Demonstrations; Association for Computational Linguistics: Uppsala, Sweden, July 2010; pp. 7–12.
3. Na, S.-H.; Kim, Y.-K. Phrase-Based Statistical Model for Korean Morpheme Segmentation and POS Tagging. *IEICE Trans. Inf. & Syst.* **2018**, *E101.D*, 512–522, doi:10.1587/transinf.2017EDP7085.
4. Klein, G.; Kim, Y.; Deng, Y.; Senellart, J.; Rush, A. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In Proceedings of the Proceedings of ACL 2017, System Demonstrations; Association for Computational Linguistics: Vancouver, Canada, July 2017; pp. 67–72.
5. Sennrich, R.; Firat, O.; Cho, K.; Birch, A.; Haddow, B.; Hitschler, J.; Junczys-Dowmunt, M.; Läubli, S.; Miceli Barone, A.V.; Mokry, J.; et al. Nematus: A Toolkit for Neural Machine Translation. In Proceedings of the Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics; Association for Computational Linguistics: Valencia, Spain, April 2017; pp. 65–68.
6. Vaswani, A.; Bengio, S.; Brevdo, E.; Chollet, F.; Gomez, A.; Gouws, S.; Jones, L.; Kaiser, \Lukasz; Kalchbrenner, N.; Parmar, N.; et al. Tensor2Tensor for Neural Machine Translation. In Proceedings of the Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track); Association for Machine Translation in the Americas: Boston, MA, March 2018; pp. 193–199.
7. Cho, S.W.; Lee, E.-H.; Lee, J.-H. Phrase-Level Grouping for Lexical Gap Resolution in Korean-Vietnamese SMT. In Proceedings of the Computational Linguistics; K.

- Hasida and W. P. Pa, eds. Springer Singapore: Singapore, 2018; Vol. 781, pp. 127–136.
8. Nguyen, Q.-P.; Shin, J.-C.; Ock, C.-Y. Korean Morphological Analysis for Korean-Vietnamese Statistical Machine Translation. *dzkjdxbywb* **2017**, *15*, 413–419, doi:10.11989/JEST.1674-862X.61005104.
 9. Cho, S.-W.; Kim, Y.-G.; Kwon, H.-S.; Lee, E.-H.; Lee, W.-K.; Cho, H.-M. Embedded Clause Extraction and Restoration for the Performance Enhancement in Korean-Vietnamese Statistical Machine Translation. In Proceedings of the Proceedings of the 28th Annual Conference on Human & Cognitive Language Technology; Busan, Korea, October 2016; pp. 280–284.
 10. Chung, T.; Gildea, D. Unsupervised Tokenization for Machine Translation. In Proceedings of the Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Singapore, August 2009; pp. 718–726.
 11. Kim, H. Korean National Corpus in the 21st Century Sejong Project. *Proceedings of the 13th NIJL International Symposium* **2006**.
 12. Tan, L.; Bond, F. Building and Annotating the Linguistically Diverse NTU-MC (NTU-Multilingual Corpus). In Proceedings of the Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation; Institute of Digital Enhancement of Cognitive Processing, Waseda University: Singapore, December 2011; pp. 362–371.
 13. Ott, M.; Edunov, S.; Baeovski, A.; Fan, A.; Gross, S.; Ng, N.; Grangier, D.; Auli, M. Fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In Proceedings of the Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations); Association for Computational Linguistics: Minneapolis, Minnesota, June 2019; pp. 48–53.
 14. Modrzejewski, M.; Exel, M.; Buschbeck, B.; Ha, T.-L.; Waibel, A. Incorporating External Annotation to Improve Named Entity Translation in NMT. In Proceedings of the Proceedings of the 22nd Annual Conference of the European Association for Machine Translation; European Association for Machine Translation: Lisboa, Portugal, November 2020; pp. 45–51.
 15. Generalizing Back-Translation in Neural Machine Translation - ACL Anthology Available online: <https://aclanthology.org/W19-5205/> (accessed on 27 March 2022).

16. Johnson, M.; Schuster, M.; Le, Q.V.; Krikun, M.; Wu, Y.; Chen, Z.; Thorat, N.; Viégas, F.; Wattenberg, M.; Corrado, G.; et al. Google’s Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics* **2017**, *5*, 339–351, doi:10.1162/tacl_a_00065.
17. Bojar, O.; Federmann, C.; Fishel, M.; Graham, Y.; Haddow, B.; Koehn, P.; Monz, C. Findings of the 2018 Conference on Machine Translation (WMT18). In Proceedings of the Proceedings of the Third Conference on Machine Translation: Shared Task Papers; Association for Computational Linguistics: Belgium, Brussels, October 2018; pp. 272–303.
18. Östling, R.; Tiedemann, J. Neural Machine Translation for Low-Resource Languages. *arXiv:1708.05729 [cs]* **2017**.
19. Tong, A.; Diduch, L.; Fiscus, J.; Haghpanah, Y.; Huang, S.; Joy, D.; Peterson, K.; Soboroff, I. Overview of the NIST 2016 LoReHLT Evaluation. *Machine Translation* **2018**, *32*, 11–30, doi:10.1007/s10590-017-9200-8.
20. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv:1609.08144 [cs]* **2016**.
21. Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; Dauphin, Y.N. Convolutional Sequence to Sequence Learning. In Proceedings of the Proceedings of the 34th International Conference on Machine Learning; PMLR, July 17 2017; pp. 1243–1252.
22. Huang, J.-X.; Lee, K.-S.; Kim, Y.-K. Hybrid Translation with Classification: Revisiting Rule-Based and Neural Machine Translation. *Electronics* **2020**, *9*, 201, doi:10.3390/electronics9020201.
23. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In Proceedings of the Proceedings of the 31st International Conference on Neural Information Processing Systems; Curran Associates Inc.: Red Hook, NY, USA, December 4 2017; pp. 6000–6010.
24. Voita, E.; Serdyukov, P.; Sennrich, R.; Titov, I. Context-Aware Neural Machine Translation Learns Anaphora Resolution. In Proceedings of the Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Association for Computational Linguistics: Melbourne, Australia, July 2018; pp. 1264–1274.

25. Guo, Z.; Nguyen, M.L. Document-Level Neural Machine Translation Using BERT as Context Encoder. In Proceedings of the Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop; Association for Computational Linguistics: Suzhou, China, December 2020; pp. 101–107.
26. Vu, V.-H.; Nguyen, Q.-P.; Shin, J.-C.; Ock, C.-Y. UPC: An Open Word-Sense Annotated Parallel Corpora for Machine Translation Study. *Applied Sciences* **2020**, *10*, 3904, doi:10.3390/app10113904.
27. Ballier, N.; Amari, N.; Merat, L.; Yunès, J.-B. The Learnability of the Annotated Input in NMT Replicating (Vanmassenhove and Way, 2018) with OpenNMT. In Proceedings of the LREC; Calzolari, N., Ed.; ELRA: Marseille, France, May 2020; pp. 5631–5640.
28. Nguyen, Q.-P.; Vo, A.-D.; Shin, J.-C.; Tran, P.; Ock, C.-Y. Korean-Vietnamese Neural Machine Translation System With Korean Morphological Analysis and Word Sense Disambiguation. *IEEE Access* **2019**, *7*, 32602–32616, doi:10.1109/ACCESS.2019.2902270.
29. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers); Association for Computational Linguistics: Minneapolis, Minnesota, June 2019; pp. 4171–4186.
30. Zhu, J.; Xia, Y.; Wu, L.; He, D.; Qin, T.; Zhou, W.; Li, H.; Liu, T.-Y. Incorporating BERT into Neural Machine Translation. In Proceedings of the arXiv:2002.06823 [cs]; Addis Ababa, Ethiopia, April 26 2020.
31. Nguyen, D.Q.; Tuan Nguyen, A. PhoBERT: Pre-Trained Language Models for Vietnamese. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020; Association for Computational Linguistics: Online, November 2020; pp. 1037–1042.
32. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations. *arXiv:1909.11942 [cs]* **2020**.

33. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *undefined* **2019**.
34. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. SpanBERT: Improving Pre-Training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics* **2020**, 8, 64–77, doi:10.1162/tacl_a_00300.
35. Le, H.; Vial, L.; Frej, J.; Segonne, V.; Coavoux, M.; Lecouteux, B.; Allauzen, A.; Crabbé, B.; Besacier, L.; Schwab, D. FlauBERT: Unsupervised Language Model Pre-Training for French. In Proceedings of the Proceedings of the 12th Language Resources and Evaluation Conference; European Language Resources Association: Marseille, France, May 2020; pp. 2479–2490.
36. Scheible, R.; Thomczyk, F.; Tippmann, P.; Jaravine, V.; Boeker, M. GottBERT: A Pure German Language Model. *arXiv:2012.02110 [cs]* **2020**.
37. Cui, Y.; Che, W.; Liu, T.; Qin, B.; Wang, S.; Hu, G. Revisiting Pre-Trained Models for Chinese Natural Language Processing. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020; Association for Computational Linguistics: Online, November 2020; pp. 657–668.
38. Lee, S.; Jang, H.; Baik, Y.; Park, S.; Shin, H. KR-BERT: A Small-Scale Korean-Specific Language Model. *arXiv:2008.03979 [cs]* **2020**.
39. Clinchant, S.; Jung, K.W.; Nikoulina, V. On the Use of BERT for Neural Machine Translation. In Proceedings of the Proceedings of the 3rd Workshop on Neural Generation and Translation; Association for Computational Linguistics: Hong Kong, November 2019; pp. 108–117.
40. Koehn, P. Europarl: A Parallel Corpus for Statistical Machine Translation. In Proceedings of the Proceedings of Machine Translation Summit X: Papers; Phuket, Thailand, September 13 2005; pp. 79–86.
41. Steinberger, R.; Pouliquen, B.; Widiger, A.; Ignat, C.; Erjavec, T.; Tufiş, D.; Varga, D. The JRC-Acquis: A Multilingual Aligned Parallel Corpus with 20+ Languages. In Proceedings of the Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06); European Language Resources Association (ELRA): Genoa, Italy, May 2006.

42. Rafalovitch, A.; Dale, R. United Nations General Assembly Resolutions: A Six-Language Parallel Corpus. In Proceedings of the Proceedings of Machine Translation Summit XII: Posters; Ottawa, Canada, August 26 2009.
43. Lee, J.; Lee, D.; Lee, G. Improving Phrase-Based Korean-English Statistical Machine Translation. In Proceedings of the Ninth International Conference on Spoken Language Processing; Pittsburgh, PA, USA, January 1 2006.
44. Hong, G.; Lee, S.-W.; Rim, H.-C. Bridging Morpho-Syntactic Gap between Source and Target Sentences for English-Korean Statistical Machine Translation. In Proceedings of the Proceedings of the ACL-IJCNLP 2009 Conference Short Papers; Association for Computational Linguistics: Suntec, Singapore, August 2009; pp. 233–236.
45. Park, J.; Hong, J.-P.; Cha, J.-W. Korean Language Resources for Everyone. In Proceedings of the Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation: Oral Papers; Seoul, South Korea, October 2016; pp. 49–58.
46. Tiedemann, J. OPUS – Parallel Corpora for Everyone. In Proceedings of the Proceedings of the 19th Annual Conference of the European Association for Machine Translation: Projects/Products; Baltic Journal of Modern Computing: Riga, Latvia, May 30 2016.
47. Dien, D.; W.J, K. Exploiting the Korean—Vietnamese Parallel Corpus in Teaching Vietnamese for Koreans. In Proceedings of the Proceedings of the Interdisciplinary Study on Language Communication in Multicultural Society, the Int'l Conf. of ISEAS/BUFS; Busan, Korea, May 25 2017.
48. Pennington, J.; Socher, R.; Manning, C. GloVe: Global Vectors for Word Representation. In Proceedings of the Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP); Association for Computational Linguistics: Doha, Qatar, October 2014; pp. 1532–1543.
49. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to Sequence Learning with Neural Networks. In Proceedings of the Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2; MIT Press: Cambridge, MA, USA, December 8 2014; pp. 3104–3112.
50. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations Using RNN Encoder–Decoder for

- Statistical Machine Translation. In Proceedings of the Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP); Association for Computational Linguistics: Doha, Qatar, October 2014; pp. 1724–1734.
51. Ravichandiran, S. *Getting Started with Google BERT*; Packt Publishing Ltd: Birmingham B3 2PB, UK, 2021; ISBN 978-1-83882-159-3.
 52. MUHAMMAD ALI AL, K. *A Dictionary of Theoretical Linguistics (English-Arabic with an Arabic English Glossary)*; LIBARAAIRIE DU LIBAN: LIBAN, 2008; ISBN 10: 9953865744.
 53. Liu, X.; He, P.; Chen, W.; Gao, J. Multi-Task Deep Neural Networks for Natural Language Understanding. In Proceedings of the Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Florence, Italy, July 2019; pp. 4487–4496.
 54. Lafferty, J.D.; McCallum, A.; Pereira, F.C.N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In Proceedings of the Proceedings of the Eighteenth International Conference on Machine Learning; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, June 28 2001; pp. 282–289.
 55. Krogh, A. Hidden Markov Models for Labeled Sequences. In Proceedings of the Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C: Signal Processing (Cat. No.94CH3440-5); October 1994; Vol. 2, pp. 140–144 vol.2.
 56. Blunsom, P. Maximum Entropy Markov Models for Semantic Role Labelling. In Proceedings of the Proceedings of the Australasian Language Technology Workshop 2004; Sydney, Australia, December 2004; pp. 109–116.
 57. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. Neural Architectures for Named Entity Recognition. In Proceedings of the Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; Association for Computational Linguistics: San Diego, California, June 2016; pp. 260–270.
 58. Dos Santos, C.; Guimarães, V. Boosting Named Entity Recognition with Neural Character Embeddings. In Proceedings of the Proceedings of the Fifth Named Entity

Workshop; Association for Computational Linguistics: Beijing, China, July 2015; pp. 25–33.

59. Zhou, J.; Xu, W. End-to-End Learning of Semantic Role Labeling Using Recurrent Neural Networks. In Proceedings of the Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers); Association for Computational Linguistics: Beijing, China, July 2015; pp. 1127–1137.
60. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780, doi:10.1162/neco.1997.9.8.1735.
61. Matteson, A.; Lee, C.; Kim, Y.; Lim, H. Rich Character-Level Information for Korean Morphological Analysis and Part-of-Speech Tagging. In Proceedings of the Proceedings of the 27th International Conference on Computational Linguistics; Association for Computational Linguistics: Santa Fe, New Mexico, USA, August 2018; pp. 2482–2492.
62. Min, Jihong; Jeon, Joon-Woo; Song, Kwang-Ho; Kim, Yoo-Sung A Study on Word Sense Disambiguation Using Bidirectional Recurrent Neural Network for Korean Language. *Journal of the Korea Society of Computer and Information* **2017**, *22*, 41–49, doi:10.9708/JKSCI.2017.22.04.041.
63. Kang, M.; Kim, B.; Lee, J.S. Word Sense Disambiguation Using Embedded Word Space. *Journal of Computing Science and Engineering* **2017**, *11*, 32–38, doi:10.5626/JCSE.2017.11.1.32.
64. Nguyen, D.Q.; Nguyen, D.Q.; Vu, T.; Dras, M.; Johnson, M. A Fast and Accurate Vietnamese Word Segmenter. In Proceedings of the Proc. 7th International Conference on Language Resources and Evaluation; Miyazaki, Japan, May 2018; p. 6.
65. Nguyen, A.-D.; Nguyen, K.-H.; Ngo, V.-V. Neural Sequence Labeling for Vietnamese POS Tagging and NER. *arXiv:1811.03754 [cs]* **2018**.
66. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. In Proceedings of the Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Association for Computational Linguistics: Berlin, Germany, August 2016; pp. 1715–1725.

67. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. Bleu: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Philadelphia, Pennsylvania, USA, July 2002; pp. 311–318.
68. Snover, M.; Dorr, B.; Schwartz, R.; Micciulla, L.; Makhoul, J. A Study of Translation Edit Rate with Targeted Human Annotation. In Proceedings of the Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers; Association for Machine Translation in the Americas: Cambridge, Massachusetts, USA, August 8 2006; pp. 223–231.