University of Tsukuba's Machine Translation System for IWSLT20 Open Domain Translation Task

Hongyi Cui[⋆ †] **Yizhen Wei**[⋆ †] **Shohei Iida**[†] **Masaaki Nagata**[‡] **Takehito Utsuro**[†]
[†]Graduate School of Systems and Information Engineering, University of Tsukuba, Japan
[‡]NTT Communication Science Laboratories, NTT Corporation, Japan

Abstract

In this paper, we introduce University of Tsukuba's submission to the IWSLT20 Open Domain Translation Task. We participate in both Chinese→Japanese and Japanese→Chinese directions. For both directions, our machine translation systems are based on the Transformer architecture. Several techniques are integrated in order to boost the performance of our models: data filtering, large-scale noised training, model ensemble, reranking and postprocessing. Consequently, our efforts achieve 33.0 BLEU scores for Chinese→Japanese translation and 32.3 BLEU scores for Japanese→Chinese translation.

1 Introduction

In this paper, we introduce University of Tsukuba's submission to the IWSLT20 Open Domain Translation Task. The goal of this shared task is to promote: the research on translation between Asian languages, exploitation of noisy web corpora for machine translation, and smart processing of data and provenance. To have an overview look of the IWSLT20 Open Domain Translation Task, readers may refer to Ansari et al. (2020) for further details. We participated in both Chinese→Japanese and Japanese→Chinese directions.

It is widely acknowledged that a neural machine translation (NMT) system requires a large amount of training data. Meanwhile, the training process of NMT models may consume a long period of time and lots of computing resources. Considering the limitation of our computing power, our goal is to boost the performance of NMT systems with fewer and smaller components that require less time and computing resources. Our models are based on the base Transformer as described in Vaswani et al. (2017) without special parameter fine-tuning. For data preprocessing, firstly, various orthodox methods including punctuation normalization, tokenization as well as byte pair encoding (Sennrich et al., 2016a) which have been widely used in recent researches are applied. Besides, we also apply manual rules, aiming to clean the provided parallel data, the monolingual data and the synthetic data which is generated by ourselves for data augmentation. For the sake of a better use of all provided data, we do a back-translation for either the source-side and the target-side monolingual data. Meanwhile, inspired by noised training method (Edunov et al., 2018; Wu et al., 2019; He et al., 2020), we add noise to the source sentences of the synthetic parallel corpus to make the translation models more robust and to improve its generalization ability. In addition, in inference phrase, we apply the model ensemble strategy while top n-best hypotheses are kept for further multi-features reranking process. At last, post-processing is applied to correct the inconsistent punctuation form.

This paper is organized as follows: in Section 2, we describe our data preprocessing and data filtering. Details of each component of our systems are described in Section 3. The results of the experiments for each component and language pair are summarized in Section 4.

2 Data

For all of our submissions, we only use datasets provided by the organizers. For training parallel data, we use the concatenation of **web_crawled_parallel_filtered** and **existing_parallel**. For data augmentation, since the unaligned data are extremely huge, we choose to use the separate side of the pre-filtered

[⋆] Equal contribution.

sentences (**web_crawled_parallel_unfiltered**) as monolingual data. Also, development dataset provided by the organizer is used for development and model evaluation.

2.1 Data Preprocessing

The provided parallel training data contains different forms of characters, for example, full-width form and half-width form. To get a normalized form, we remove all the spaces between characters and perform NFKC-based text normalization.

Chinese sentences are segmented with the default mode of Jieba[1] and Japanese sentences are segmented with Mecab[2] using mecab-ipadic-NEologd[3] dictionary. To limit the size of vocabularies of NMT models, we use byte pair encoding(BPE) (Sennrich et al., 2016b) with 32K split operations separately for both side.

2.2 Data Filtering

The provided datasets built from the web data are very noisy and can potentially decrease the performance of a system. To get a clean form, we filter the parallel training corpus with the following rules:

- Filter out duplicate sentence pairs.
- Filter out sentence pairs which have identical source-side and target-side sentences.
- Filter out sentence pairs with more than 10 punctuations or imbalanced punctuation ratio.
- Filter out sentence pairs which contains half or more tokens that are numbers or letters.
- Filter out sentence pairs which contain HTML tags or emoji.
- Filter out sentence pairs with wrong languages identified by langid.[4]
- Filter out sentence pairs exceeding length ratio 1.5.
- Filter out sentence pairs with less than 3 words or more than 100 word.

[1] https://github.com/fxsjy/jieba
[2] https://taku910.github.io/mecab
[3] https://github.com/neologd/mecab-ipadic-neologd
[4] https://github.com/saffsd/langid.py

	Ja	**Zh**
Parallel data(in sents.)	20.9M	20.9M
+ Filtering(in sents.)	9.8M	9.8M
+ Filtering(in subwords.)	164.5M	128.1M
Monolingual data(in sents.)	161.5M	161.5M
+ Filtering(in sents)	17.8M	17.6M
+ Filtering(in subwords.)	308.3M	254.9M

Table 1: Statistics of the provided data. Notice that we treat two sides of the provided unfiltered dataset separately as monolingual data, therefore the number of monolingual data in terms of sentence pairs are identical as before data filtering.

The same data filtering strategies except those designed for sentence pairs are also employed on monolingual data. Details of the preprocessed dataset in terms of the amount of sentences and BPE subwords are listed out in millions in Table 1.

3 System

3.1 Baseline System

We adopt the base Transformer as our machine translation system following the settings as described in Vaswani et al. (2017), consisting of 6 encoder layers, 6 decoder layers, 8 heads, with an embedding dimension of 512 and feed-forward network dimension of 2048. The dropout probability is 0.2. For all experiments, we adopt the Adam optimizer (Kingma and Ba, 2014) using β_1 = 0.9, β_2 = 0.98, and ϵ = 1e-8. The learning rate is scheduled using inverse square root schedule with a maximum learning rate 0.0005 and 4000 warmup steps. We train all our models using fairseq[5] (Ott et al., 2019) on two NIVIDA 2080Ti GPUs with a batch size of around 4096 tokens. During training, we employ label smoothing of value 0.1. We average the last 5 model checkpoints and use it for decoding.

3.2 Large-scale Noised Training

It is widely known that the performance of a NMT system relies heavily on the amount of parallel training data. Back-translation (Sennrich et al., 2016a) and Self-training (Zhang and Zong, 2016) are effective and commonly used data augmentation techniques to leverage the monolingual data to augment the original parallel dataset.

In our case, we leverage both the source-side and target-side monolingual data to help the train-

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

ing. Specifically, we train a baseline NMT model with the provided parallel corpus at first. Then, target-side monolingual sentences are translated by a target-to-source NMT model and source-side monolingual sentences are translated by a source-to-target NMT model.

Inspired by noised training method (Edunov et al., 2018; Wu et al., 2019; He et al., 2020), we add noise to the source sentences of the synthetic parallel corpus to make the translation models more robust and to improve its generalization ability. Specifically,

- We randomly replace a word by a special unknown token with probability 0.1.
- We randomly delete the words with probability 0.1.
- We randomly swap the words with constraint that no further than 3 words apart.

Then we add the synthetic parallel data to original parallel data and train a new NMT model.

3.3 Model Ensemble

Model ensemble is a common method to boost translation performance. However, due to the huge amount of training data and our limited computing power, we do not ensemble multiple strong models with different random seeds. Instead, we only combine three models trained on filtered data with different random seeds and one model trained through large-scale noised training. All individual models used for model ensemble are the average of the last 5 model checkpoints.

3.4 Reranking

Reranking is a method of improving translation quality by rescoring a list of n-best hypotheses. For our submissions, we generate n-best hypotheses through a source-to-target NMT model and then train a reranker using k-best MIRA (Cherry and Foster, 2012). The features we use for reranking are:

- **Left-to-right NMT Feature:** We keep the original perplexity by the original translation model as a L2R reranking feature.
- **Right-to-left NMT Feature:** In order to address exposure bias problem, we train a right-to-left (R2L) NMT model using the same training data but with inverted target word order. We invert the hypothesis sequence and use the perplexity score given by the right-to-left NMT model as R2L feature.
- **Target-to-Source NMT Feature:** To reduce inadequate translation, we use the perplexity score given by the target-to-source NMT model as T2S feature. In addition, we also use the score generated by a target-to-source right-to-left model as a reranking feature.
- **Length Feature:** We also design a length feature that quantifies the difference between the ratio of each sentence pair and the optimal ratio. The optimal ratio is determined according to the training parallel corpus.

	Ja→Zh	
System	**Dev**	**Test**
Baseline	28.19	22.0
+ Data filtering	28.60	–
+ Noised training	29.42	–
+ Model ensemble	30.03	–
+ Reranking	30.29	–
+ Postprocessing	30.53	–
Our Submission	–	32.3

Table 2: BLEU scores on Japanese → Chinese.

	Zh→Ja	
System	**Dev**	**Test**
Baseline	27.29	26.3
+ Data filtering	32.37	–
+ Noised training	32.21	–
+ Model ensemble	32.82	–
+ Reranking	33.24	–
+ Postprocessing	33.26	–
Our Submission	–	33.0

Table 3: BLEU scores on Chinese → Japanese.

3.5 Postprocessing

Since we perform NFKC-based text normalization on the training corpus, we also employ a post-processing algorithm on the generated hypothesis. To be more specific, we change half-width punctuations to full-width punctuations.

4 Results

Results and ablations for Ja→Zh and Zh→Ja are shown in Table 2 and Table 3 respectively. We report character-based BLEU calculated with the provided script. Reference for BLEU calculation on development dataset is in the raw form which has not been NFKC-normalized. Each line in the table represents for the result yielded from the model to which techniques declared in current line and all previous lines are added.

4.1 Japanese→Chinese

For Ja→Zh, data filtering improves our baseline performance on development data by + 0.41 BLEU scores. The addition of synthetic data and large-scale noised training further improves model performance by + 0.82 BLEU scores. We further gain + 0.61 BLEU scores and + 0.26 BLEU scores after applying model ensemble and reranking. Finally, applying postprocessing on top of generated hypothesis gives another improvement of 0.24 BLEU scores. The final BLEU score of our submission is 32.3.

4.2 Chinese→Japanese

For Zh→Ja, data filtering plays an important role and improves our baseline performance on development data by + 5.08 BLEU scores. The addition of synthetic data and large-scale noised training slightly hurt the performance.[6] After applying model ensemble and reranking, we further gain + 0.61 BLEU scores and + 0.42 BLEU scores respectively. Finally, applying postprocessing on top of the generated hypothesis gives another 0.02 BLEU scores. The final BLEU score of our submission is 33.0.

5 Conclusion

This paper describes University of Tsukuba's submission to IWSLT20 open domain translation task. We trained standard Transformer models and adopted various techniques for better performance, including data filtering, large-scale noised training, model ensemble, reranking and postprocessing. We demonstrated the effectiveness of our approach and achieved 33.0 BLEU scores for Chinese→Japanese translation and 32.3 BLEU scores for Japanese→Chinese translation.

[6]However, we also notice that, when BLEU is calculated with the NFKC-normalized reference it will be slightly improved if large-scale noised training is added (33.83 vs. 33.97). Therefore, we still adopt large-scale noised training for our final submission.

References

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondrej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, and Changhan Wang. 2020. Findings of the IWSLT 2020 Evaluation Campaign. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT 2020)*, Seattle, USA.

Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *NAACL*, pages 427–436.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *EMNLP*, pages 489–500.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *ICLR*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *ICLR*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL*, pages 48–53.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *ACL*, pages 86–96.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *ACL*, pages 1715–1725.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Lijun Wu, Yiren Wang, Yingce Xia, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2019. Exploiting monolingual data at scale for neural machine translation. In *EMNLP-IJCNLP*, pages 4207–4216.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *EMNLP*, pages 1535–1545.