Saarland University

Charles University

Master Thesis

Indonesian-English Neural Machine Translation

submitted in fulfillment of the degree requirements of the

MSc in Language Science and Technology at **Saarland University**

as part of the

European Masters Program in Language and Communication Technologies

Author:
Meisyarah Dwiastuti

Supervisors:
Mgr. Martin Popel, Ph.D.
Prof. Dr. Josef van Genabith

2019

Eidesstattliche Erklärung
Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Ich versichere, dass die gedruckte und die elektronische Version der Masterarbeit inhaltlich übereinstimmen.

Declaration
I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged. I assure that the electronic version is identical in content to the printed version of the Master's thesis.

Saarbrücken, 16th December 2019 signature of the author

Alhamdulillahirabbil'alamin.

I would like to thank my supervisor at Charles University, Martin Popel, for his support and guidance since the beginning of the work until the very end. I would also like to thank my supervisor at Saarland University, Josef van Genabith, for his valuable feedback and recommendations of related work to my thesis.

I am very grateful to the European Union who has granted me Erasmus Mundus scholarship to study with European Master Program in Language and Communication Technologies (LCT). I would like to thank Bobbye Pernice, Markéta Lopatková, Kristýna Kysilková, and Jürgen Trouvain for their help in important matters during my study. I would also like to express my appreciation to all the professors, fellow students, and colleagues whom I encountered during this Master program.

This thesis would not have been possible without the support and unconditional love from my family. Special thanks to Rifqi, Iza, and Ocha, whom I mostly talked about my thesis to, and the Indonesian community in Prague who has made the living-abroad experience feel more like at home.

Title: Indonesian-English Neural Machine Translation

Author: Meisyarah Dwiastuti

Supervisors:
Mgr. Martin Popel, Ph.D., Charles University
Prof. Dr. Josef van Genabith, Saarland University

Abstract: In this thesis, we conduct a study on neural machine translation (NMT) for an under-studied language, Indonesian, specifically for English-Indonesian (EN-ID) and Indonesian-English (ID-EN) in a low-resource domain, TED talks. Our goal is to implement domain adaptation methods to improve the low-resource EN-ID and ID-EN NMT systems. First, we implement model fine-tuning method for EN-ID and ID-EN NMT systems by leveraging a large parallel corpus containing movie subtitles. Our analysis shows the benefit of this method for the improvement of both systems. Second, we improve our ID-EN NMT system by leveraging English monolingual corpora through back-translation. Our back-translation experiments focus on how to incorporate the back-translated monolingual corpora to the training set, in which we investigate various existing training regimes and introduce a novel *4-way-concat* training regime. We also analyze the effect of fine-tuning our back-translation models with different scenarios. Experimental results show that our method of implementing back-translation followed by model fine-tuning makes an improvement in our ID-EN NMT systems by up to 2.3 BLEU points over a system without back-translation. Our ID-EN NMT systems show a comparable performance with Google Translate on WIT^3 TED talks tst2015-6 and tst2017plus test sets.

Keywords: neural machine translation, domain adaptation, back-translation

Contents

Introduction

Neural machine translation (NMT) has been lately considered as the state-of-the-art method in machine translation (MT) [Bojar et al., 2018]. As a data-driven method, the size of the parallel data used to train an NMT system has a big influence on its outstanding performance. This method performs well on the high-resource language pairs (e.g. English-French, English-German), but still struggles on the low-resource ones [Koehn and Knowles, 2017]. However, the context of *low-resource* itself depends on the domain of the MT systems we build. The translation generated by data-driven MT systems relies on the data used for training them. While domain-specific corpora are usually scarce, there might be large out-of-domain parallel corpora or monolingual corpora in the source or target language. One can leverage those corpora to deal with in-domain data scarcity through *domain adaptation* methods [Chu and Wang, 2018].

The languages we focus on are Indonesian (ID) and English (EN). Despite the huge number of Indonesian speakers (more than 200 millions), research on Indonesian NMT is still lack of interest, even towards the heavily researched language like English. We suspect one of the reasons is the limited good-quality resource (i.e. parallel data), as pointed by Trieu et al. [2017] and Adiputra and Arase [2017] who build Indonesian-Vietnamese and Japanese-Indonesian NMTs, respectively. While English-Indonesian (EN-ID) NMT has been implemented by Lakew et al. [2018] for a study on language variety, we could not find any work related to Indonesian-English (ID-EN) NMT, which motivates us to conduct this study.

Our motivation to build an ID-EN NMT system is not only because of the lack of works concerning this topic, but also two factors that give us a chance to improve the low-resource ID-EN NMT. The first is the recent release of a huge parallel corpus, OpenSubtitles2018, containing more than 9 million parallel sentences in spoken language domain [Lison et al., 2018]. While the other ID-EN parallel corpora have different domains and much smaller sizes, we think of leveraging this huge dataset for some domain-specific NMT using a domain adaptation method. The second factor is that we can take advantage of the resource richness of English as the target language – there are a couple of large English monolingual corpora available. Monolingual corpora have been shown beneficial to improve NMT performance [Sennrich et al., 2016].

Thesis Statement

The main goal of this thesis is to improve the low-resource ID-EN NMT by leveraging the available out-of-domain parallel corpora and large target monolingual corpora through domain adaptation methods. We focus on spoken language domains because of the amount of available data we have. More specifically, we build ID-EN NMT systems for speech-styled language, i.e. TED talk[1] domain.

In order to reach our goal, we build our ID-EN NMT systems using the Transformer model [Vaswani et al., 2017], which is considered as the state-of-the-art

[1] https://www.ted.com/

NMT model for many language pairs [Bojar et al., 2018], and conduct experiments applying domain adaptation methods. The objective of our experiments is to study the effect of the domain adaptation methods on the translation quality evaluated by automatic evaluation. The objective can be elaborated into the following experiments:

1. We leverage a large out-of-domain parallel data using model fine-tuning [Luong and Manning, 2015] to improve low-resource EN-ID and ID-EN NMT systems.

2. We leverage English monolingual corpora to improve ID-EN NMT system through back-translation method [Sennrich et al., 2016]. In this experiment, we also:

    - try different approaches to incorporate back-translated (*synthetic*) data to train our ID-EN NMT and introduce a novel one, *4-way-concat*.
    - study the effect of fine-tuning for the pre-trained back-translation systems using the synthetic data.

Outline

The remainder of this thesis is organized as follows:

- **Chapter 1** describes background knowledge needed to understand the approaches used in this thesis and related works.

- **Chapter 2** describes the setups for experiments conducted within this thesis, consisting of data preparation, NMT model architecture and general training setups.

- **Chapter 3** describes the setups, results and analysis for domain adaptation experiment using model fine-tuning.

- **Chapter 4** describes the setups, results and analysis for domain adaptation through back-translation.

- **Chapter 5** describes the final evaluation of our NMT systems on unseen data.

- **Conclusion** summarizes our work and presents the possibilities for future research.

Published Work

This thesis is the revised version of the thesis that the author has submitted at Charles University and defended on September 9th, 2019:

- Meisyarah Dwiastuti. 2019. Indonesian-English Neural Machine Translation. Univerzita Karlova, Matematicko-fyzikální fakulta. Prague, Czech Republic.

A part of this thesis draws from the following work by its author:

- Meisyarah Dwiastuti. 2019. English-Indonesian Neural Machine Translation for Spoken Language Domains. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, Florence, Italy.

1. Background and Related Work

In this chapter, we provide background information on Indonesian, methods in machine translation, the Transformer model, as well as the domain adaptation methods that will be used in our experiments.

1.1 Indonesian Language

Indonesian or *Bahasa Indonesia* belongs to the Austronesian language family. While there are hundreds of languages spoken in Indonesia, Indonesian refers to the official national language of Indonesia. Indonesian's writing system uses the Latin alphabet without any diacritics, similarly to English. While the typical word order in Indonesian is Subject-Verb-Object, it also allows different orders. Grammatically, the language does not make use of any case nor gender and the tenses do not change the form of the verbs. Most of the word constructions are derivational morphology, whose complexity includes affixation, clitics, and reduplication.

The use of the vocabulary relates to the context when the speakers use the language. It is more obvious in the spoken language, for example, the use of honorifics or more formal terms when speaking to older people or in a formal situation. In Indonesian, honorifics are simply found as pronouns, for either the first, second, or third person. For example, ”I” in English can be translated to ”*saya*” (more formal) or ”*aku*” (less formal).

1.2 Machine Translation

Machine translation (MT) is an automatic system for translating sentences from a natural language (source) to another one (target). In this section, we briefly explain some MT methods and works related to the method, focusing on the most prominent ones and the ones on ID-EN or EN-ID pairs.

1.2.1 Rule-based and statistical methods

Two common methods used for MT systems before the rise of NMT is rule-based MT (RBMT) and statistical MT (SMT). RBMT makes use of sets of rules to translate the source language into the target language. The rules involve linguistic information of both source and target languages, such as lexicon, morphology, and other syntactic and semantic analysis. The variety of RBMT is big. Some of the well-known RBMT systems are Systran [Toma, 1977] and Apertium [Forcada et al., 2011]. Related to EN-ID RBMT, Adji [2010] combines direct and transfer approach using Annotated Disjunct based on Link Grammar formalism.

While RBMT focuses on the process of translation, such as how to represent the word, how to translate it, and so on, SMT focuses on the output, meaning the system directly tries to model what the most likely translation of a given sentence is. It can be done by feeding the model with a lot of translation examples, i.e. bilingual or parallel corpus. As it has seen a lot of examples many times, it will learn how to align the words between the two languages. This ability is called a translation model. For fluency of the translation output, SMT also contains a language model (LM) of the target language. While both translation and language models may produce many possible target sentences, the SMT selects the most probable one. In phrase-based SMT, those models along with other features like phrase penalty, reordering, distortion, etc, are used as features to train a log-linear model.

Phrase-based SMT was once the state-of-the-art method until NMT has shown a promising result [Jean et al., 2015b, Luong and Manning, 2015]. Moses [Koehn et al., 2007] is a popular framework for phrase-based SMT. Several works on ID-EN and EN-ID SMT, also using Moses, tried to include some linguistic information to improve the performance, such as morphological information [Larasati, 2012a], word-level similarity [Larasati, 2012b], or part-of-speech tag information [Sujaini et al., 2014]. While neural networks were used in conjunction with EN-ID SMT, e.g. to replace the statistical LM in SMT with neural LM [Hermanto et al., 2015], NMT refers to learning method using a single end-to-end neural network architecture.

1.2.2 Neural machine translation

The use of neural networks for MT has started decades ago [Castaño et al., 1997, Neco and Forcada, 1997] before a temporary break until the computation resources (i.e. GPUs) were capable of training larger models, i.e. deep neural networks. Deep neural networks are known for their ability to learn continuous representations from their input as a fixed-size vector, which came out to be useful in MT [Kalchbrenner and Blunsom, 2013, Cho et al., 2014].

The architecture used for NMT is encoder-decoder which consists of an encoder and a decoder. The encoder transforms the input sentence into a vector, which is usually also known as latent representation. The decoder makes use of this representation to generate the target sentence. Recurrent neural network (RNN) has been commonly used in this kind of architecture [Sutskever et al., 2014, Bahdanau et al., 2015, Jean et al., 2015b, Luong and Manning, 2015] using its gated cell variants, such as Long Short-term Memory (LSTM) units [Hochreiter and Schmidhuber, 1997] or Gated Recurrent Units (GRU) [Cho et al., 2014]. While convolutional based NMT has proven similar translation quality to RNN-based NMT with much faster computation time [Gehring et al., 2017], the current state-of-the-art model, the Transformer model [Vaswani et al., 2017] does not use any of them and instead has shown a superior result using *self-attention*. Lakew et al. [2018] use the Transformer model to build an EN-ID NMT for a study on language variety. However, we are not aware of any works using the model for ID-EN NMT prior to the beginning of this thesis.

In this thesis, we use the Transformer model. While we will briefly explain the overview of NMT architecture and learning in the rest of this subsection, we will present the general architecture of the Transformer model in Section 1.2.3.

Encoder-decoder architecture

NMT systems utilize an encoder-decoder architecture. The encoder receives the input tokens from the source sentence and maps them into a latent representation. The input tokens are usually represented as one-hot vectors which are then associated with some continuous representation (embedding) using an embedding weight matrix. More formally, if $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ contains one-hot vectors $\mathbf{x}_i$ of the input sentence with length n, then $\mathbf{e} = (\mathbf{e}_1, \ldots, \mathbf{e}_n)$ contains the continuous representation of the tokens, in which $\mathbf{e}_i = \mathbf{W}^e \cdot \mathbf{x}_i$ and $\mathbf{W}^e$ is the embedding weight matrix. The continuous representations are then fed to a neural network architecture, i.e. the encoder, which maps them into a latent representation $\mathbf{z} = (z_1, \ldots, z_k)$. This latent representation is considered to hold the information from the input sentence.

Given the latent representation $\mathbf{z}$, the task of the decoder is to generate the target sentence $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_m)$ one element at a time. It also utilizes a neural network architecture which works as a language model in an auto-regressive way. It means that when generating the subsequent token, the model also takes the output at the current time step as an input. However, during training, since we have the gold reference of target sentence, at time step t the decoder takes the gold output of time step $t-1$ instead of taking the predicted output by the model. This approach is called *teacher-forcing training.*

The predicted token $\hat{\mathbf{y}}_t$ is obtained by, firstly, transforming the hidden representation in the decoder $\mathbf{s}_t$ to a vector of the same size of the vocabulary using some function g, as shown in Equation 1.1. This vector is supposed to represent the scores of the similarity between $\mathbf{s}_t$ to the tokens in the vocabulary. Secondly, one can represent it as a probability vector by applying softmax function over the tokens in the vocabulary, as shown in Equation 1.2. The predicted token at a time t is the one with the highest probability.

$$\mathbf{v}_t = g(\mathbf{s}_t), \mathbf{v}_t \in \mathbb{R}^{|V|} \tag{1.1}$$

$$\hat{\mathbf{y}}_t = \text{softmax}(\mathbf{v}_t) \tag{1.2}$$

$$softmax(\mathbf{v}_t)_i = \frac{v_{ti}}{\sum_j v_{tj}} \tag{1.3}$$

Attention mechanism

We know that the encoder needs to summarize all important information from the source sentence in a latent representation. In the classic RNN-based encoder-decoder architecture, this leads to an issue when the sentence is too long, especially when translating sentences that are longer than any sentence in the training examples. The model struggles with a long-term dependency as it has to compress all information from the whole source sequence in a fixed-size vector.

While the basic decoder makes use of the summary in that vector to generate the output at each time step, Bahdanau et al. [2015] proposed to use the information from some specific part of the source sentence. At each decoding time step, the decoder is allowed to look at the encoding of the input tokens at each time step and *attend* to some specific part of the input which the decoder thinks is important based on some attention score. This approach is known as *attention mechanism.* Not only that it improves the performance of NMT systems on translating long sentences, but the way the attention score is computed makes it also serve as a soft-alignment between the source and target sentences.

Suppose that the decoder wants to generate a text $\mathbf{y}_i$ at time step i. The attention mechanism works as follows:

- We compute $e_{ij} = a(\mathbf{s}_{i-1}, \mathbf{h}_j)$, which is the alignment score of the output at position i with the input at any position j by using some scoring function a on the hidden representation of the decoder $\mathbf{s}$ and the encoder $\mathbf{h}$.
- We compute the attention weight $\alpha_{ij} = \text{softmax}(\mathbf{e}_i)_j = \dfrac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$, representing how much the output at position i should attend to the input at position j.
- We compute the context vector $\mathbf{c}_i = \sum_j \alpha_{ij}\mathbf{h}_j$.
- The context vector is then used as additional input to the hidden representation at time step i, such that $\mathbf{s}_i = f(\mathbf{s}_{i-1}, \mathbf{y}_{i-1}, \mathbf{c}_i)$.

Handling the Out-of-Vocabulary Problem

Conventional NMT systems use words as the tokens for the source and target sentences. Since translation is an open-vocabulary problem, the words can be variously many. However in practice the vocabulary size of NMT systems is typically limited to 30,000 – 50,000 words [Sennrich et al., 2016] which leaves the unseen words to be marked with the unknown symbol `<unk>`. There are some solutions to handle this Out-of-Vocabulary (OOV) problem, such as to use a dictionary look-up [Jean et al., 2015a, Luong et al., 2015], characters as tokens [Ling et al., 2015], character to word representation [Lee et al., 2016], and a hybrid of characters and word representations [Luong and Manning, 2016].

Today's common practice is to use *subword* units (or just subwords) as the tokens for NMT [Sennrich et al., 2016]. The advantage of using subwords is that the vocabulary keeps the most frequent subwords in various length, so the rare words can be represented as a sequence of subwords. The subword vocabulary is trained on the training data using some word segmentation algorithm [Sennrich et al., 2016, Macháček et al., 2018, Kudo, 2018].

Training

Since the whole architecture presents an end-to-end model, an NMT model can be trained using backpropagation to optimize *cross-entropy loss* (also known as *negative log-likelihood loss*) of the softmax layer on the decoder, which can be formally written as follows:

$$NLL(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{j}^{|V|} y_j \log \hat{y}_j = -\log \hat{y}_{gold} \tag{1.4}$$

where $\mathbf{y}$ is the one-hot vector of the gold token, $\hat{\mathbf{y}}$ is the probability vector from the softmax layer, and $|V|$ is the size of the vocabulary. Since the element of the one-hot vector is all zeros except one element with value of one whose index representing the token index, the summation is the same as only computing the logarithm of the probability at the gold reference index.

The optimization can use any gradient-based method, in which it computes the negative gradient of the loss function evaluated on a mini-batch and uses as the direction of the optimization steps (training steps). As pointed by Shazeer and Stern [2018], *adaptive* gradient-based methods have been empirically outperforming conventional stochastic gradient descent across a variety of domains. During their parameter updates, these methods divide the gradient by the square root of a vector summarizing the history of squared gradients. They require more memory capacity to hold the history vector whose length is equal to the number of the model parameters. For example, one of the commonly used methods, Adam [Kingma and Ba, 2015], has two additional values for each model parameter. It means the memory requirement of Adam is three times as large as the number of model parameters. Shazeer and Stern [2018] propose Adafactor, which they show to produce similar quality results to Adam on a common machine translation task but require less memory. The method is thus expected to allow training larger models on the machine with equal memory capacity.

Inference

During inference, the decoder generates a sequence of token with length m by selecting the sequence with the highest probability, which can be written as follows:

$$\begin{aligned}(\mathbf{y}_1, \dots, \mathbf{y}_m) &= \operatorname*{argmax}_{\mathbf{y}} \prod_{t=1}^{m} P(\mathbf{y}_t | \mathbf{x}, \mathbf{y}_1, \dots, \mathbf{y}_{t-1}) \\ &= \operatorname*{argmax}_{\mathbf{y}} \sum_{t=1}^{m} \log P(\mathbf{y}_t | \mathbf{x}, \mathbf{y}_1, \dots, \mathbf{y}_{t-1}) \\ &= \operatorname*{argmax}_{\mathbf{y}} \log P(\mathbf{y} | \mathbf{x})\end{aligned} \tag{1.5}$$

One approach to select the sequence is by using greedy search, in which at each time step t the decoder always picks the token resulted by the highest $P(\mathbf{y}_t | \mathbf{x}, \mathbf{y}_1, \dots, \mathbf{y}_{t-1})$. It means that it keeps only one hypothesis at a time step. On the other hand, beam search approach keeps k hypotheses at a time step, in which k is referred to the beam size. We start the decoding by selecting k best hypotheses, i.e. with the highest probability. For the next time step we will have $k.|V|$, and again only select the k best ones. The advantage of beam search over greedy search is that we have bigger search space to find the best sequence. While a big k tends to yield a better approximation of the best sequence, the computation also grows up in both time and space. Therefore, the selection of k should consider the computation cost.

Since output length m is not necessarily the same as the input sentence length n, in the training we preprocess the sentence to have a start of sentence symbol (`<sos>`) and an end of sentence symbol (`<eos>`), at the beginning and the end of the sentences. Thus, during inference the model should stop the decoding process once it generates the `<eos>` symbol.

In Equation 1.5, the multiplication of the probability values ($P(x) \in (0, 1)$) will result smaller value when the sentence is longer (also valid for the summation of logarithm of probabilities). This property will make the search algorithms (e.g. greedy search, beam search) tend to select shorter sentences. To handle this length variance, we normalize the probability of the sequence. One can instead use the formulas introduced by Wu et al. [2016] as follows:

$$\begin{aligned} s(\mathbf{y},\mathbf{x}) &= \log(P(\mathbf{y}|\mathbf{x}))/lp(\mathbf{y}) + cp(\mathbf{x};\mathbf{y}) \\ lp(\mathbf{y}) &= \frac{(5+|\mathbf{y}|)^{\alpha}}{(5+1)^{\alpha}} \\ cp(\mathbf{x};\mathbf{y}) &= \beta * \sum_{i=1}^{|\mathbf{x}|} \log(\min(\sum_{i=1}^{|\mathbf{y}|} p_{i,j}, 1.0)) \end{aligned} \tag{1.6}$$

where $p_{i,j}$ is the attention probability of the j-th target word $\mathbf{y}_j$ on the i-th source word $\mathbf{x}_i$. The formulas compute the score of the sequence $\mathbf{y}$ and $\mathbf{x}$ given the trained model, $s(\mathbf{y},\mathbf{x})$, by modifying the probability obtained from the beam search algorithm as shown in Equation 1.5. The modifications involve a length penalty (lp) and coverage penalty (cp) in which α and β are their parameters, respectively.

1.2.3 Transformer

The architecture of the Transformer [Vaswani et al., 2017] is also based on the encoder-decoder architecture. But instead of using RNN or convolutional neural network, it relies on *self-attention* mechanisms to compute the latent representation in both of its encoder and decoder. It does not only outperform the previously reported state-of-the-art models in terms of translation quality (reported in BLEU score), but also trains significantly faster than the recurrent- and convolution-based counterparts. In this subsection, we explain briefly the main properties of the Transformer.

Self-attention

In Subsection 1.2.2, we have described the attention mechanism at the decoder as an ability to look at the input representation at different position when processing the output at a position (time step). Self-attention mechanism works similarly. When encoding a token, it allows the model to look at the other token representations in the sentence in order to get a better representation of this token.

The attention mechanism in the Transformer introduces 3 abstractions for a token, namely *query*, *key*, and *value*. Each of them is a vector, in which query and key vectors have length d_k and value vector has length d_v. The attention function can be seen as a mapping of a *query* (from the token we are interested in) and a key-value pairs (from other tokens in the sequence) to result an output vector, which is the latent representation of the token at a position. The computation is performed as follows:

- Let $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ be the embedding of the input sequence with length n. We compute the query $\mathbf{q}_t$, key $\mathbf{k}_t$, and value $\mathbf{v}_t$ of each $\mathbf{x}_t$ by multiplying it with the corresponding weight matrices W^Q, W^K, W^V respectively.

- Suppose i is the position of the token we are interested in. We compute the alignment score e_{ij} as the dot product of the query q_i and key k_j divided by a scaling factor $\sqrt{d_k}$.
- We compute the attention weight $\alpha_{ij} = \dfrac{\exp(e_{ij})}{\sum_l \exp(e_{il})}$
- The output vector $\mathbf{o}_i$ is the sum of the weighted value vector $\mathbf{v}_j$ across any position j: $\mathbf{o}_i = \sum_j \alpha_{ij}\mathbf{v}_j$.

In practice, the computation is done as a matrix computation, such that the queries, keys, and values are packed together as matrices Q, K, and V respectively. Thus, the output matrix can be obtained as follows:

$$O = \text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1.7}$$

Multi-head attention

Vaswani et al. [2017] found it beneficial to have multiple attention functions run in parallel. This is done by having multiple sets of matrices Q, K, and V to project the input embedding. Suppose that the number is h (the authors used $h = 8$ in the original paper), thus the operation will result h output matrices. These matrices are then concatenated in the d_v axis and projected by a parameter matrix W^O to a matrix with the original attention output size. While the multi-head attention is considered to be able to learn better different representations for a token, the concatenation of the outputs summarize them in a single representation as the output of the attention layer.

Positional encoding

In order to keep track of the word order in the sequence, the Transformer adds a positional encoding to the input embedding of a token before feeding it to the encoder or the decoder. The size of this encoding is the same as the input embedding, denoted as d_{model}. While any encoding can be used to represent the position of the token in the sequence, the authors use sine and cosine functions as follows:

$$\text{PE}_{(pos,2i)} = sin\left(\frac{pos}{1000^{2i/d_{model}}}\right)$$

$$\text{PE}_{(pos,2i+1)} = cos\left(\frac{pos}{1000^{2i/d_{model}}}\right)$$

where pos is the absolute position of the token in the sequence and i is the index in the embedding.

Model architecture

The whole architecture is shown in Figure 1.1. The model consists of an encoder and a decoder. The encoder is composed of a stack of N layers ($N = 6$). Each

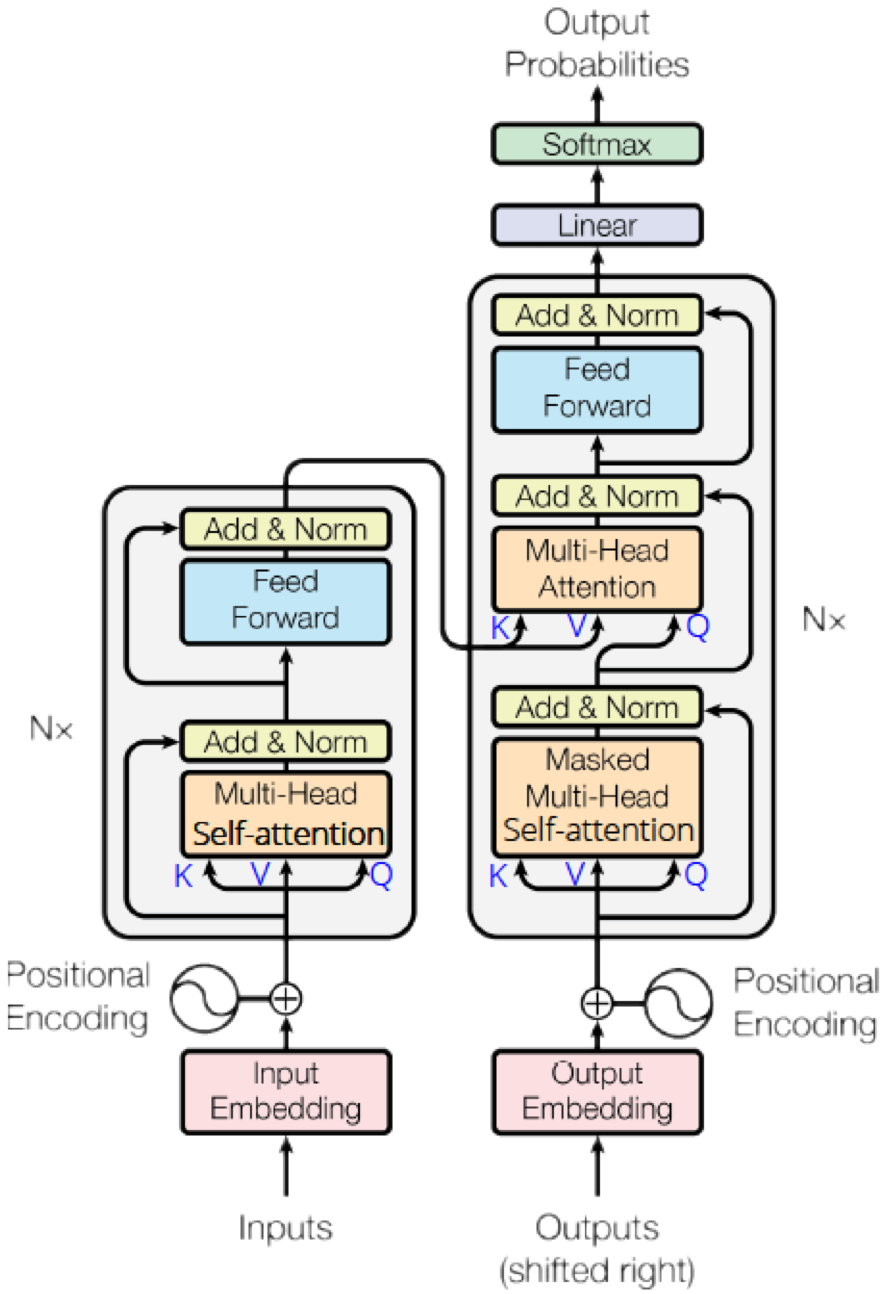


Figure 1.1: Model architecture of the Transformer, adopted from [Vaswani et al., 2017] with modification.

layer consists of 2 sublayers, namely a multi-head self-attention sublayer and a position-wise fully-connected sublayer. Around each sublayer there is a residual connection and a layer normalization. The model uses $d_{model} = 512$.

While the decoder is also composed of a stack of N layers, there is an additional sublayer in each layer. This sublayer is a multi-head attention over the output of the encoder stack, which performs similarly to the attention mechanism in the decoder of general NMT described in Subsection 1.2.2. In this case, while the queries are the output of the self-attention sublayer, the keys and values come from the output of the encoder stack. Moreover, in order to keep the auto-regressive property, the self-attention sublayer in the decoder is masked, such that each position in the decoder can only attend to the tokens up to that position. The stacks are followed by a linear projection and softmax layer, similar to a general NMT architecture.

1.2.4 Evaluation

There are two types of evaluation for MT, namely manual evaluation and automatic evaluation. In this thesis, we only conduct the evaluation using the automatic method. While there are many options for automatic metrics used for MT evaluation, we use BLEU [Papineni et al., 2002] since it is widely used in MT research.

The computation of BLEU score is based on two components, namely n-gram precision and brevity penalty. A simple way to compute the precision of an n-gram MT output candidate is to count up the number of overlapping n-grams in the candidate with the reference and divide by the total number of occurrence of all n-grams in the candidate (e.g. the number of words for unigrams). This becomes problematic if the candidate contains *overgenerated* n-grams that match the reference, which result a high precision but the sentence is not structurally closer to the reference, as shown in Example 1. Therefore Papineni et al. [2002] proposed the modified n-gram precision, in which one uses the *clipped count* of the n-gram and divides it by the real count of the n-gram in the candidate. The clipped count is obtained by, firstly, taking the maximum of the n-gram counts in any single reference (assuming there are more than one reference). Then one can take the minimum between this maximum reference count and the count in the candidate.

Example 1
Candidate: the the the the the the the
Reference: <u>the</u> mug is on <u>the</u> table
Simple unigram precision: 7/7
Modified unigram precision: 2/7

Therefore the modified n-grams precision for all candidates in the test corpus is computed as follows:

$$P_n = \frac{\sum_{C \in \{Candidates\}} \sum_{\text{n-gram} \in C} Count_{clip}(\text{n-gram})}{\sum_{C \in \{Candidates\}} \sum_{\text{n-gram} \in C} Count(\text{n-gram})} \tag{1.8}$$

While too long candidates get a penalty from the modified n-gram precision, BLEU metric applies a brevity penalty for candidates shorter than the reference as follows:

$$\mathrm{BP} = \begin{cases} 1, & \text{if } c > r \\ \exp(1 - r/c), & \text{if } c \leq r \end{cases} \tag{1.9}$$

where c is the length of the candidate and r is the effective reference corpus length.

Then, the BLEU score is computed as follows:

$$\mathrm{BLEU} = \mathrm{BP}.\exp(\sum_{n=1}^{N} w_n \log P_n) \tag{1.10}$$

where usually N is set to 4 and the weights w_n use uniform distribution $1/N$.

1.3 Domain Adaptation

While NMT systems need a lot of data to result in a high-quality translation, in-domain parallel corpora to train a domain-specific NMT system are usually scarce. In such a case, one can leverage large out-of-domain parallel corpora or monolingual corpora from either language side to train a good system, then adapt it using the small in-domain corpora. This approach is called *domain adaptation*.

Chu and Wang [2018] classify various domain adaptation methods into 2 classes, namely model-centric methods and data-centric. While model-centric methods utilize models trained on different domains, data-centric methods focus on the selection or generation of the domain-related data, In this thesis, we apply a model-centric method, namely model fine-tuning, and a data-centric method, namely back-translation.

1.3.1 Model fine-tuning

Domain adaptation using model fine-tuning is inspired by transfer learning method, where one can leverage a pre-trained model for the target task. The pre-trained model has been usually trained on a larger data, thus it is supposed to be able to give some general information for the translation. In the context of domain-adaptation in NMT, the model is first trained on a large out-of-domain parallel data, then it is fine-tuned on the in-domain data [Luong and Manning, 2015].

Since the in-domain data is usually small, the fine-tuning might result in a model that overfits on the in-domain data. In order to reduce the performance degradation on general-domain data after the model fine-tuning, a number of solutions have been proposed. Freitag and Al-Onaizan [2016] use an ensemble of the pre-trained and fine-tuned models. Chu et al. [2017] add a target domain tag to the sentences in the corpora and, firstly, train the model on out-of-domain only, then fine-tune it on the mix of out-of-domain and (upsampled) in-domain data. [Dakwale and Monz, 2017], inspired by knowledge distillation, modify the method such that they assume learning out-of-domain and in-domain are two different task, and either the training is set to optimize 2 objectives (multi-objective fine-tuning) or they add two output layers on the fine-tuned model where each learns general-domain or in-domain (multi-output fine-tuning).

1.3.2 Back-translation

Back-translation is one of the methods to leverage the target monolingual data to improve the fluency of MT output. One can use a reverse MT system (target-to-source) to translate the target monolingual data. The translation and the target monolingual data, usually referred as *synthetic* data, are then added the to the training set to train the original MT system (source-to-target). Figure 1.2 illustrates the process. Despite being referred as synthetic data, note that only the source side of the data is actually synthetic.

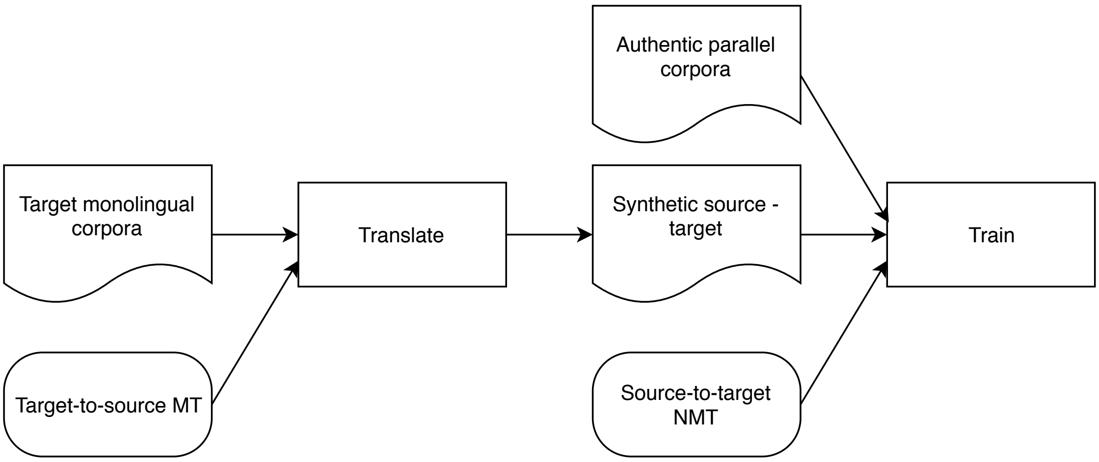

Authentic parallel corpora
Target monolingual corpora
Translate
Synthetic source - target
Train
Target-to-source MT
Source-to-target NMT

Figure 1.2: The flow of back-translation.

While back-translation method has been used since the era of phrase-based SMT [Bertoldi and Federico, 2009, Bojar and Tamchyna, 2011], Sennrich et al. [2016] have shown that it also improves the NMT translation quality. They demonstrate that mixing the authentic and synthetic parallel data improve the NMT performance in both low- and high-resource settings. In domain-adaptation context they show that the method performs well on the model fine-tuned on a synthetic data, whose target is in-domain but back-translated by a reverse system without in-domain knowledge. However, as pointed by Poncelas et al. [2018] and Fadaee and Monz [2018], it is unclear which factors actually contribute to the NMT translation performance. Moreover, the effect of the factors can be different for low- and high-resource settings [Edunov et al., 2018].

The quantity of synthetic data

Regarding the authentic-to-synthetic ratio in the training set, Sennrich et al. [2016] experiment with 1:1 ratio on both high- and low-resource settings and show the improvement caused by the back-translation. García-Martínez et al. [2017] find that adding more synthetic data is always beneficial in low-resource setting, indicating the importance of the amount of the training data. Meanwhile in high-resource setting, when the amount of the synthetic data is much larger than the authentic data, the performance of the NMT model is worse than the model trained on the authentic data only since the model gets biased towards the source sentences in the synthetic data [Poncelas et al., 2018, Fadaee and Monz, 2018].

The quality of synthetic data

Several works have tried to investigate the effect of the quality of the synthetic source sentences. Sennrich et al. [2016] randomly sample the sentences from the large monolingual corpora. Meanwhile Fadaee and Monz [2018] extend the random sampling method to focus on increasing the occurrences of words that are difficult to predict in the target language. Thus, the synthetic data containing such words is expected to optimally benefit the translation quality. Imamura et al. [2018] generate more than one synthetic source sentence from the target sentence by using sampling instead of beam search. Both approaches are experimented in high-resource settings. Furthermore, Edunov et al. [2018] show that using sampling or noised beam outputs for generating the source sentence outperforms beam search in high-resource setting, but not in low-resource setting. However, Caswell et al. [2019] argue that the noise in the synthetic source generated by the noised-beam does not diversify the source sentences but only to label the sentence as synthetic. Thus, they propose the *tagged back-translation* in which they add a synthetic tag as an additional token at the beginning of the synthetic source sentences. This idea is similar to the use of tag in multi-domain NMT [Kobus et al., 2016] in which one can consider the sentences with and without the tag belong to 2 different domains.

Incorporating synthetic data

The simple way how to incorporate the synthetic data to the training set is by combining it with the authentic data and shuffle the set. While not explicitly described, we assume that all the previously mentioned works use this so-called *mixed* (or *shuffled*) training regime Sennrich et al. [2017]. Another way the authors have described is the *fine-tuned* training regime, in which the model is firstly trained on the authentic data only, then is fine-tuned on the mixture of authentic and synthetic.

Popel [2018] introduces *concat* training regime, in which the authentic and synthetic data are combined with concatenation, instead of mixing them. He shows that during the training, the evaluation on the development set shows that the BLEU score is decreased when moving from authentic to synthetic data block, but improved when moving from synthetic to authentic data block, even slightly better than the *mixed* regime. The advantage of using the concat regime becomes more apparent when the author leverages *checkpoint averaging*, a technique of model ensemble by averaging the weights in the last N checkpoints element-wise and yielding a single averaged model. When the optimal ratio of checkpoints between authentic and synthetic data is found, the performance improvement in the *concat* regime is higher, as illustrated in Figure 1.3. In his experiment he uses 8 checkpoints and the optimal authentic:synthetic ratio is 6:2.

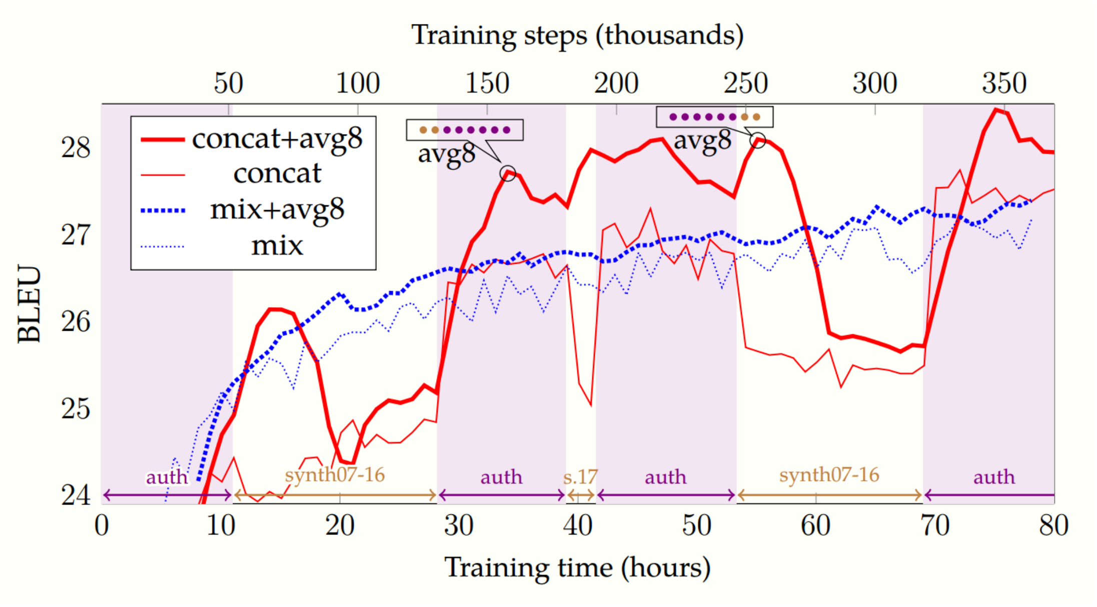


Figure 1.3: The curve of BLEU score on the dev set when training with back-translation using *concat* regime, adopted from [Popel, 2018]. The checkpoint average benefits the model as it improves the peaks caused by the transitions from authentic to synthetic data block.

2. Experiment Setup

To reach our objectives as described in the Introduction, we run two sets of experiments, namely training EN-ID and ID-EN NMT systems using model fine-tuning and training ID-EN NMT systems using back-translation. In this chapter, we elaborate on the corpora we use to train and evaluate our NMT systems, the NMT architecture we use, and the general training setup used in all experiments. We explain more thoroughly the setup for more specific experiments in the corresponding chapter, namely in Chapter 3 (model fine-tuning) and 4 (back-translation).

2.1 English-Indonesian Datasets

In this section, we summarize the dataset we use for our experiments. We consider the TED talks corpus as our in-domain data and OpenSubtitles2018 as our out-of-domain data. Although both are spoken language corpora, in Section 2.1.1 we describe the properties of the corpora and what makes them different, as well as pre-processing we conduct on the corpora.

Generally, the datasets from different resources often have different format, hence we need to process them in order to obtain the same format and ready-to-use version for our experiments. The data used in our experiments are composed of one source file and one target file. The file has one-sentence-per-line format, in which each corresponding line of the two files is the translation of each other.

2.1.1 Parallel corpora

In this section, we describe the properties and the partition of the parallel corpora we use for our experiments.

WIT^3 TED talk

This corpus contains transcriptions of TED talks and their translations to different languages prepared by WIT^3 for IWSLT2017 (International Workshop on Spoken Language Translation) [Cettolo et al., 2012]. We use the EN-ID version of the corpus as our in-domain dataset. Although we presume most of the talks in the corpus are originally in English, the human translators consider more about the meaning and the objective of the speakers instead of the literal translation. Thus, we expect similar translation quality for both EN-ID and ID-EN in terms of adequacy and fluency.

Table 2.1 shows the partitions of the corpus that we use for our experiments. We notice that tst2017plus provided at the website[1] contains a small part of the train data. Thus, we remove the overlapping part from the original train data and result in partition `train-mod` as shown in the table. We use the merge of tst2013 and tst2014 as our in-domain dev set. The average number of words in a sentence in the train set is around 14.6 for Indonesian and 16.8 for English.

[1] https://wit3.fbk.eu/mt.php?release=2017-01-more, accessed on 25th February 2019

Partition	Purpose	#sentences	#words	
			ID	EN
train-mod	in-domain train	106,916	1,558,693	1,793,422
tst2013	in-domain dev	1,034	16,279	18,623
tst2014	in-domain dev	878	13,331	14,515
tst2015-16	in-domain test	980	14,091	16,954
tst2017-plus	in-domain test	1,448	18,986	22,912
Total		111,256	1,621,380	1,866,426

Table 2.1: The partitions of WIT^3 TED talks parallel corpus

OpenSubtitles2018

OpenSubtitles parallel corpora contain subtitles of movies or TV episodes obtained from OpenSubtitles website [Lison et al., 2018]. The 2018 version of the corpora contains more than 9 millions EN-ID parallel sentences from 9827 movies, which benefits us in tackling the lack of resource to train NMT. Unlike TED talks, movie subtitles comprise conversations and thus can be used to analyse the dialogue phenomena and the property of colloquial language [Lison et al., 2018]. Moreover, the average number of words in a sentence is shorter than sentences in TED talks, namely around 5.1 for Indonesian and 6 for English, as computed from our train set.

Partition	Purpose	#sentences	#words	
			ID	EN
train	out-of-domain train	9,268,870	47,044,960	54,976,466
dev_A	not used	1,221	6,954	8,371
test_A	not used	1,603	7,234	8,295
dev_B	out-of-domain dev	1,049	5,193	6,529
test_B	out-of-domain test	1,066	5,613	6,383
Total		9,273,809	47,069,954	55,006,044

Table 2.2: The partitions of OpenSubtitles2018 parallel corpus

Since OpenSubtitles2018 corpus does not have partition for train/dev/test sets, we split the corpus into those three sets as follows:

1. We extract dev and test sets with two different schemes: document level and sentence level.

   - For document level (`dev_A` and `test_A`), we randomly select 2 transcription documents for each set and append all sentence pairs in the documents to the set.
   - For sentence level (`dev_B` and `test_B`), we randomly select 5 documents for each set. We randomly select around 200 sentence pairs from each document and append those to the development set (and test set respectively).

2. We append the remaining sentence pairs to the train set.

Table 2.2 shows the statistics of the partitions for OpenSubtitles2018 corpus. While we do not use partitions `dev_A` and `test_A` in this thesis, we expect they can be used in future works to evaluate MT output quality in document level.

2.1.2 Monolingual data

In our back-translation experiments, we use English monolingual data to produce synthetic Indonesian data by translating the English data with our trained EN-ID NMT. The list of monolingual data used in our experiments is shown in Table 2.3.

Dataset	#sentences	#words
WIT^3 TED talks	136,951	2,440,065
Crawled TED talks	8,202	136,679
news-discussion2013 (ori)	9,555,910	150,221,485
news-discussion2013 (sample)	3,750,000	57,458,781

Table 2.3: English monolingual corpora used in our back-translation experiments. Only the light rows are back-translated. The dark row is reported only to inform the readers about the original size of the corpus.

Besides the parallel corpora, WIT^3 also has the monolingual collections of the TED talks. Since there are more English transcriptions than the Indonesian translations, we extract the English monolingual talks that are not contained in the EN-ID parallel corpus using the processing scripts from WIT^3. We obtain $136,951$ English sentences as shown in Table 2.3. In order to obtain more in-domain data, we crawl all TED talks from 2018 until 13th May 2019 and extract their English transcriptions. Since not all transcriptions are available, we can only obtain $8,202$ English sentences.

In addition to TED talks monolingual data, we also use a subset of news-discussion2013 from WMT website.[2] Although the dataset consists of written discussions in an online forum, the language used is more casual than common written language corpora, e.g. News Crawl. Moreover, the average word in a sentence of this dataset is around 15.3, which is close to the average word of English sentences in our in-domain parallel corpus.

Since the corpus is noisy, we filter the sentences containing more than 400 characters and URLs from the original corpus using a simple regular expression, because we think they are not useful for our training data. The expression is as follows:

```
grep -Ev "https?:" | grep -v "www\." | grep -Ev "\.(com)|(org)|
(gov)|(co\..*)/+" | grep -Ev "~@.* " | awk ’length<401’
```

[2] http://data.statmt.org/news-discussions/en/

Note that the expression is too simple to cover all possible URLs, yet we minimize the occurrence. Then we sample only 3,750K sentences for our back-translation experiments, as we will further describe in Section 4.1.

2.2 General Setup

We run all experiments in this thesis using Tensor2tensor (T2T) version 1.11.0 [Vaswani et al., 2018] with TensorFlow 1.12.0 as the backend. We train our models on one or four GeForce GTX Ti 1080 GPUs.

2.2.1 Terminology

For clarity, we define some practical terms used in this thesis as follows:

- **Batch size** or mini-batch size refers to the number of training examples in one training step when using a gradient-based optimization method. In T2T, the parameter `batch_size` indicates the approximate number of tokens (subwords) in one batch. The number of tokens in an example is specified as the maximum of source and target subwords [Popel and Bojar, 2018]. The number is only an approximation since some padding is still needed. The parameter is set for use in one GPU.

- **Maximum sequence length** specifies the maximum number of subwords in a sentence, either in source or target language. Sentences that are longer than the specified size are exempted from the training. Setting it to a large value allows the model to learn understanding or generating longer sentences. But it also takes more space in the batch. Such sentences might not be too frequent. If they are in a batch in which other sentences are much shorter, the batch is not used efficiently as the other sentences need padding. However, setting it too low will discard many sentences from the training data.

- **Training epoch** refers to one complete pass of the training data. As pointed by Popel and Bojar [2018], it is not easy to measure the number of training epochs in T2T since it only reports the training steps. It requires a user-specified script to compute the training epoch.

- **Training steps** or iterations specify the number of weight updates the optimizer does. This number is also equal to the number of mini-batches processed during the training.

2.2.2 Model and training hyperparameters

We use the Transformer model with hyperparameter set `transformer_base` [Vaswani et al., 2017]. To be more specific, some important hyperparameters in this setting are:

Number of encoder stacks (N_{enc})	$= 6$
Number of decoder stacks (N_{dec})	$= 6$
Embedding size (d_{model})	$= 512$
Hidden units in fully-connected layers (d_{ff})	$= 2048$
Number of heads in multi-head attention layers (h)	$= 8$
Size of key/query vectors (d_k)	$= 64$
Size of value vectors (d_v)	$= 64$

For the vocabulary, we use the default subword units implemented in T2T, *SubwordTextEncoder*, which is shared between source and target languages with approximate size of 32,678 units. Our data is not tokenized.

We set the maximum sequence length to 150, following the suggestion of Popel and Bojar [2018]. We set the learning rate to 0.2 (default setting in T2T) and learning rate warm-up steps to 8000. This warm-up setup allows the learning rate to grow linearly in the first 8000 steps and then decrease to the inverse square root of the step number [Vaswani et al., 2017]. For the batch size, we use the size of 2048 for the model fine-tuning experiments and larger sizes (6000 and 8000) in the back-translation experiments. We optimize our model using the Adafactor optimizer [Shazeer and Stern, 2018].

We evaluate our model on the development set during the training and the test set after the model selection using case-insensitive BLEU score computed by the built-in command `t2t-bleu`.

2.2.3 Decoding

During decoding, we use beam search algorithm implemented in T2T by default. The implementation follows Equation 1.6 without the coverage penalty.[3] We initially set the hyperparameters to the default value from T2T, namely beam size=4 and length penalty parameter (α)=0.6. These hyperparameters are then optimized in our beam search experiment (see Section 3.2) and we use the optimized values for all experiments.

We run a post-processing script using simple regular expressions, as shown in Figure 2.1, to cut unnecessary repetitions in the end of the translations. These repetitions often occur when we set the length penalty parameter too large. The script deletes:

- the repetition of phrases that consecutively occur more than twice and keeps the first instance (we assume the phrases consist of 1-4 words),
- over-reduplication on Indonesian sentences, marked by hyphens.

Example 2 below shows an example of the use of the post-processing script in our EN-ID baseline. In the example, the over-duplicated phrases ”*hal-hal-...*” are normalized to the correct form ”*hal-hal*”. The phrase ”*menjadi sesuatu yang berarti,*” which occurs four times is reduced to once.

[3] `https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/utils/beam_search.py#L610`

```perl
#!/usr/bin/env perl
use strict;
use warnings;
use utf8;
use open qw(:std :utf8);

while(<>){
    s/( \S{2,})\1{2,}/$1/g;
    s/(-\S{2,})\1{2,}/$1/g;
    s/(-\S{2,})\1{1,}/$1/g;
    s/( \S+ \S+)\1{2,}/$1/g;
    s/( \S+ \S+ \S+)\1{2,}/$1/g;
    s/( \S+ \S+ \S+ \S+)\1{2,}/$1/g;
    print;
}
```

Figure 2.1: Post-processing script for decoding.

Example 2
(Indonesian)
Before post-processing:
... untuk mengambil informasi bahwa hal-hal-hal yang tidak berarti, sehingga hal-hal yang tidak berarti dalam hal-hal-hal-hal-hal-hal yang tidak berarti, menjadi sesuatu yang berarti, menjadi sesuatu yang berarti, menjadi sesuatu yang berarti, menjadi sesuatu yang berarti, menjadi sesuatu yang berarti dalam arti
After post-processing:
... untuk mengambil informasi bahwa hal-hal yang tidak berarti, sehingga hal-hal yang tidak berarti dalam hal-hal yang tidak berarti, menjadi sesuatu yang berarti, menjadi sesuatu yang berarti dalam arti

3. Domain Adaptation with Model Fine-tuning

In this chapter we explore the effectiveness of leveraging out-of-domain parallel data for the low-resource EN-ID and ID-EN NMT systems through model fine-tuning, as described in Section 1.3.1. Firstly, we introduce three scenarios to train non-adapted NMT models: one with low-resource setting and two with high-resource setting. Then we tune the beam search hyperparameters to obtain a better-quality translation in terms of BLEU score. After that, we fine-tune the high-resource models by continuing the training on the in-domain data. We report the BLEU score on the development set and analyze the result of all scenarios.

3.1 Training EN-ID and ID-EN NMT

We use TED talks parallel corpus as our in-domain data and OpenSubtitles2018 as our out-of-domain data (see Section 2.1.1). For both EN-ID and ID-EN we train 3 models with the following scenario:

1. IN (low-resource): trained on only the in-domain data.
2. OUT (high-resource): trained on only the out-of-domain data.
3. MIX (high-resource): trained on the mixture of in- and out-of-domain data.

We train the three models on a single GPU using the batch size of 2048 for 500,000 steps and save the checkpoint hourly. In the IN scenario, we stop the training earlier since the model seems to overfit. We select the checkpoint with the highest BLEU score, namely at step 137,603 for EN-ID and 191,022 for ID-EN.

Table 3.1 shows the performance of the models decoded using the default beam search hyperparameters on the in-domain development set. For both EN-ID and ID-EN systems, MIX models obtain the best BLEU score. Thus, we will use the MIX models to tune the hyperparameters for beam search for both translation directions. We leave further analysis on the final result in Section 3.3.

	EN-ID	**ID-EN**
IN	26.52	26.73
OUT	24.07	25.87
MIX	27.66	28.48

Table 3.1: Initial BLEU score for the non-adapted models

3.2 Tuning beam search hyperparameters

In this experiment, our goal is to select the beam search hyperparameters, i.e. beam size and length penalty parameter that result in the highest BLEU score in

α	EN-ID					
	bs=1	bs=4	bs=5	bs=6	bs=10	bs=20
0.2	25.74	27.47	27.46	27.45	27.18	26.88
0.4	25.74	27.56	27.60	27.58	27.36	27.06
0.6	25.74	27.66	27.71	27.71	27.59	27.26
0.8	25.74	27.76	27.86	27.87	27.95	27.72
1	25.74	27.92	28.05	28.09	28.14	27.93
1.2	25.74	28.00	**28.12**	**28.22**	**28.29**	**28.11**
1.5	25.74	**28.08**	28.05	28.11	28.24	27.29
2	25.74	17.93	15.83	15.19	13.27	11.90

α	ID-EN					
	bs=1	bs=4	bs=5	bs=6	bs=10	bs=20
0.2	28.24	28.19	27.91	27.96	27.67	27.39
0.4	28.24	28.36	28.09	28.10	27.86	27.57
0.6	28.24	28.48	28.25	28.28	28.14	27.93
0.8	28.24	28.61	28.53	28.57	28.44	28.34
1	28.24	28.70	28.68	28.71	28.62	28.62
1.2	28.24	28.87	**28.84**	28.94	**28.86**	**28.80**
1.5	28.24	**28.92**	**28.84**	**29.01**	28.70	27.87
2	28.24	16.54	14.74	14.32	13.20	12.96

Table 3.2: The BLEU scores of all beam size (*bs*) and length penalty (α) combinations for beam search experiment for EN-ID and ID-EN. The black border marks the combinations fall under the time limit. The blue border marks the best combination under the time limit. The blue cells are the actual best combinations. The scores in bold are the best BLEU scores for each *bs*. The light gray cells are the combinations improved by our post-processing script.

a reasonable time limit. We will then use the values for the decoding stage in the rest of our experiments, including for translating the monolingual corpus for the back-translation experiments. We set the time limit to 150 seconds for decoding the whole in-domain development set (1912 sentences). For this experiment, we use the last MIX model checkpoint (after 500k training steps). We try all combinations of the following values:

- beam size (*bs*): 1 (greedy search), 4, 5, 6, 10, 20.
- length penalty parameter (α): 0.2, 0.4, 0.6, 0.8, 1, 1.2, 1.5, 2.0.

Table 3.2 shows the result of this experiment. While for EN-ID the best combinations are *bs*=6 and α=1.2, for ID-EN we obtain *bs*=4 and α=1.2.

Although our chosen combinations have lower BLEU scores than the actual best combinations, the differences are less than 0.2. The result shows that the model gains the best BLEU score when $\alpha > 1$ complemented by the post-processing script. However, too large α (=2) reduces the performance drastically. The light gray cells when α is large in the tables indicate that our post-processing script is useful to normalize too long sentences caused by repetitions.

3.3 Fine-tuning NMT

We continue the training of our high-resource models, OUT and MIX, on the in-domain data for 50,000 steps. We save the checkpoint every 20 minutes. This continuation also means we use the same vocabulary and the learning rate is set to the last value in the first training phase and continues to decay according to the original schedule. Our models generate the translations using the tuned beam size and length penalty values.

	EN-ID		**ID-EN**	
	in-domain	out-of-domain	in-domain	out-of-domain
IN	26.77	14.59	27.03	19.87
OUT	24.03	27.10	26.22	34.12
MIX	28.22	**27.83**	28.87	**34.70**
OUT+FINE	32.31 (+8.29)	18.34	31.34 (+5.12)	30.27
MIX+FINE	**33.93** (+5.71)	19.19	**32.79** (+3.92)	31.08

Table 3.3: BLEU scores of our models on the in- and out-of-domain development sets for the model fine-tuning experiment.

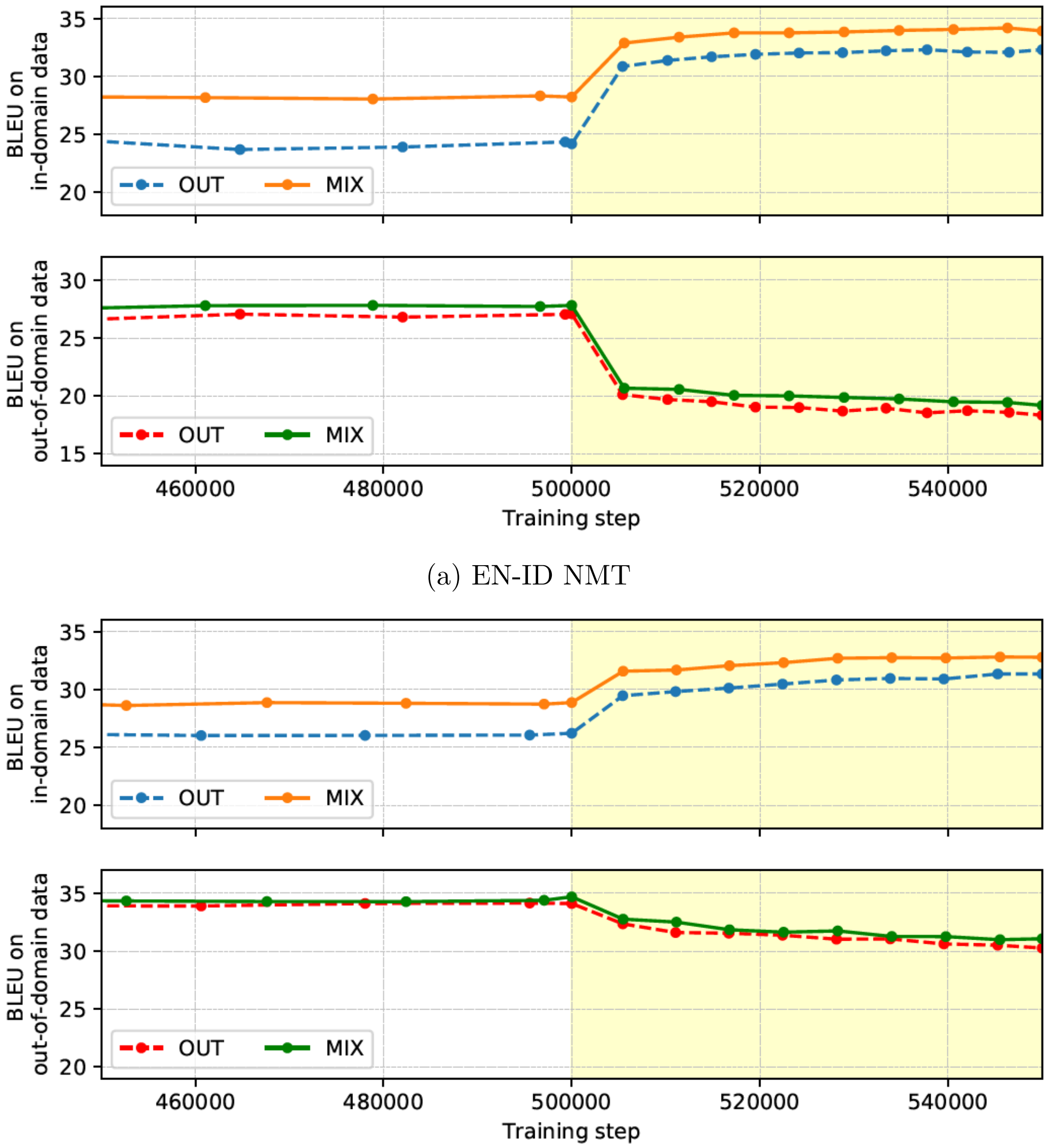
Table 3.3 shows our models' performance evaluated on the development sets and Figure 3.1 illustrates the effect of fine-tuning for in- and out-of-domain development sets.

We describe the evaluation on the in-domain data as follows:

- For both EN-ID and ID-EN, the model trained on in-domain data only (IN) works better than out-of-domain data only (OUT) although the training-data sizes are significantly different (3:100 ratio for in- : out-of-domain on word-level).
- Domain adaptation (fine-tuning) helps to improve our high-resource models, OUT and MIX. We observe the fine-tuning method has a higher impact on the out-of-domain model than on the mixture model. Nevertheless, MIX+FINE shows the best performance for both EN-ID and ID-EN.
- The impact is also higher for EN-ID than ID-EN. We hypothesize the style difference between the two spoken language corpora (TED talks and OpenSubtitles) is more apparent in Indonesian than in English.

The evaluation on the out-of-domain data shows an unsurprising result, namely the performance drops after the fine-tuning due to model overfit to the in-domain data. However, the effect is different for EN-ID and ID-EN as shown in Figure 3.1. The performance drop in EN-ID NMT is more significant than in ID-EN NMT.

The effect of fine-tuning for EN-ID NMT shown in Figure 3.1a indicates, although we can reduce the overfitting effect by setting a limit for the fine-tuning duration, it does not really help since the performance has been decreased significantly in the beginning. We have enumerated several existing solutions to handle this overfitting issue in Section 1.3.1. Since in this thesis we focus on the improvement on the specific-domain (TED talks) we set from the beginning, we do not perform any of the solutions. Thus, we leave the issue for a potential future work.



(a) EN-ID NMT

(b) ID-EN NMT

Figure 3.1: BLEU score on the dev sets during the training of OUT and MIX models. The yellow area shows the fine-tuning phase. On each subfigure, the upper plot shows the BLEU curves on TED talks dev set (in-domain), while the lower on OpenSubtitles2018 `dev_B` set (out-of-domain).

4. Back-translation

In this chapter, we explore how to leverage English monolingual corpora to train ID-EN NMT systems through back-translation. We focus on the method to incorporate the synthetic data to the training set. First, we train an EN-ID NMT system that is adapted to our target domain. Second, we use the trained EN-ID system to translate the large English monolingual corpora to Indonesian. Then, we use the translation as our synthetic data to train our ID-EN NMT.

We try two training regimes known from related work (see Section 1.3.2), namely the *shuffled* and *concat* regimes. Moreover, we introduce a new training regime, *4-way-concat*, and investigate its performance. We compare the performance of our back-translation models with a baseline that is only trained on authentic data. We also try different scenarios for fine-tuning our back-translation models. We report the performance of our ID-EN NMT systems on the development sets.

4.1 Translating Monolingual Corpora

4.1.1 Sampling

As shown in Table 2.3, we use English monolingual data from three sources. For our TED talks monolingual corpora, we translate all the sentences. This gives us the ratio of authentic-to-synthetic in-domain data around 7:10 with respect to the number of English words. Additionally, we sample randomly sentences from the original news-discussion2013 corpus. We expect to have the authentic-to-synthetic ratio nearly 1:1 with respect to the number of English words. Thus we sample 3,750K sentences, which give us around 57M English words. It is comparable to our out-of-domain parallel corpus containing around 55M English words (see Table 2.2).

4.1.2 Training EN-ID NMT

As described in Section 2.1.2, our large monolingual corpus does not belong to speech-styled spoken language. In order to generate synthetic data that is closer to the style of our target domain, we train the EN-ID system using MIX+FINE scenario from the previous chapter. The training uses 1 GPU with a batch size of 8000. We train the model for 7 days (around 1,210K steps). We stop because we think the performance has reached sufficient BLEU score on the in-domain development set and the improvement is no longer substantial – the last 4 days of training (around 27K steps) only gives us 1 BLEU point improvement, considering we will fine-tune the model. Then we fine-tune the model on in-domain data until convergence after around 4 hours of training. For decoding, we use the tuned beam size of 6 and length penalty of 1.2, and apply the post-processing script. The performance of the model evaluated on the development set using BLEU score is shown in Table 4.1. Note that the result is better than the model we have presented in the previous chapter (as shown in Table 3.3) because we train it much longer using a four times larger batch size.

	in-domain	out-of-domain
MIX	31.27	28.89
MIX+FINE	36.33	21.47

Table 4.1: BLEU score of EN-ID NMT system on development sets

4.1.3 Additional filtering

Since the translation of the news-discussion2013 corpus contains noisy sentences with repetitive HTML escapes (ca. 0.2%), we remove such sentences from the corpus resulting in 3,742,009 sentences. We do not find such noise in TED talks synthetic data so we do not filter those translations.

4.2 Training Data for ID-EN NMT

Our final training data for our ID-EN NMT systems are shown in Table 4.2. We categorize our data into 4 *data blocks* based on their type (authentic, -AUTH, or synthetic, -SYNTH) and domain (in-domain, IN-, or out-of-domain, OUT-). Note that our in-domain data are TED talks transcription and our out-of-domain data are from OpenSubtitles2018 and news-discussion2013.

Data Block	Data Source	#sent's (K)	#words (K)	
			ID	**EN**
IN-AUTH	WIT^3 `train_mod`	107	1,559	1,793
OUT-AUTH	OpenSubtitles2018 `train`	9,269	47,045	54,976
IN-SYNTH	WIT^3 monolingual, crawled TED talks	145	2,263	2,577
OUT-SYNTH	news-discussion2013	3,742	50,808	57,311
	Total	13,263	101,6745	116,657

Table 4.2: Our ID-EN NMT training data sizes (in thousands). The data sources are described in Section 2.1.

4.3 Extended Concat Training Regime

In Section 1.3.2, we have mentioned several ways to incorporate synthetic data to the training set. In this thesis we introduce a novel approach which we call *4-way-concat* regime. Inspired by Popel [2018], we concatenate our data blocks in the following order: OUT-AUTH, OUT-SYNTH, IN-AUTH, and IN-SYNTH. We also try using the checkpoints average since it is found to be beneficial for the *concat* regime [Popel, 2018], especially when the ratio of checkpoints from the consecutive blocks is optimal.

While the order of -AUTH and -SYNTH blocks criss-crossing in the *4-way-concat* regime is similar to the setup in *concat* regime, we expect the duo IN-AUTH and IN-SYNTH blocks in the end perform as an internal fine-tuning. We hypothesize the setup will take advantage of the checkpoint average to gain better performance than the non-*concat* regimes. Moreover, although we call the back-translated news-discussion2013 sentences as out-of-domain data, they have been actually translated by a domain-adapted system, thus should have some kind of in-domain knowledge. Since the in-domain data (IN-AUTH+IN-SYNTH) is still much smaller than the out-of-domain one, we expect the internal fine-tuning takes advantage of the domain-adapted OUT-SYNTH data block.

Additionally, we also try modifying the data blocks, in which we replace the OUT-SYNTH data block with the shuffled mixture of OUT-SYNTH and OUT-AUTH. We compare this *4-way-concat* method with the standard one that we introduce.

4.4 Experiment Setup

We conduct experiments by building ID-EN NMT systems with different training regimes, as follows:

1. AUTHENTIC (baseline): trained on authentic parallel data only.

2. SHUFFLED: trained on the shuffled mixture of authentic and synthetic parallel data, equal to the *mixed* regime by Sennrich et al. [2017]. We use the name ”SHUFFLED” as not to confuse the reader with our existing model MIX.

3. CONCAT: trained on the concatenation of authentic and synthetic data. The sentences are shuffled internally inside each data block, but not across the data block. The approach is equal to the *concat* regime by Popel [2018].

4. 4CONCAT: trained using our novel *4-way-concat* method, as described in the previous section.

5. 4CONCATMOD (additional): trained using the modified *4-way-concat* method, as described in the previous section.

Figure 4.1 illustrates the incorporation of the authentic and synthetic data in our experiments. Note that the baseline system (AUTHENTIC) is trained on less data, 4CONCATMOD is trained on more data in an epoch (containing upsampled OUT-AUTH), while the other systems are trained on the same amount of data.

For each of the systems, we fine-tune the model on in-domain data, similarly to the fine-tuning experiment we have performed in Chapter 3. We set 3 different fine-tuning scenarios:

1. FINE-A: the model is fine-tuned on IN-AUTH data.

2. FINE-S: the model is fine-tuned on IN-SYNTH data.

3. FINE-AS: the model is fine-tuned on the mixture of IN-AUTH and IN-SYNTH data (data are shuffled).

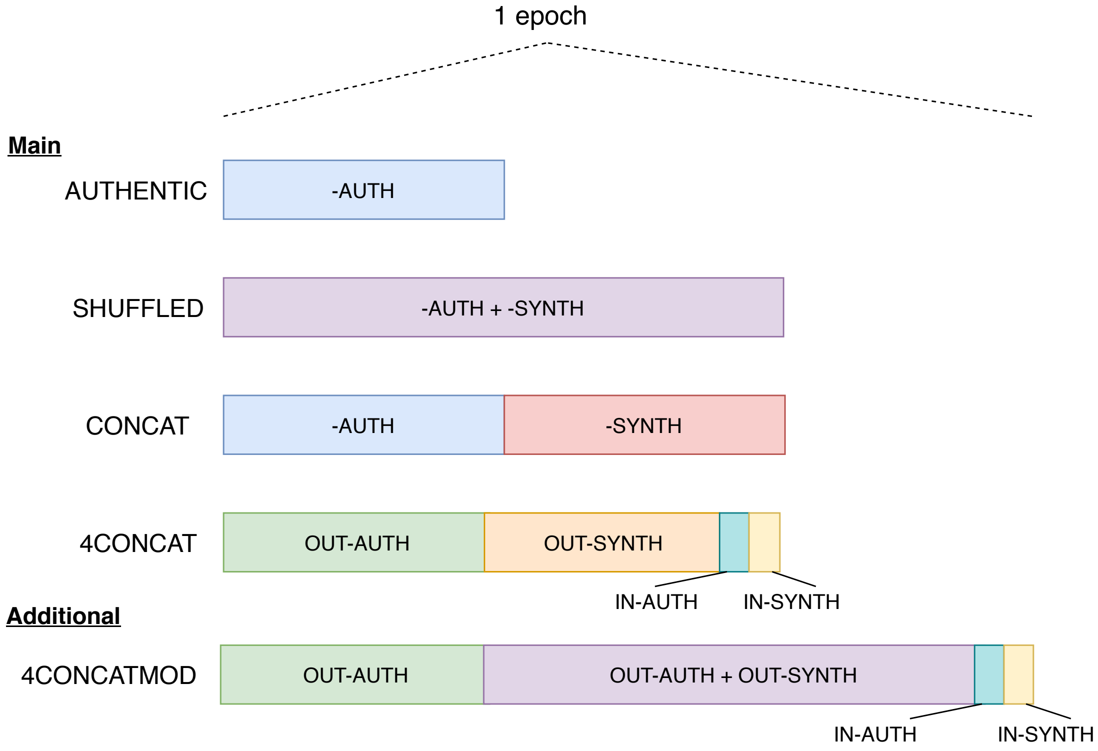


Figure 4.1: Illustration of the training data blocks in our experiments for 1 epoch.

We train the models using 4 GPUs with batch size of 6000. We train the four *main* systems (i.e. without fine-tuning) for around 60 hours and save the checkpoints every 30 minutes. We fine-tune the models until convergence (up to around 12 hours) and save the checkpoints every 10 minutes. For all experiments, we use the vocabulary obtained from the joint subword units of the authentic parallel data, similar to the setup used by Sennrich et al. [2016]. The reason is that the subword units learned from that data are enough to handle unknown words, as we have described in Section 1.2.2. We also use checkpoint average with 8 last checkpoints. We report the BLEU score on both in- and out-of-domain development sets.

4.5 Experiment Result

In this section, we present the result of our back-translation experiments.

4.5.1 Summary

We summarize the performance of all ID-EN systems in our back-translation experiments in this subsection, while we present the in-depth analysis in the following subsections. From each experiment, we select the checkpoint with the highest BLEU score with respect to the in-domain development set and report the score in Table 4.3.[1]

[1] Some results shown are different from the one in the previous version of this thesis since we found some technical errors. So we re-ran some experiments.

System	Checkpoint step	Hour	BLEU	
			In-domain	Out-of-domain
AUTHENTIC	444261	60.5	32.09	***36.26***
+FINE-A	455409	1.5	35.11	34.30
+FINE-S	465734	3.2	33.44	32.93
+FINE-AS	495234	7.8	*35.19*	33.21
SHUFFLED	412531	61.0	32.43	*35.42*
+FINE-A	464634	6.5	*35.63*	31.50
+FINE-S	433128	2.5	33.51	32.17
+FINE-AS	504328	11.0	35.45	32.09
CONCAT	446491	62.2	32.91	*35.06*
+FINE-A	492698	5.8	*35.68*	30.82
+FINE-S	449137	0.3	33.53	32.62
+FINE-AS	509333	7.7	35.45	31.98
4CONCAT	396057	55.0	32.46	*34.87*
+FINE-A	429730	4.0	*35.44*	32.23
+FINE-S	404350	1.0	33.25	32.26
+FINE-AS	495899	12.2	35.30	31.67
4CONCATMOD	437043	60.0	32.88	*35.56*
+FINE-A	479265	5.2	***35.84***	32.33
+FINE-S	449407	1.5	33.64	33.21
+FINE-AS	491782	6.7	35.51	32.49

Table 4.3: BLEU score of our ID-EN NMT systems on the development sets. Bold values represent the highest scores for some test set. Italic values represent the highest scores for each system.

For the main systems without fine-tuning, CONCAT performs the best on the in-domain data, obtaining around 0.5 and 0.03 BLEU point better than our proposed 4CONCAT and 4CONCATMOD, respectively. However, among all, our 4CONCATMOD+FINE-A system obtains the best score. As expected, the system trained without additional synthetic data performs the worst in our experiment, yet its score is only around 0.3 BLEU point lower than of SHUFFLED. Fine-tuning with the authentic data only generally performs better than the other two counterparts. For the evaluation on the out-of-domain data, AUTHENTIC obtains the best BLEU score among all systems. Note that the models used in the evaluation are the ones considered the best on the in-domain data, not necessarily on the out-of-domain one.

Figure 4.2 illustrates the overall performance of the four main systems (excluding 4CONCATMOD) with respect to the BLEU score on the in-domain development set. While AUTHENTIC and SHUFFLED form regular increase, CONCAT and 4CONCAT show more dynamic performance during 60 hours of training.

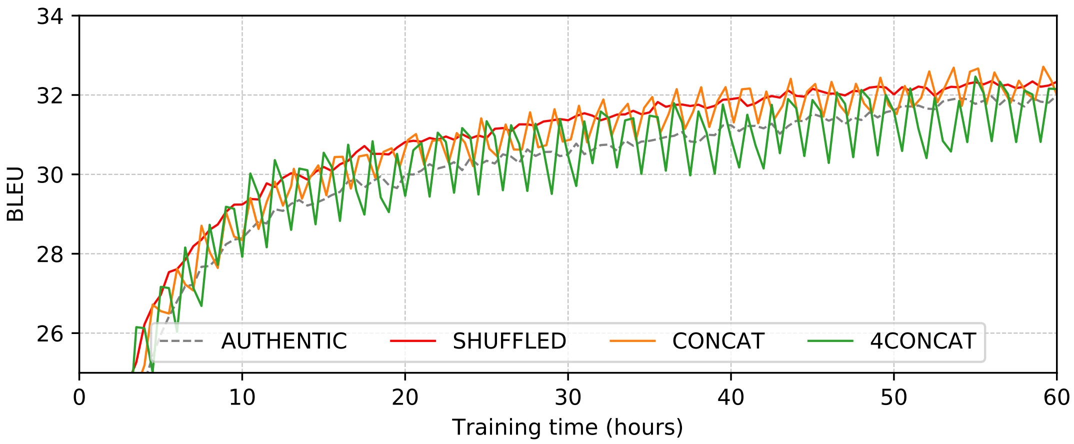


Figure 4.2: BLEU scores of the four main systems (without fine-tuning, 4CONCATMOD is considered as additional) on the in-domain development set.

4.5.2 The quality of the synthetic data

We measure the quality of the synthetic data relative to the authentic parallel data by training a system on the synthetic data only, which we call SYNTHETIC. The training setup is identical to the one used for training the other main systems.

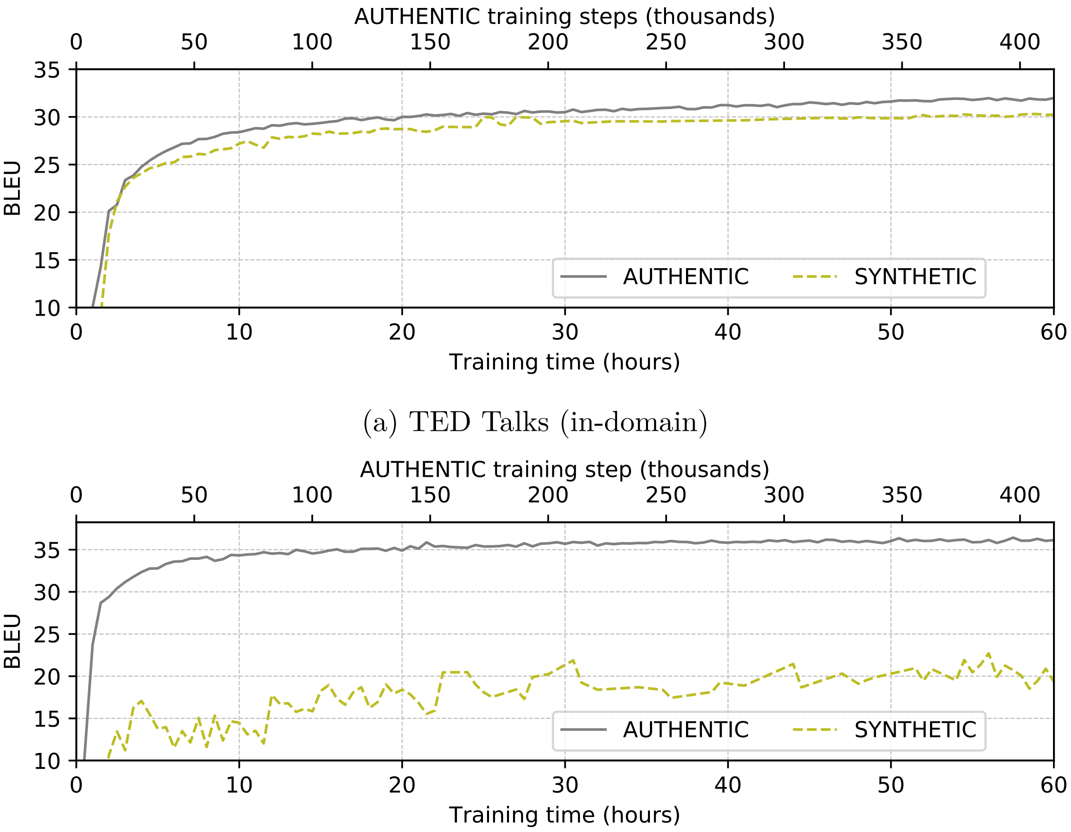
We compare the performance to AUTHENTIC. Figure 4.3 shows their BLEU score evaluation on the in- and out-of-domain development sets. On both sets, AUTHENTIC obtains higher BLEU scores than SYNTHETIC during the training time. The poorer performance of SYNTHETIC on the in-domain set (around 1.6 BLEU points less at the end of the training) shows that the quality of the synthetic data is still lower than the authentic data. We hypothesize that the much lower BLEU score of SYNTHETIC on the out-of-domain development set is because the EN-ID NMT system, which is used to translate the synthetic Indonesian sentences in the data, is fine-tuned to the TED Talks domain.

4.5.3 Training with shuffled regime

Figure 4.4 shows the BLEU scores of SHUFFLED and AUTHENTIC on both in- and out-of-domain development sets during the training. Although both systems are trained for the same training hours (comparable training steps), note that the size of the training data of AUTHENTIC is half the size of the training data of SHUFFLED. Thus, the numbers of their training epochs are not equal.

The figure shows that the performance of SHUFFLED is better than the baseline on the in-domain development set, which indicates adding synthetic data to the training set is beneficial for ID-EN NMT. While the improvement in early training (the first 10 hours) is around 1.25 BLEU point, it is reduced to as small as about 0.3 point after 60 hours of training. The BLEU scores on both systems still seem to grow, thus we hypothesize if we train the systems longer, there is a chance that their performances will get even closer.[2] However, due to a limited time for working on this thesis, we focus on the comparison of our systems in the

[2] 0.2 point after 84 hours. We continued the training after we had finished all experiments in this chapter. Thus, we still report the result after 60-62 hours (our specified time constraint).



(a) TED Talks (in-domain)

(b) OpenSubtitles (out-of-domain)

Figure 4.3: Performance evaluation on the development sets of the ID-EN NMT system trained on the authentic data only (AUTHENTIC) and the one trained on the synthetic data only (SYNTHETIC).

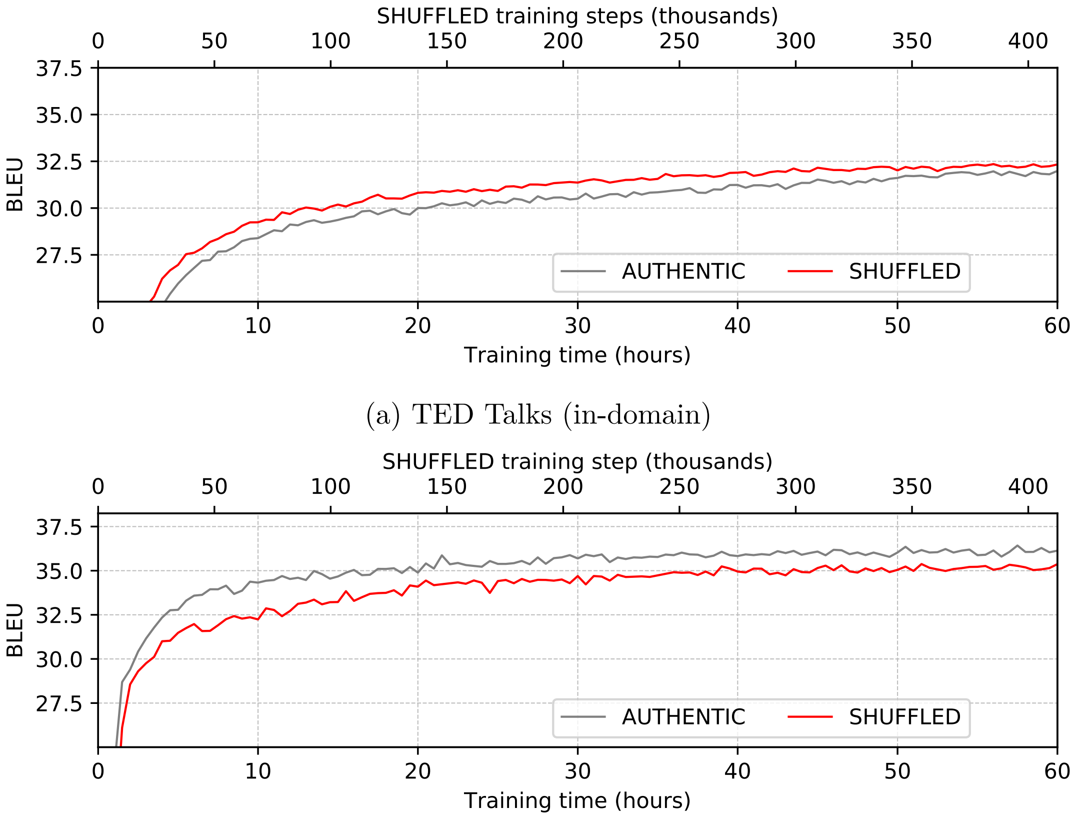
same time constraint that we set.

On the other hand, the performance of SHUFFLED on the out-of-domain set is poorer than of AUTHENTIC. This is expected because the training data of AUTHENTIC is dominated by sentences from OpenSubtitles corpus. Adding more sentences from TED Talks domain worsens the performance evaluation on the hold-out set of the subtitles corpus.

4.5.4 Training with concat regime

Figure 4.5 shows a closer look of the performance of our CONCAT system during training. The training loss forms a noticeable pattern, in which the loss is 1-2 units lower when training on the synthetic data than on the authentic data. This corresponds to the performance evaluation on the development set, in which the BLEU scores form *peaks* and *valleys*. The peaks can be up to 0.5 BLEU score higher than the performance of SHUFFLED after an equal training hours. The training stops at step 446K.

The pattern is in line with the finding of Popel [2018]. The performance changes across the data blocks, showing the system performs better when trained on the authentic block than the synthetic block. While this finding is expected as it is in line with our analysis in Subsection 4.5.2, it is not easy to estimate



(a) TED Talks (in-domain)

(b) OpenSubtitles (out-of-domain)

Figure 4.4: Performance evaluation of SHUFFLED vs AUTHENTIC on the development sets.

the data block spans in Figure 4.5 because, technically, T2T reports the training steps instead of the epochs. To get more evidence that the higher BLEU score checkpoints indeed belong to the authentic data block training, we run an experiment in which we train a model on the concatenation of two sets of the authentic training data and one set of the synthetic data (data not shown). Thus, the authentic block is around two times larger than the synthetic data. The evaluation curve on the development set shows similar performance changes to Figure 4.5, but the spans with higher BLEU scores are longer than the lower ones. The result indicates the spans with higher BLEU scores are when the model is trained on the authentic data.

Regarding the use of checkpoint average, we find that using the average of the last 8 checkpoints does not help the performance. The reason could be that the data blocks are too small that the method cannot capture the optimal ratio of the number of checkpoints from the authentic and synthetic blocks. Note that our data is about ten times smaller compared to English-Czech data used by Popel [2018], which contains about 122M sentences in the mixture of authentic and synthetic data.

Thus, we run an experiment in which we triple the size of the data blocks and save the checkpoint every 15 minutes (two times more often) so we can have more checkpoints in a data block. We call the model from this experiment CONCAT3x. We try different number of checkpoints: 4, 6, 8, 10. However, the result in Figure

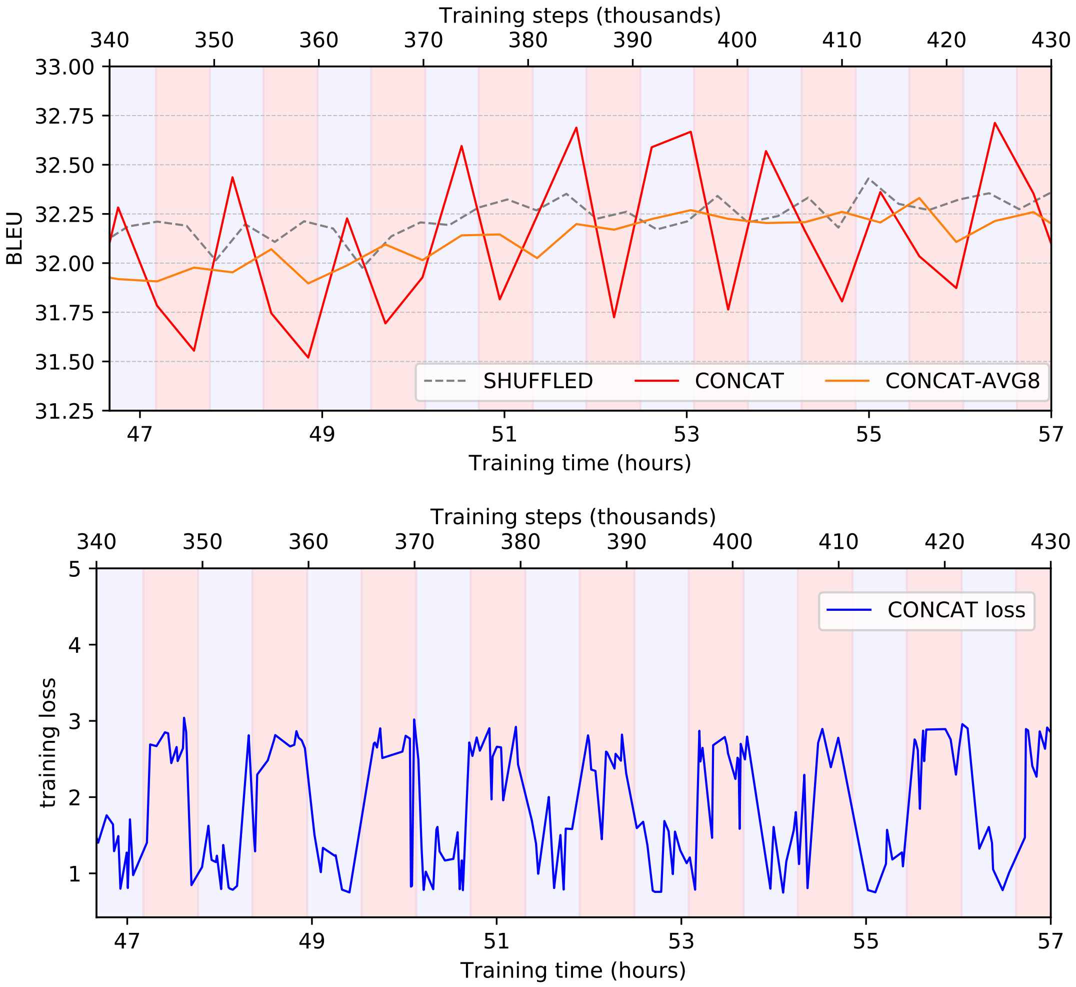


Figure 4.5: The performance of our CONCAT system during training shown by the BLEU scores on the in-domain development set (top) and the training loss (bottom). The training steps are relative to the CONCAT training. The light blue spans represent the authentic data blocks. The red spans represent the synthetic data blocks.

4.6 shows the averaged models do not perform better than the original (non-averaged) one. Although the method may take more checkpoints from a data block, they are actually trained on the same set of data, thus not guaranteed to learn more new information. We conclude that the checkpoint average from the last N checkpoints is not helpful for our CONCAT system.

4.5.5 Training with 4-way-concat regime

Similar to CONCAT, the evaluation of 4CONCAT on the in-domain development set results in peaks and valleys on the BLEU score curve, as shown in Figure 4.7. While in the early training steps (10-18 hours) the peaks of 4CONCAT are slightly higher than of CONCAT (0.2-0.6 BLEU score, see Figure 4.2), the peaks of 4CONCAT are lower than of CONCAT after getting closer to the end of the training time. Furthermore, the figure also shows that using checkpoint average with $N = 8$ does not help the performance.

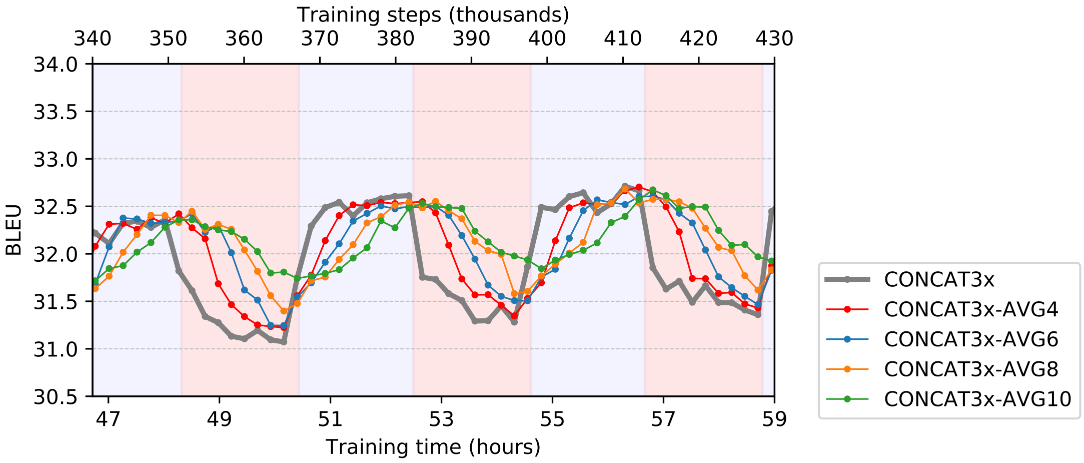


Figure 4.6: The performance of the model trained with *concat* regime with tripled-size data blocks and shorter checkpoint saving (CONCAT3x) and its checkpoint averaged models with various N (-AVGN). The light blue spans represent the authentic data blocks. The red spans represent the synthetic data blocks.

The sudden increases of the performance occur in the transition from OUT-SYNTH data block. Our internal fine-tuning on IN- data blocks does not seem to help due to the very small size of the blocks in comparison to OUT- data blocks, as shown in Figure 4.7. Since the IN- data blocks are very small, there is only a small chance that our checkpoints saved in every 30 minutes are obtained during the training on these blocks. It is still not observable when we try to analyze it from the training loss curve (data not shown) since the loss looks very similar to of CONCAT and due to the small size of IN- data blocks. Thus, we cannot conclude that the distinct increases in BLEU score are due to the information the model learns from the IN- data blocks

We also try to upsample each data block three times larger and run similar experiment to CONCAT3x to see whether enlarging the data blocks will help us investigate the effect of the transition, especially into or out of the IN- data blocks, or be beneficial for the checkpoint average. We call this experiment as 4CONCAT3x. Figure 4.9 shows there are sudden increases occuring at the end of epochs. We argue the IN- data blocks play a role on these increases. However, the model cannot preserve such performance as the BLEU scores drop regularly in the OUT-AUTH blocks and drastically in the transitions to OUT-SYNTH. This behavior shows that taking out the in-domain data leaves the model performs worse in a longer period. Nevertheless, the peaks of 4CONCAT3x are higher than of 4CONCAT (+0.5 BLEU point on the both highest peaks).

We also try using checkpoint average with $N = \{4, 6, 8, 10\}$ on 4CONCAT3x. In Figure 4.9, we can see that using $N = 4$ generally results in higher BLEU scores than the non-averaged model, yet the peaks are not always significantly higher than. Meanwhile, the other values of N do not seem to give better scores. We hypothesize the way we design the data blocks does not allow the average method to find a good ratio of ”higher score checkpoints” and ”lower score checkpoints”. In Figure 1.3, Popel [2018] utilizes a small block of synthetic data between two authentic data blocks. The size is enough to cause a sudden jump yet not too large that a few checkpoints saved in that block (lower score checkpoints) can

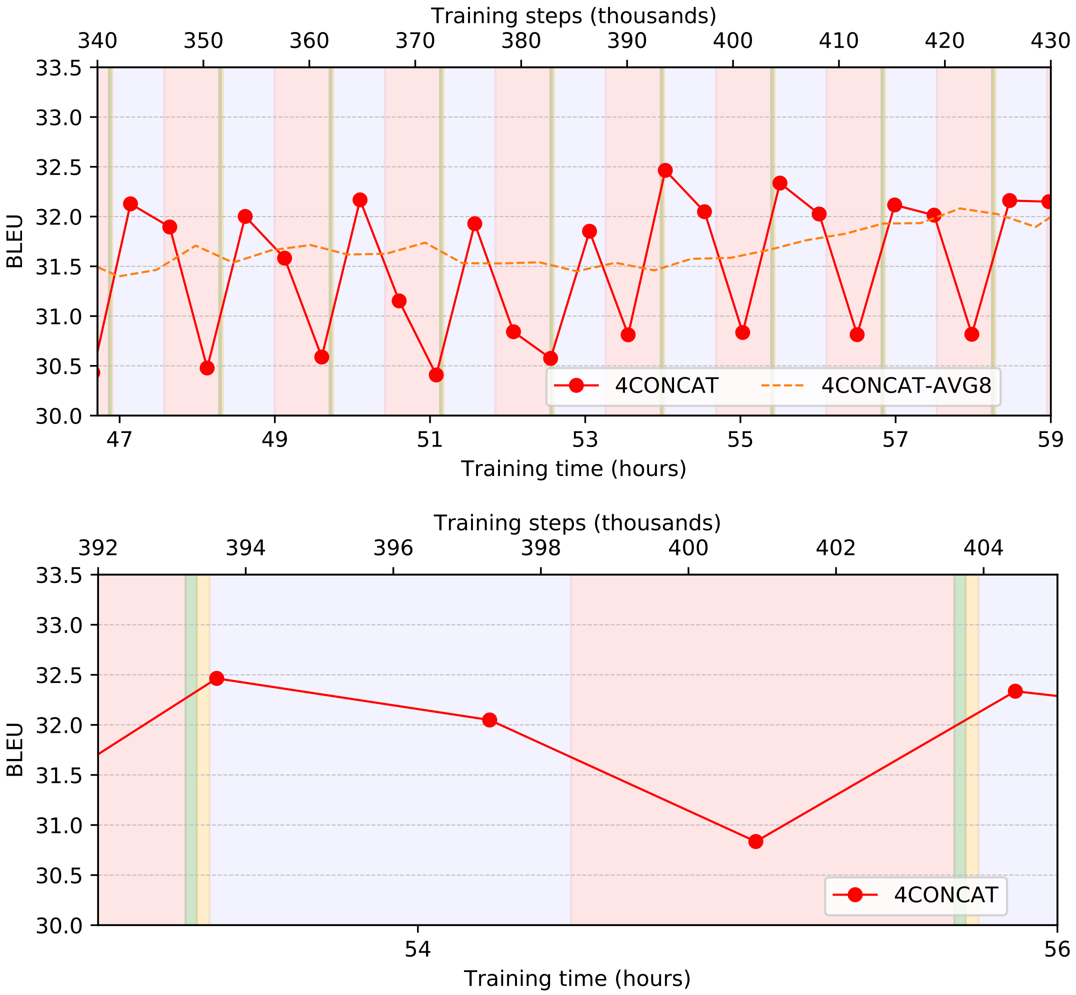


Figure 4.7: The performance of our 4CONCAT system during training shown by the BLEU scores on the in-domain development set. The bottom figure shows the closer look to an epoch in the 4CONCAT training. The light blue spans represent OUT-AUTH data blocks. The red spans represent OUT-SYNTH data blocks. The thick lines between the red and blue spans are IN-AUTH and IN-SYNTH data blocks.

be averaged with the checkpoints from the authentic data blocks (larger score checkpoints). However, we do not conduct such an experiment in this thesis and we consider it as interesting future work.

The performance of our 4CONCATMOD system on the in-domain development set, as shown in Figure 4.8, is generally better than 4CONCAT: the BLEU scores that form peaks and valleys in the BLEU curve of 4CONCATMOD are higher than of 4CONCAT. The figure shows that the peaks in the curve of 4CONCATMOD are less frequent and the valleys are longer. The latter is expected since we include the authentic data into the second block of our training data blocks (see Figure 4.1) which makes the block almost two times larger than of CONCAT. Although there is no obvious increase in the BLEU score during the training in those second blocks, we hypothesize the improvement (relative to 4CONCAT) is obtained in the overall training. Overall, the BLEU score of the peaks of 4CONCATMOD are slightly higher than CONCAT. But the difference is very small

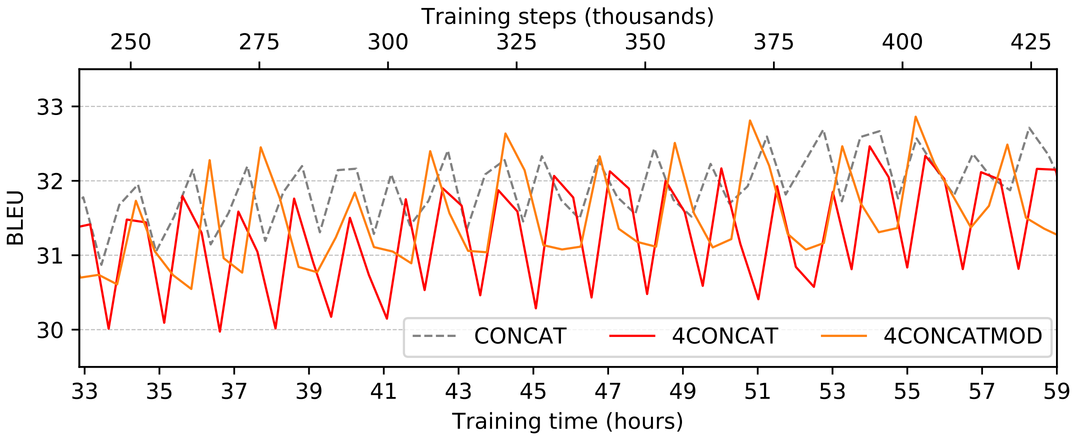


Figure 4.8: The comparison of the BLEU scores of our *concat*-based systems on the in-domain development set.

that we conclude that their performances are comparable.

4.5.6 Effects of fine-tuning

Figure 4.10 shows the effect of fine-tuning with different in-domain data, after being trained using different training regimes. The best BLEU scores after the fine-tuning approaches are shown in Table 4.3. We summarize the findings as follows:

- According to Table 4.3, the best BLEU score on the in-domain development set is obtained when fine-tuning 4CONCATMOD on the authentic data (+FINE-A). It gives around 2.8 BLEU point improvement over 4CONCATMOD and is 0.4-2.8 BLEU points higher than the other +FINE-A systems. The highest BLEU scores on the out-of-domain development sets are obtained by the systems without fine-tuning. This is expected and in line with our observation in Subsection 3.3

- Fine-tuning on different data results in different training time required until convergence:

  - fine-tuning on the authentic data only takes 1-7 hours
  - fine-tuning on the synthetic data only takes 1-3.5 hours
  - fine-tuning on the mixture of authentic and synthetic data takes 7.5 - 12 hours

- From the last four subfigures in Figure 4.10, we learn that there is no obviously distinct behavior among the performance of the back-translation systems (SHUFFLED, CONCAT, 4CONCAT, 4CONCATMOD) fine-tuned with the same scenario after around 12 hours of fine-tuning. This suggests that no matter which training regime is used for the training, the fine-tuning always gives improvement. What differs is the effect of the data used for the fine-tuning.

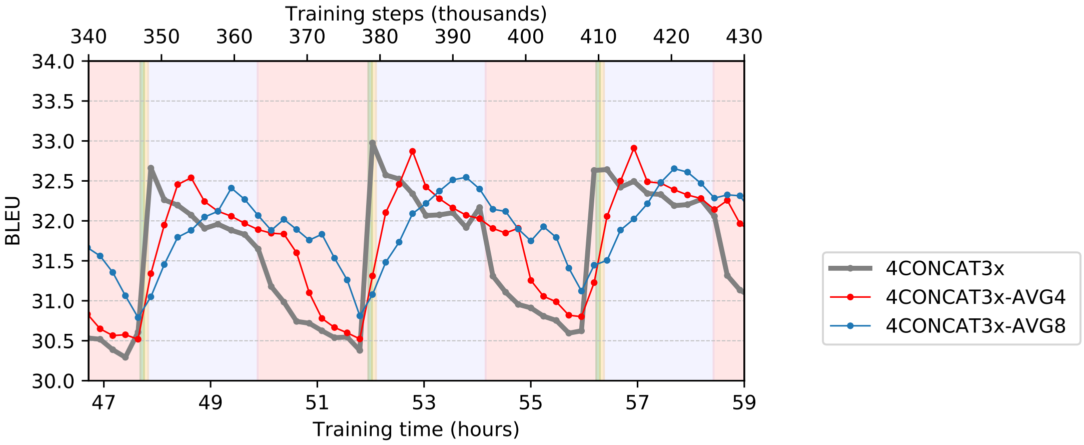
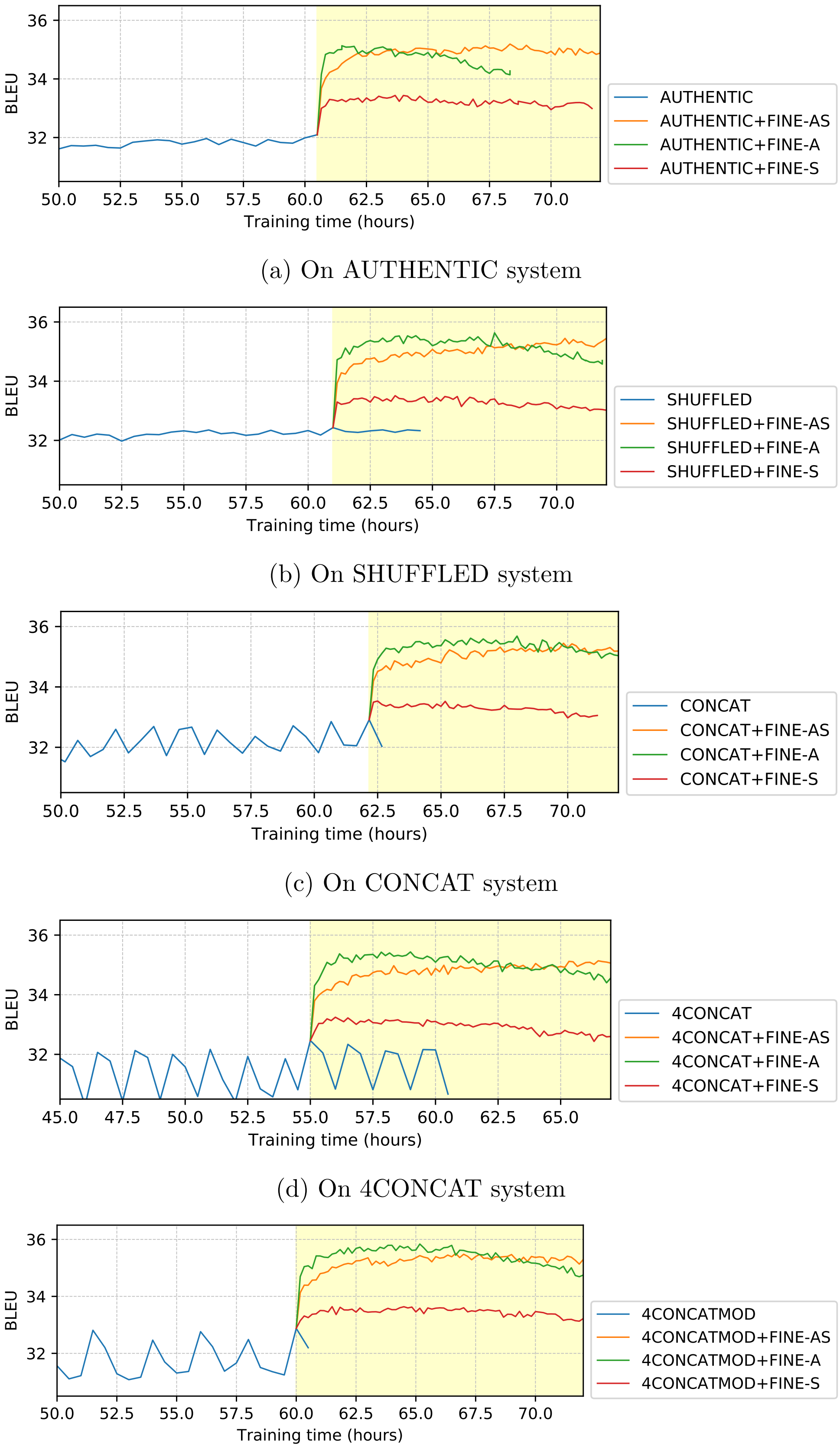


Figure 4.9: The performance of the model trained with *4-way-concat* regime with tripled-size data blocks and shorter checkpoint saving (4CONCAT3x) and its checkpoint averaged models with various N (-AVGN) on the in-domain dev set. The colored spans are relative to 4CONCAT3x and not to 4CONCAT, so are the training steps. The blue spans represent OUT-AUTH data blocks. The red spans represent OUT-SYNTH data blocks. The thick lines between the red and blue spans are IN-AUTH and IN-SYNTH data blocks.

- For all systems, fine-tuning on the synthetic data only (+FINE-S) gives the lowest BLEU scores among the three approaches. Despite having comparable size to the in-domain authentic data and being translated by a fine-tuned NMT system, our in-domain synthetic data does not give much improvement for the fine-tuning. Our +FINE-S systems only obtain 0.6-1.3 BLEU points improvement over the non-fine-tuned ones.

- Table 4.3 shows that all of our back-translation systems obtain the highest BLEU scores after being fine-tuned on the authentic data only (+FINE-A). Those systems gain 1.6-2.3 BLEU points higher than their +FINE-AS counterparts. Meanwhile, fine-tuning with mixed data (+FINE-AS) on AUTHENTIC gives only <1 BLEU point over AUTHENTIC+FINE-A. While fine-tuning on authentic data results in early convergence and overfit, adding the synthetic data does not seem to give improvement. We argue that it is due to the quality of our synthetic data. However, the slower convergence of the +FINE-AS systems to the +FINE-A ones also indicates that they are less prone to overfit to the in-domain set. The BLEU scores on the out-of-domain development set (data not presented) support our argument. It shows that the scores of the +FINE-A systems on the out-of-domain data also decrease faster than of the +FINE-AS systems. We hypothesize that if we have synthetic data with better quality, we can gain higher BLEU scores for the +FINE-AS systems while keeping the performance more general than the +FINE-A systems.



(a) On AUTHENTIC system

(b) On SHUFFLED system

(c) On CONCAT system

(d) On 4CONCAT system

(e) On 4CONCATMOD system

Figure 4.10: Performance of fine-tuning different ID-EN NMT systems trained using different training regime (in yellow span), evaluated on the in-domain development set.

5. Evaluation

We evaluate our EN-ID and ID-EN NMT systems on the unseen test sets, namely our in-domain test sets (WIT^3 TED talks tst2015-16 and tst2017plus) and our hold-out OpenSubtitles2018 test set (`test_B`). Our EN-ID NMT systems are the ones that we use to translate the monolingual corpora in Section 4.1, which also represent the result of the model fine-tuning experiments in Chapter 3. Meanwhile, our ID-EN NMT systems are from the back-translation experiments in Chapter 4.

We also compare the evaluation result of our NMT systems with the commercial system, Google Translate. We translate the test sentences using their Web API[1]. They claim the default model used in their system is an NMT model.

We compute the BLEU scores and the statistical significance with bootstrap resampling [Koehn, 2004] using MT-ComparEval [Klejch et al., 2015], an interface that is able to summarize the comparison of different MT systems based on some metrics. While the computation of BLEU in MT-ComparEval uses *international-tokenization* similar to `t2t-bleu`, we found out those two tools report slightly different scores for some systems. In this chapter, we report the case-insensitive scores computed by MT-ComparEval.

5.1 EN-ID NMT Evaluation on Test Sets

Table 5.1 shows the result for the evaluation of our EN-ID systems on the test sets. Google Translate obtains the best scores on both TED talks test sets by around 2 BLEU points better than our MIX+FINE. The improvement of our MIX-FINE system around +3 BLEU points over the non-adapted MIX system is statistically significant ($p < 0.05$). For the evaluation on `test_B`, our MIX system obtains the best BLEU score, similarly to the evaluation on the development set. Its better performance than Google Translate might be because `test_B` is a hold-out set of OpenSubtitles2018 and the training set of the MIX system is dominated by OpenSubtitles2018 without further fine-tune to a certain domain.

System	tst2015-6	tst2017-plus	test_B
Google Translate	**35.15**	**34.67**	30.55
MIX	↓ 30.74	↓ 30.33	↑ **39.92**
MIX+FINE	↓ 33.17 ↑	↓ 32.52 ↑	↑ 35.24 ↓

Table 5.1: Evaluation of our EN-ID NMT systems on the test sets with BLEU score. Bold texts represent the best scores on a given test set. The arrows mark the statistical significance ($p < 0.05$) relative to Google Translate (left) and the non-adapted system (right).

We conduct a manual analysis on the tst2017plus translations to see in which situations the systems succeed or fail. For the comparison of MIX and MIX+FINE translations, the analysis from MT-ComparEval on unigram shows

[1] https://cloud.google.com/translate/, as of 18th July 2019.

Example 1	
Source	Because I know , and I know all of you know , this isn ’ t Tuscany .
Reference	Karena saya tahu , saya tahu Anda semua tahu , ini bukan Tuscany .
MIX	Karena aku tahu , dan aku tahu kalian semua tahu , ini bukan Tuscany .
MIX+FINE	Karena saya tahu , dan saya tahu Anda semua tahu , ini bukan Tuscany .
Example 2	
Source	We ’ re not real people . We are there to inspire .
Reference	Kami bukan orang betulan . Kami ada untuk menginspirasi .
MIX+FINE	Kita bukan orang sungguhan . Kita ada di sana untuk menginspirasi .
Google	Kami bukan orang sungguhan . Kami ada untuk menginspirasi .
Example 3	
Source	What my father could not give to my sisters and to his daughters , I thought I must change it .
Reference	Apa yang ayah saya tidak dapat berikan kepada saudara - saudara perempuan saya dan kepada anak - anak perempuannya , saya pikir saya harus mengubahnya .
MIX+FINE	Apa yang tidak bisa diberikan ayah saya kepada saudara perempuan saya dan putrinya , saya pikir saya harus mengubahnya .
Google	Apa yang ayah saya tidak bisa berikan kepada saudara perempuan saya dan anak perempuannya , saya pikir saya harus mengubahnya .

Figure 5.1: EN-ID translation examples of tst2017-plus from different systems. The colors mark the related segments across the translations. The underline marks the colored segments mismatched to the reference.

that MIX+FINE wins on sentences containing pronouns ”*saya*” (*I*) and ”*Anda*” (*you*). We observe such cases are dominated by the transformation from a less formal pronouns generated by MIX (”*aku*”, ”*kamu/kau/kalian*”) to a more formal one by MIX+FINE (”*saya*”, ”*Anda*”). Example 1 in Figure 5.1 illustrates this case.

For the comparison of the translations from MIX+FINE and Google Translate, in the figure, Example 2 illustrates a case when context is needed for word selection. The pronouns ”*we*” in English can be translated as either an inclusive pronouns (”*kita*”) or an exclusive one (”*kami*”). While the translation by MIX+FINE is also acceptable, it does not match the single reference. This shows the ineffectiveness of using a single reference.

Example 3 shows the translation of ”*daughters*” as ”*anak perempuan*” or ”*putri*” (synonyms) cannot be captured by a single reference. This example also shows when MIX+FINE and Google Translate fail to translate plural words, which are usually marked by reduplication forms of Indonesian nouns.

In this section, we have shown how our EN-ID NMT systems perform on unseen data. The domain-adapted system still performs worse than the commercial system on TED talks data according to the automatic evaluation result. We expect our domain-adapted EN-ID NMT can still be improved. One potential approach worth trying is to also leverage Indonesian monolingual corpora through back-translation and run our an iterative back-translation [Hoang et al., 2018] using our back-translation ID-EN NMT system.

5.2 ID-EN NMT Evaluation on Test Sets

No	System	tst2015-16	tst2017-plus	test_B
	Google Translate	32.18	↑ 32.01	↓ 38.05
	AUTHENTIC (baseline)	30.75	30.51	**44.93**
1	SHUFFLED	30.42	30.63	44.26
2	CONCAT	31.02 ↑	30.78	44.03
2	4CONCAT	30.08	30.12	43.00
2*	4CONCATMOD	30.96 ↑	31.08 ↑	44.82
3*	2*+FINE-A (the best on dev set)	↑ 32.21 ↑	↑ 32.13	↓ 40.87 ↓
3*	2*+FINE-S	31.21	↑ 31.38	↓ 41.58 ↓
3*	2*+FINE-AS	↑ **32.24** ↑	↑ **32.81** ↑	↓ 41.13 ↓

Table 5.2: Evaluation of our ID-EN NMT systems on the test sets with BLEU score. Bold texts represent the best scores on a given test set. ↑↓ mark statistical significance ($p < 0.05$). The arrows on the left are relative to AUTHENTIC. The arrows on the right are relative to the system on the previous number: {2,2*} to 1, 3* to 2*.

Table 5.2 shows the evaluation result of our ID-EN systems on unseen test sets. Regarding the improvement after using back-translation based on the evaluation on the in-domain test sets, our back-translation systems obtained higher BLEU scores than AUTHENTIC, except for 4CONCAT. This reflects our finding of the benefit of utilizing synthetic data. While 4CONCAT obtains the lowest BLEU scores on both in-domain test sets, the other *concat*-based systems, CONCAT and 4CONCATMOD, perform better than SHUFFLED with 0.1-0.6 BLEU point improvement, where the scores of 4CONCATMOD are significantly better.

Regarding the effect of model fine-tuning, we compare only the fine-tuned 4CONCATMOD systems due to the overall better results on the development set than the other back-translation systems. The improvement after model fine-tuning relative to the baseline is around 0.5 to 2.3 BLEU points. From the table, we can see that 4CONCATMOD+FINE-AS obtains the best BLEU score on the-domain test sets, with statistically significant difference to the scores of AUTHENTIC and the non-fine-tuned system 4CONCATMOD. While 4CONCATMOD+FINE-A is not the best system as what we observe from the evaluation on the development set, we find that the difference between its BLEU scores and of 4CONCATMOD+FINE-AS are not statistically significant.

Relative to the commercial system, the BLEU scores of our 4CONCATMOD +FINE-A and +FINE-AS systems surpass the scores of Google Translate on all test sets. All of our trained systems show higher BLEU scores on the out-of-domain test set than Google Translate. This finding is the same as what we find from the evaluation of our EN-ID systems in the previous Subsection and we hypothesize the reason is also the same. However, none of our fine-tuned systems have significantly better nor worse result than Google Translate on both in-domain test sets. We conclude that our ID-EN NMT systems are comparable to Google Translate for TED talks domain.

The overall performance are 1-3 scores lower than the evaluation on the development set.[2] However, the evaluation results on the in-domain test sets still show that our method of implementing back-translation followed by model fine-tuning improves the performance of our ID-EN NMT systems by up to around 2.3 BLEU points with respect to the baseline system.

[2] We also compute the BLEU scores using `t2t-bleu` to confirm this.

Conclusion and Future Work

In this thesis, we build ID-EN and EN-ID NMT systems for the low-resource TED talks domain using the state-of-the-art model, the Transformer. We implement two domain-adaptation methods to improve their performance. Firstly, we have shown how large out-of-domain parallel corpora can be leveraged to improve low-resource EN-ID and ID-EN NMT systems through model fine-tuning. Our experiments on spoken language domains leverage movie subtitles corpus to adapt to the TED talks domain. Although the impact is slightly different for EN-ID and ID-EN NMT systems, we have shown that the method has succeeded in improving the performance of both systems.

Secondly, we have shown how to leverage large English monolingual corpora to improve ID-EN NMT systems through back-translation. Our experiments focus on different approaches for incorporating the back-translated monolingual corpora to the training set of the ID-EN systems. We implement the existing approaches, *shuffled* and *concat* training regimes, and introduce a new approach called *4-way-concat* regime. In this novel approach, we set the training data as the concatenation of four different data blocks sequentially: out-of-domain authentic, out-of-domain synthetic, in-domain authentic, and in-domain synthetic data blocks. We also try another variant of concatenating the four data blocks. Moreover, we try different fine-tuning scenarios for our back-translation systems. Our back-translation experiment findings are as follows:

- The overall performance of the systems that incorporate monolingual corpora is better than the system that does not.
- While we cannot confirm that the system trained using *4-way-concat* regime (4CONCATMOD) is better than the one using the original *concat* regime (CONCAT), we have shown that on their best performance, both systems are better than the system trained using *shuffled* regime.
- Averaging the model checkpoints in the *concat*-based training regimes does not help improve the performance if the data blocks are small.
- Model fine-tuning is always helpful to adapt the model to a specific-domain, regardless the training regime used for training the main model.
- Fine-tuning on in-domain authentic data results in overfitting. Leveraging synthetic data in the fine-tuning seems to alleviate the issue.

Moreover, we have evaluated our systems and a commercial system, Google Translate, on the unseen test sets. First, we have evaluated the EN-ID NMT system that we use to translate English monolingual corpora. Although the result confirms the improvement caused by model fine-tuning method, our best EN-ID NMT system still performs worse than Google Translate on the TED talks domain. Second, we have evaluated several ID-EN NMT systems that we have from the back-translation experiments. The result does not only confirm our findings, but it also shows that our best ID-EN NMT system performs comparably to the commercial system on the TED talks domain. Our method of implementing back-translation followed by model fine-tuning improves the performance of our

ID-EN NMT systems by around 0.5-2.3 BLEU points. Our best system, trained using *4-way-concat* and then fine-tuned on the synthetic in-domain data, obtains BLEU score of 32.24 and 32.81 on WIT^3 TED talk tst2015-6 and tst2017plus, respectively, which are insignificantly higher than the score of Google Translate.

There are many directions of future work to improve the current EN-ID and ID-EN NMT systems. In this thesis, our focus on using use model fine-tuning is to improve the system performance on the target-domain, regardless the degradation of the performance on other domains. To build a more general EN-ID or ID-EN NMT system, future works can experiment with the extended model fine-tuning methods that deal with overfitting issues.

We have conducted back-translation experiments only on ID-EN NMT. The potential future work is to improve our EN-ID NMT systems by leveraging Indonesian monolingual corpora. We can also make use of our current best ID-EN NMT system to translate the monolingual corpora, i.e. an experiment with iterative back-translation to improve the performance of NMT systems on both directions. Moreover, in this thesis our exploration in the back-translation method for ID-EN NMT focuses on the incorporation of the synthetic data to the training data. It is also interesting to explore the effect of back-translation for the system based on different factors, like the quantity or the quality of the synthetic data.

Bibliography

Cosmas Krisna Adiputra and Yuki Arase. Performance of Japanese-to-Indonesian Machine Translation on Different Models. *23rd Annual Meeting of the Speech Processing Society of Japan (NLP2017)*, pages 757–760, March 2017. URL http://www.anlp.jp/proceedings/annual_meeting/2017/pdf_dir/A5-5.pdf.

Teguh Bharata Adji. *Annotated Disjunct for Machine Translation.* PhD thesis, Universiti Teknologi Petronas, 2010. URL http://utpedia.utp.edu.my/2948/1/Annotated_Disjunct_for_Machine_Translation,_by_Teguh_Bharata_Adji,_Ph.D_in_IT.pdf.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR 2015*, pages 1–15, 2015.

Nicola Bertoldi and Marcello Federico. Domain Adaptation for Statistical Machine Translation with Monolingual Resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 182–189, Athens, Greece, 2009. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W09-0432.

Ondřej Bojar and Aleš Tamchyna. Improving Translation Model by Monolingual Data. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 330–336, Edinburgh, Scotland, 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W11-2138.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors. *Proceedings of the Third Conference on Machine Translation.* Association for Computational Linguistics, Belgium, Brussels, 2018. URL http://www.aclweb.org/anthology/W18-64.

M. Asunción Castaño, Francisco Casacuberta, and Enrique Vidal. Machine translation using neural networks and finite-state models. 1997. URL https://pdfs.semanticscholar.org/ad2c/fa96bf9149c8d2d5be606edf4203933e6194.pdf.

Isaac Caswell, Ciprian Chelba, and David Grangier. Tagged back-translation. *CoRR*, abs/1906.06442, 2019. URL http://arxiv.org/abs/1906.06442.

Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy, May 2012.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL `https://www.aclweb.org/anthology/D14-1179`.

Chenhui Chu and Rui Wang. A Survey of Domain Adaptation for Neural Machine Translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA, aug 2018. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/C18-1111`.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. An Empirical Comparison of Simple Domain Adaptation Methods for Neural Machine Translation. *CoRR*, abs/1701.03214, 2017. URL `http://arxiv.org/abs/1701.03214`.

Praveen Dakwale and Christof Monz. Fine-tuning for neural machine translation with limited degradation across in- and out-of-domain data. In *Proceedings of the 16th Machine Translation Summit (MT-Summit 2017)*, pages 156–169, 2017.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. *CoRR*, abs/1808.09381, 2018. URL `http://arxiv.org/abs/1808.09381`.

Marzieh Fadaee and Christof Monz. Back-translation sampling by targeting difficult words in neural machine translation. *CoRR*, abs/1808.09006, 2018. URL `http://arxiv.org/abs/1808.09006`.

Mikel L. Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, 2011. doi: 10.1007/s10590-011-9090-0. URL `https://doi.org/10.1007/s10590-011-9090-0`.

Markus Freitag and Yaser Al-Onaizan. Fast Domain Adaptation for Neural Machine Translation. *CoRR*, abs/1612.06897, 2016. URL `http://arxiv.org/abs/1612.06897`.

Mercedes García-Martínez, Ozan Caglayan, Walid Aransa, Adrien Bardet, Fethi Bougares, and Loïc Barrault. LIUM Machine Translation Systems for WMT17 News Translation Task. In *Proceedings of the Second Conference on Machine Translation*, pages 288–295, Copenhagen, Denmark, sep 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4726. URL `https://www.aclweb.org/anthology/W17-4726`.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional Sequence to Sequence Learning. *CoRR*, abs/1705.03122, 2017. URL `http://arxiv.org/abs/1705.03122`.

Andi Hermanto, Teguh Bharata Adji, and Noor Akhmad Setiawan. Recurrent neural network language model for english-indonesian machine translation: Experimental study. In *2015 International Conference on Science in Information Technology (ICSITech)*. IEEE, October 2015. doi: 10.1109/icsitech.2015.7407791. URL `https://doi.org/10.1109/icsitech.2015.7407791`.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. Iterative Back-Translation for Neural Machine Translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia, 2018. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W18-2703`.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, nov 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`.

Kenji Imamura, Atsushi Fujita, and Eiichiro Sumita. Enhancement of Encoder and Attention Using Target Monolingual Corpora in Neural Machine Translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 55–63, Melbourne, Australia, 2018. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W18-2707`.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On Using Very Large Target Vocabulary for Neural Machine Translation. In *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)*, pages 1–10, Beijing, China, 2015a. Association for Computational Linguistics. doi: 10.3115/v1/P15-1001. URL `https://www.aclweb.org/anthology/P15-1001`.

Sébastien Jean, Orhan Firat, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. Montreal neural machine translation systems for WMT'15. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 134–140, Lisbon, Portugal, September 2015b. Association for Computational Linguistics. doi: 10.18653/v1/W15-3014. URL `https://www.aclweb.org/anthology/W15-3014`.

Nal Kalchbrenner and Phil Blunsom. Recurrent Continuous Translation Models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA, 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/D13-1176`.

Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

Ondřej Klejch, Eleftherios Avramidis, Aljoscha Burchardt, and Martin Popel. MT-compareval: Graphical evaluation interface for machine translation development. *The Prague Bulletin of Mathematical Linguistics*, (104):63–74, 2015. ISSN 0032-6585.

Catherine Kobus, Josep Maria Crego, and Jean Senellart. Domain Control for Neural Machine Translation. *CoRR*, abs/1612.06140, 2016. URL http://arxiv.org/abs/1612.06140.

Philipp Koehn. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain, 2004. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W04-3250.

Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. *CoRR*, abs/1706.03872, 2017. URL http://arxiv.org/abs/1706.03872.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1557769.1557821.

Taku Kudo. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. *CoRR*, abs/1804.10959, 2018. URL http://arxiv.org/abs/1804.10959.

Surafel Melaku Lakew, Aliia Erofeeva, and Marcello Federico. Neural Machine Translation into Language Varieties. In *Proceedings of the Third Conference on Machine Translation*, pages 156–164, Belgium, Brussels, 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-64016.

Septina Dian Larasati. Towards an Indonesian-English SMT System : A Case Study of an Under-Studied and Under-Resourced Language , Indonesian. pages 123–129, 2012a.

Septina Dian Larasati. Improving word alignment by exploiting adapted word similarity. In *Proceedings of the Workshop on Monolingual Machine Translation (MONOMT) at AMTA 2012*, pages 41–45, San Diego, USA, 2012b. AMTA 2012 Organizing Committee.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. Fully Character-Level Neural Machine Translation without Explicit Segmentation. *CoRR*, abs/1610.03017, 2016. URL http://arxiv.org/abs/1610.03017.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. Character-based Neural Machine Translation. *CoRR*, abs/1511.04586, 2015. URL http://arxiv.org/abs/1511.04586.

Pierre Lison, Jörg Tiedemann, and Milen Kouylekov. OpenSubtitles2018: Statistical rescoring of sentence alignments in large, noisy parallel corpora. In *Proceedings of the 11th Language Resources and Evaluation Conference*, Miyazaki, Japan, May 2018. European Language Resource Association. URL `https://www.aclweb.org/anthology/L18-1275`.

Minh-Thang Luong and Christopher D. Manning. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*, Da Nang, Vietnam, 2015.

Minh-Thang Luong and Christopher D Manning. Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models. *CoRR*, abs/1604.00788, 2016. URL `http://arxiv.org/abs/1604.00788`.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. Addressing the Rare Word Problem in Neural Machine Translation. In *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)*, pages 11–19, Beijing, China, 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1002. URL `https://www.aclweb.org/anthology/P15-1002`.

Dominik Macháček, Jonás Vidra, and Ondřej Bojar. Morphological and Language-Agnostic Word Segmentation for NMT. *CoRR*, abs/1806.05482, 2018. URL `http://arxiv.org/abs/1806.05482`.

R. P. Neco and M. L. Forcada. Asynchronous translations with recurrent neural nets. In *Proceedings of International Conference on Neural Networks (ICNN'97)*, volume 4, pages 2535–2540 vol.4, June 1997. doi: 10.1109/ICNN.1997.614693.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL `https://doi.org/10.3115/1073083.1073135`.

Alberto Poncelas, Dimitar Shterionov, Andy Way, Gideon Maillette de Buy Wenniger, and Peyman Passban. Investigating backtranslation in neural machine translation. *CoRR*, abs/1804.06189, 2018. URL `http://arxiv.org/abs/1804.06189`.

Martin Popel. *Machine Translation Using Syntactic Analysis*. PhD thesis, Charles University, 2018.

Martin Popel and Ondřej Bojar. Training tips for the transformer model. *CoRR*, abs/1804.00247, 2018. URL `http://arxiv.org/abs/1804.00247`.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving Neural Machine Translation Models with Monolingual Data. pages 86–96, 2016.

Rico Sennrich, Alexandra Birch, Anna Currey, Ulrich Germann, Barry Haddow, Kenneth Heafield, Antonio Valerio Miceli Barone, and Philip Williams. The university of edinburgh's neural MT systems for WMT17. *CoRR*, abs/1708.00726, 2017. URL `http://arxiv.org/abs/1708.00726`.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. *CoRR*, abs/1804.04235, 2018. URL `http://arxiv.org/abs/1804.04235`.

Herry Sujaini, Kuspriyanto Kuspriyanto, Arry Akhmad Arman, and Ayu Purwarianti. A novel part-of-speech set developing method for statistical machine translation. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 12(3):581, September 2014. doi: 10.12928/telkomnika.v12i3.79. URL `https://doi.org/10.12928/telkomnika.v12i3.79`.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014. URL `http://arxiv.org/abs/1409.3215`.

Peter Toma. Systran as a multilingual machine translation system. In *Proceedings of the Third European Congress on Information Systems and Networks, Overcoming the language barrier*, page 569–581, 1977. URL `http://www.mt-archive.info/CEC-1977-Toma.pdf`.

Hai-Long Trieu, Duc-Vu Tran, and Le-Minh Nguyen. Investigating phrase-based and neural-based machine translation on low-resource settings. In *Proceedings of the 31st Pacific Asia Conference on Language, Information and Computation*, pages 384–391. The National University (Phillippines), November 2017. URL `https://www.aclweb.org/anthology/Y17-1051`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, François Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Lukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416, 2018. URL `http://arxiv.org/abs/1803.07416`.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL `http://arxiv.org/abs/1609.08144`.

List of Figures

List of Tables

List of Abbreviations

BLEU	bilingual evaluation understudy
EN	ISO 639-1 language code for English
GPU	graphics processing unit
GRU	gated recurrent units
HTML	hypertext markup language
ID	ISO 639-1 language code for Indonesian
IWSLT	International Workshop on Spoken Language Translation
LM	language model
LSTM	long short-term memory
MT	machine translation
NMT	neural machine translation
OOV	out-of-vocabulary
RBMT	rule-based machine translation
RNN	recurrent neural network
SMT	statistical machine translation
T2T	tensor2tensor
WIT^3	Web Inventory of Transcribed and Translated Talks `https://wit3.fbk.eu/`
WMT	Workshop on Machine Translation