Long Warm-up and Self-Training: Training Strategies of NICT-2 NMT System at WAT-2019

Kenji Imamura and **Eiichiro Sumita**
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan
{kenji.imamura,eiichiro.sumita}@nict.go.jp

Abstract

This paper describes the NICT-2 neural machine translation system at the 6th Workshop on Asian Translation. This system employs the standard Transformer model but features the following two characteristics. One is the long warm-up strategy, which performs a longer warm-up of the learning rate at the start of the training than conventional approaches. Another is that the system introduces self-training approaches based on multiple back-translations generated by sampling. We participated in three tasks—ASPEC.en-ja, ASPEC.ja-en, and TDDC.ja-en—using this system.

1 Introduction

This paper describes the NICT-2 neural machine translation (NMT) system at the 6th Workshop on Asian Translation (WAT-2019) (Nakazawa et al., 2019). This system employs Vaswani et al. (2017)'s Transformer base model but improves translation quality by applying the following training strategies and hyperparameters.

- We investigated the relationship between the learning rate, warm-up, and model perplexity, and found that a long warm-up allows high learning rates, and consequently the translation quality improves. According to this finding, we applied the long warm-up.
- We applied the self-training strategy, which uses multiple back-translations generated by sampling (Imamura et al., 2018) to increase the robustness of the encoder and improve the translation quality.

The remainder of this paper is organized as follows. Section 2 summarizes the system, including its settings. We describe the characteristics of our system—the long warm-up and the self-training based on multiple back-translations by sampling—in Sections 3 and 4, respectively. The results are presented in Section 5. Finally, Section 6 concludes the paper.

Corpus	Set		# Sents.	Note
ASPEC	Train		3,007,754	$\leq$ 128 tokens
	Dev.		1,790	
	Test		1,812	
TDDC	Train		1,398,184	$\leq$ 128 tokens
	Dev.	Items	2,845	
		Texts	1,153	
	DevTest	Items	2,900	
		Texts	1,114	
	Test	Items	2,129	
		Texts	1,148	

Table 1: Corpus sizes.

2 System Summary

We participated in three tasks, namely English-to-Japanese and Japanese-to-English of ASPEC (abbreviated to ASPEC.en-ja and ASPEC.ja-en, respectively), and Japanese-to-English of the TDDC (TDDC.ja-en).

The corpus used in the ASPEC tasks is Asian Scientific Paper Excerpt Corpus (Nakazawa et al., 2016), which is a collection of scientific papers. In the TDDC task, the Timely Disclosure Documents Corpus (TDDC) was used. The development and test sets of TDDC are divided into items and texts sets, which are collections of titles and body texts, respectively. The sizes of the corpora are shown in Table 1.

All corpora were divided into sub-words using the byte-pair encoding rules (Sennrich et al., 2016b) acquired from the training sets of each corpus. The rules were independently acquired from the source and target languages, to give a vocabulary size around 16K.

Proceedings of the 6th Workshop on Asian Translation, pages 141–146
Hong Kong, China, November 4, 2019.

	Attribute	Value
Model	# Layers	6
	d_{model}	512
	d_{ff}	2,048
	# Heads	8
Training	Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.99$)
	Loss function	Label-smoothed cross-entropy
	Label smoothing	0.1
	Batch size	Approx. 500 sentences
	Learning rate	0.0004
	Warm-up	Linear, approx. 5 epochs
	Cool-down	Inverse square root
	Dropout	Selected from {0.1, 0.15, 0.2}
	Clip norm	5
	Etc.	Early stopping Checkpoint averaging of 10 models
Test	Beam	10
	Length penalty	Tuned by Dev. set
	Etc.	Ensemble of 4 models

Table 2: Summary of system settings.

We used fairseq[1] as a basic translator. The model used here is the Transformer base model (six layers). Table 2 shows the hyperparameters of the model, training, and test.

Training was performed on Volta 100 GPUs (two GPUs for the ASPEC dataset and one GPU for the TDDC dataset) using 16-bit floating point computation. The training was stopped when the loss of the development set was minimized (i.e., early stopping). We also used checkpoint averaging: using the best checkpoint and the next nine checkpoints (10 checkpoints in total).

During testing, we used 32-bit floating point computation. For the final submission, four models, which were trained using different random seeds, were ensembled (Imamura and Sumita, 2017).

3 Long Warm-up

The warm-up is a technique that gradually increases the learning rate at the beginning of training. The most general strategy is to increase the learning rate linearly. Using the warm-up, the model parameters are updated toward convergence, even if they were randomly initialized. This allows us to obtain stable models.

We generally use a fixed time for the warm-up. For example, Vaswani et al. (2017) used 4,000 updates in their experiments. However, the warm-up time influences the final quality of models.

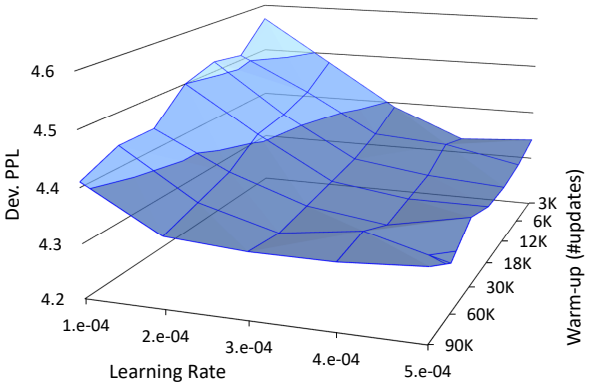

Figure 1: Relationship between learning rate, warm-up, and perplexity of development set in ASPEC.ja-en task.

Figure 1 shows how the development set perplexity (Dev. PPL) changes as the learning rate and warm-up time vary, using the ASPEC.ja-en dataset (as explained in Section 5). Lower perplexity indicates a better model. We can observe that both the learning rate and the warm-up time influence the perplexity. When we use a long warm-up, we can apply high learning rates and consequently obtain low-perplexity models. We observed a similar tendency in the TDDC.ja-en task.

Based on the above experiment, we used 0.0004 as the learning rate and set the warm-up time to 30K updates for ASPEC datasets and 14K updates for TDDC datasets.[2] These values almost minimize the development perplexity.

Recently, a variant of Adam optimization (called RAdam), which automatically adapts the learning rate and does not require any warm-up, has been proposed (Liu et al., 2019). To confirm the relationship between the warm-up and RAdam is our future work.

4 Self-Training Based on Back-Translation

Back-translation is a technique to enhance neural machine translators (NMT), particularly the decoder part of NMT, using monolingual corpora (Sennrich et al., 2016a). It translates sentences of the target language into those of the source language. A forward translator is trained using this pseudo-parallel corpus with corpora that are cre-

[1] https://github.com/pytorch/fairseq

[2] The warm-up time is around five epochs because the mini-batch size is approximately 500 sentences.

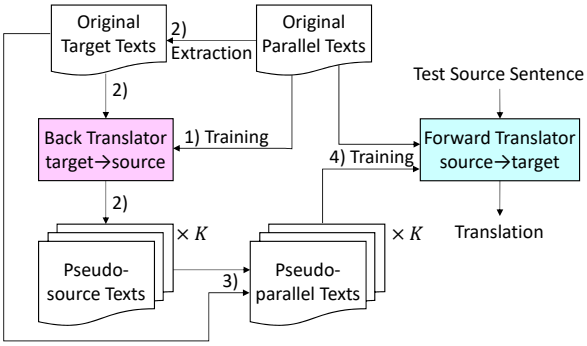

Figure 2: Data flow of self-training based on back-translation.

ated manually. The back-translation can be applied to self-training if the pseudo-parallel corpora are created from the manually created corpora that will be used for training the forward translator (Figure 2).

4.1 Back-Translation with Sampling Generation

A problem with back-translation is that the pseudo-parallel sentences become less varied than those created manually, because of machine translation. This characteristic makes it difficult for the back-translation method to enhance the encoder, in contrast to the decoder.

To solve this problem, Imamura et al. (2018) proposed a method that combines the following two methods.

- To generate diverse pseudo-parallel sentences, words are generated by sampling based on the word probability distribution (Eq. 1) instead of the maximum likelihood during the back-translation.

$$\hat{y}_t \sim \frac{\Pr(y|\boldsymbol{y}_{<t}, \boldsymbol{x})^{1/\tau}}{\sum_{y'} \Pr(y'|\boldsymbol{y}_{<t}, \boldsymbol{x})^{1/\tau}}, \quad (1)$$

  where $\hat{y}_t$, $\boldsymbol{y}_{<t}$, and $\boldsymbol{x}$ denote the generated word at time t, the history until time t, and the input word sequence, respectively. τ denotes the temperature parameter, which is used to control the diversity, but we use $\tau = 1.0$ in this paper.

- Multiple pseudo-source sentences, for a target sentence, are used for training.

Both methods are intended to enhance the encoder by increasing the diversity of source sentences, while fixing the target sentences.

4.2 Training Procedure

The training procedure is summarized as follows (Figure 2).

1) Train a back-translator from the target language to the source language, using the original parallel corpus.

2) Translate the target side of the original corpus to the source language, using the back-translator. During the back-translation, K pseudo-source sentences are generated for each target sentence, using sampling.

3) Construct K sets of pseudo-parallel sentences by pairing the target and pseudo-source sentences.

4) Build the training set by mixing the original and pseudo-parallel corpora, and train the forward translator from the source language to the target language.

4.3 Static and Dynamic Self-Training

There are two types of self-training based on back-translation, depending on the pseudo-parallel sentences and the structure of mini-batches: the static self-training (Imamura et al., 2018) and dynamic self-training (Imamura and Sumita, 2018). Steps 3 and 4 of Section 4.2 are different for each type.

Static self-training constructs a training set by combining K_{static} pseudo-parallel sentences with each of the original sentences. In this paper, we set $K_{static} = 4$. During training, the training set is fixed.

In static self-training, the number of pseudo-parallel sentences is K_{static} times larger than the number of original sentences. If we simply mix these sentences, the ratio of pseudo-parallel sentences to original sentences would be too high. To avoid this problem, we oversample the original sentences by a factor of K_{static}, instead of changing the learning rate depending on the sentence (Imamura et al., 2018). Therefore, the total size of the training set is $2K_{static}$ times larger than the size of the original corpus.

In contrast, dynamic self-training constructs $K_{dynamic}$ training sets. A training set includes one pseudo-parallel set and one original set. For each epoch, a set is randomly selected from the $K_{dynamic}$ training sets, and the model is trained using the set. In this paper, we set $K_{dynamic} = 20$.

In dynamic self-training, the size of a training set is twice the size of the original corpus. Therefore, the training time for an epoch is shorter than that for static self-training. However, a greater number of epochs are required until convergence, because diverse training sets are used.

5 Results of ASPEC and TDDC Tasks

The results of ASPEC and TDDC are shown in Tables 3 and 4, respectively. Both tables show the translation quality (BLEU) and the perplexity of the development set (Dev. PPL), depending on the model type and training method. The effect of the long warm-up has already been shown in Section 3.

5.1 Notes of Experimental Settings

The BLEU scores (Papineni et al., 2002) in the tables were computed based on the tokenizers MeCab (for Japanese (Kudo et al., 2004)) and Moses (for English (Koehn et al., 2007)). We trained four models with different random seeds. The single model rows of the tables show the average score of four models, and the ensemble rows show the score of the ensemble of four models.

The length penalty for testing was set to maximize the BLEU score of the development set. However, in the TDDC task, we used different penalties for the items and texts sets, and independently optimized according to the set.

Finally, we submitted the ensemble models for which the BLEU scores of the development set (in the single model cases) were the highest.

5.2 Results

First, we focus on the ASPEC.en-ja task. In the single model cases, the BLEU scores improved around +0.20 to +0.58 by adding static self-training to the base model. In the case of dynamic self-training, the improvements were between +0.46 and +0.60. The ensemble models have a similar tendency, and we can conclude that self-training is effective because the BLEU scores significantly improved in many cases. Comparing static and dynamic self-training, there were no significant differences, even though the scores of dynamic self-training were higher than those of static self-training.

In contrast, for the ASPEC.ja-en task, the BLEU scores of static self-training were worse than those of the base model, in both the single model and ensemble cases. However, for dynamic self-training, some BLEU scores significantly improved. Dynamic self-training tends to be more effective for the ASPEC tasks.

In terms of the training time, the number of epochs of the static self-training was lower than that of the dynamic self-training in both ASPEC.en-ja and ASPEC ja-en tasks. However, conversely, the total number of updates of the dynamic self-training was lower. As the training data size increased, the total training time increased in the static self-training. The dynamic self-training was more efficient from the perspective of training time.

For the TDDC.ja-en task, the items and texts sets have a different tendency. For the items set, the BLEU scores improved by applying static self-training, but became worse for the texts set.

Self-training based on back-translation is not always effective; that is, there are effective and ineffective datasets. Investigating the conditions that influence translation quality is our future work. Note that this phenomenon is only observed for self-training; the back-translation of additional monolingual corpora has different features.

6 Conclusions

This paper explained the NICT-2 NMT system at WAT-2019. This system employs the Transformer model and applies the following two training strategies.

- We employed the long warm-up strategy and trained the model using a high learning rate.
- We also employed the self-training strategy, which uses multiple back-translations generated by sampling.

Acknowledgments

This work was supported by the "Research and Development of Enhanced Multilingual and Multipurpose Speech Translation Systems" a program of the Ministry of Internal Affairs and Communications, Japan.

References

Kenji Imamura, Atsushi Fujita, and Eiichiro Sumita. 2018. 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.

Task /Lang	Model		Dev. PPL ↓	BLEU ↑ Dev.	DevTest	Test	Remark
ASPEC en-ja	Single	Base	3.21	43.80	43.05	44.11	392K updates (65 epochs)
		Static ST	3.12	44.38 (+)	43.25	44.54 (+)	2,162K updates (45 epochs)
		Dynamic ST	3.14	**44.40** (+)	**43.51** (+)	**44.71** (+)	810K updates (71 epochs)
	Ensemble (4 Models)	Base	–	45.14	44.01	45.13	
		Static ST	–	**45.53**	44.41 (+)	45.64 (+)	
		Dynamic ST	–	45.39	**44.58** (+)	**45.66** (+)	Submitted model
ASPEC ja-en	Single	Base	4.42	27.90	26.12	28.40	161K updates (27 epochs)
		Static ST	4.31	27.05 (-)	25.97	27.39 (-)	798K updates (16 epochs)
		Dynamic ST	4.26	**28.24** (+)§	**26.62** (+)§	**28.57** §	502K updates (42 epochs)
	Ensemble (4 Models)	Base	–	28.94	27.31	**29.41**	
		Static ST	–	28.47	26.73 (-)	28.25 (-)	
		Dynamic ST	–	**29.19** §	**27.59** §	29.40 §	Submitted model

Table 3: Results of ASPEC tasks.

The bold values indicate the highest score among base, static self-training (ST), and dynamic ST. The (+) and (-) symbols denote significant improvement and degradation, respectively, from the base model ($p \leq 0.05$). The § symbol indicates that there is a significant difference between static and dynamic ST cases.

Task /Lang	Model / Training		Dev. PPL	BLEU ↑ Dev. Items	Dev. Texts	DevTest Items	DevTest Texts	Remark
TDDC ja-en	Single	Base	2.76	52.75	**52.45**	54.28	**52.76**	115K updates (42 epochs)
		Static ST	2.74	52.94	51.78 (-)	**54.99** (+)§	51.92 (-)	335K updates (15 epochs)
		Dynamic ST	2.68	**52.95**	51.35 (-)	54.41	51.82 (-)	239K updates (43 epochs)
	Ensemble (4 Models)	Base	–	**54.35**	**54.74**	55.56	**54.98**	Submitted text model
		Static ST	–	54.34	53.53 (-)	**56.60** (+)§	53.81 (-)	Submitted item model
		Dynamic ST	–	54.29	53.04 (-)	55.92	53.97 (-)	

Table 4: Results of TDDC task.

The bold values indicate the highest score among base, static self-training (ST), and dynamic ST. The (+) and (-) symbols denote significant improvement and degradation, respectively, from the base model ($p \leq 0.05$). The § symbol indicates that there is a significant difference between static and dynamic ST cases.

Kenji Imamura and Eiichiro Sumita. 2017. Ensemble and reranking: Using multiple models in the NICT-2 neural machine translation system at WAT2017. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*, pages 127–134, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Kenji Imamura and Eiichiro Sumita. 2018. NICT self-training approach to neural machine translation at NMT-2018. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 110–115, Melbourne, Australia.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of EMNLP 2004*, pages 230–237, Barcelona, Spain.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. *ArXiv*, abs/1908.03265.

Toshiaki Nakazawa, Chenchen Ding, Raj Dabre, Hideya Mino, Isao Goto, Win Pa Pa, Nobushige Doi, Yusuke Oda, Anoop Kunchukuttan, Shantipriya Parida, Ondřej Bojar, and Sadao Kurohashi. 2019. Overview of the 6th workshop on Asian translation. In *Proceedings of the 6th Workshop on Asian Translation*, Hong Kong.

Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. 2016. ASPEC: Asian scientific paper excerpt corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2204–2208, Portorož, Slovenia.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, Pennsylvania, USA.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL-2016, Volume 1: Long Papers)*, pages 86–96, Berlin, Germany.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany.

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