The NiuTrans Machine Translation Systems for WMT20

Yuhao Zhang[1], Ziyang Wang[1], Runzhe Cao[1], Binghao Wei[1], Weiqiao Shan[1], Shuhan Zhou[1], Abudurexiti Reheman[1], Tao Zhou[1], Xin Zeng[1], Laohu Wang[1], Xiaoqian Liu[1], Xunjuan Zhou[1], Yongyu Mu[1], Jingnan Zhang[1], Yinqiao Li[1], Bei Li[1], Tong Xiao[1,2] and Jingbo Zhu[1,2]

[1]NLP Lab, School of Computer Science and Engineering,
Northeastern University, Shenyang, China
[2]NiuTrans Research, Shenyang, China
yoohao.zhang@gmail.com, wangziyang@stumail.neu.edu.cn
{xiaotong,zhujingbo}@mail.neu.edu.cn

Abstract

This paper describes NiuTrans neural machine translation systems of the WMT20 news translation tasks. We participated in Japanese↔English, English→Chinese, Inuktitut→English and Tamil→English total five tasks and rank first in Japanese↔English both sides. We mainly utilized iterative back-translation, different depth and widen model architectures, iterative knowledge distillation and iterative fine-tuning. And we find that adequately widened and deepened the model simultaneously, the performance will significantly improve. Also, iterative fine-tuning strategy we implemented is effective during adapting domain. For Inuktitut→English and Tamil→English tasks, we built multilingual models separately and employed pretraining word embedding to obtain better performance.

1 Introduction

This paper describes the NiuTrans submissions to the WMT20 news tasks, including English→Chinese (EN→ZH), Tamil→English (TA→EN), Inuktitut→English (IU→EN) and Japanese↔English (JA↔EN) five directions and all of our systems were built with constrained data sets. Some useful methods in the WMT18 (Wang et al., 2018) and WMT19 (Li et al., 2019) submissions are also reused this time, such as model ensemble, knowledge distillation (KD) et al., and we explore some novel approaches this year.

For this participation, we experimented with some deeper and wider Transformer (Vaswani et al., 2017) architectures to get reliable baselines, nucleus sampling (Holtzman et al., 2020) in back-translation to generate more suitable pseudo bilingual sentences, more effectively fine-tuning strategy to adapt domain. Particularly in the low-resources tasks, {TA,IU}→EN, we built multilingual neural machine translation by using some similar language to get better performance and further replaced decoder's word embedding by an English pretraining Transformer language model's which trained by two monolingual in-domain data corpora.

Furthermore, we presented a new fine-tuning pattern which could significantly improve the BLEU score on the test set, and it worked well on all five tasks whether it is a low or rich resource. We carefully rethought this strategy and found the main gain came from domain adaptation and improved inferior translations.

Our systems and this paper followed six main steps:1) data preprocessing and filter, 2) iterative back-translation to generate pseudo bilingual data, 3) using different model architectures to enhance the diversity of translation, 4) iterative knowledge distillation by in-domain monolingual data, 5) iterative fine-tuning with in-domain using small training batch, 6) translation post-process.

2 System Overview

2.1 Data Preprocessing and Filtering

For EN→ZH and JA↔EN tasks, we first normalized the punctuation in Chinese and Japanese monolingual data by using Moses (Koehn et al., 2007) `normalize-punctuation.perl` script. English and Inuktitut sentences were segmented by Moses, while Chinese, Japanese and Tamil used NiuTrans (Xiao et al., 2012), MeCab[1] and IndicNLP[2] separately for word segmentation. After converting numbers and punctuation into English pattern, and then we normalized English words in Japanese sentences to Japanese by using Sudachi (Takaoka et al., 2018).

As previous work (Wang et al., 2018) indicated that it's important to clean data strictly, so this year

[1]https://github.com/taku910/mecab
[2]https://github.com/anoopkunchukuttan/indic_nlp_library

we used a stricter data filter scheme than Li et al. (2019) and the rules were following:

- Filter sentences length ratio lower than 0.4 or upper than 3 and punctuation ratio more than 0.3.
- Remove sentences that have the long word which consist of more 40 characters or words more than 200.
- Remove repeated n-gram translation and repeated sentences except for IU.
- Filter out the sentences whose alignment scores obtained by fast-align are lower than -6.
- Detecting language and delete other languages or have a special HTML label.
- Filter sentences in which parentheses on both sides do not correspond.
- Use Unicode to filter sentences that other characters more than 10.

And when we cleaned monolingual data still employed those rules and particularly there were some lines which include two or more sentences, we write a script to cut them into several sentences.

2.2 Iterative Back Translation

Back-translation is an effective way to boost translation quality by using mono data to produce pseudo training parallel data. Also, it can alleviate domain adapted problems by carefully choosing the in-domain target data. As Edunov et al. (2019); Bogoychev and Sennrich (2019) stated, due to the test target side only consisted of manual translations, back translation didn't bring evident BLEU increase on the test set. Despite our experiments proved that the deeper architectures still showed apparent improvements as the number of data increases.

As Li et al. (2019) stated, it's crucial to select in-domain mono data for back-translation. After picking out English mono data, we first used 50 million news data to train a language model (LM) built with Transformer structures, then ranked cleaned mono data which scored by trained language model before. However, it's hard to find massive in-domain data for other languages to train a neural LM, so the better choice was using a statistical method, in here we selected XenC toolkit[3] (Rousseau, 2013). The in-domain data consisted of the valid set source side and News Commentary high-quality mono data. For avoiding the short sentence ranked too high, each score was multiplied by a length penalty when using both approaches to score these data.

[3]https://github.com/antho-rousseau/XenC

We chose a sample base model as our back-translation model rather ensemble model which may gain a little improvement, but needed spending huge decoding time. For multilingual model back-translation, we followed Johnson et al. (2017)'s work adding a target language label in the source side, so translations could be adapt to the target language.

This year we also followed previous work Edunov et al. (2019) and we added a new pseudo data produce methods–Nucleus Sampling, according to Holtzman et al. (2020)'s work. For all tasks we participated in, we first employed the beam search approach to generate the best translations as pseudo data and the scale of the pseudo was about 1:1 to real data. Then merge those data to retrain model and do back-translations again. Repeated those steps until the valid set BLEU have few increases then stop iterative back-translation processing. Notably, during the second back-translation, for EN→ZH task we used topk sampling and the k is 10 following last year, while for JA↔EN tasks, nucleus sampling method which the p was set 0.9 preferred better comparing topk, whereas for other tasks, {TA,IU}→EN, simply sampling was better.

2.3 Multilingual Model

For TA→EN and IU→EN, building a multilingual model is a simple and effective way to boost performance because of knowledge transfer. For TA→EN task, we added six other similar languages and only one language Russian (RU) for IU→EN task, because there were no other languages witch a relationship with IU. For TA, We up-sampled the TA data then shuffled all the train data so that each training batch could have TA data with high-probability. As for IU, we only added 0.3 million RU high quality data, then we directly merged two languages as training data. To enhance the effect of transfer learning, we utilized only one model which all the language shared the same parameters including word embeddings and vocab. Bilingual data were reused to fine-tune the model for adapting parameters to the target language after model convergence.

Model Tag	Depth	Hidden Size	Filter Size	RPR Attention
Base	6	512	2048	✗
Big	6	1024	4096	✗
Deep25	25	512	2048	✗
Deep25-filter	25	512	4096	✗
Deep30-RPR	30	512	2048	✓
DLCL35-RPR	35	512	2048	✓
DLCL40-RPR	40	512	2048	✓
Deep15-filter-768-RPR	15	768	4096	✓

Table 1: Transformer Architectures.

2.4 Model Architectures and Ensemble

Inspired by deep network Wang et al. (2019), we tried to use simple deep, or deep and wide network architectures based on the Transformer to explore the relationship of performance and model parameters. We mainly carried out experiments on the structures of the model in Table 1. And we kept six decoder layers unchanged because it only could gain a few improvements tough many model parameters increased.

Deep Network: This model structure simply changes encoder layers, hidden size and other hyper-parameters based on vanilla Transformer.

DLCL Network: For a deeper network, we employed DLCL (Wang et al., 2019) to get more diverse models.

Filter size: This hyper-parameter represents the dimension size of feed-forward network (FFN) and simply increasing this could bring some improvements (Wang et al., 2018; Sun et al., 2019; Bawden et al., 2019). Notably, when using the deep Transformer architecture, the training time and model parameters will increase sharply with the augment of the FFN size.

RPR and relative length: The relative position representation (RPR) (Shaw et al., 2018) improves self-attention by adding relative position information. The relative length which we set 8 is the key parameter of this method.

For choosing models to ensemble, we utilized the ensemble search method which used a script to traverse all possible combinations then recorded the best one. For JA↔EN, we chose 6 of 10 while other tasks were 4 of 10.

2.5 Iterative KD and Fine-tuning

Sun et al. (2019) showed the self-learning strategy is a very effective approach to improve performance when the test set only composed of manual translations and we mainly reused (Li et al., 2019) iterative KD strategy to implement self-learning. Specifically, we designed a new iterative fine-tuning process which consists of three steps: 1)using ensemble models to decode valid and test source side sentences then fine-tune models with those pseudo data, 2) fine-tune with the valid set by a small training batch and learning rate, 3) self-learning with in-domain data which chose by only test source side. Repeat these steps two or three times according to the increase of the valid score in the third step. Figure 1 shows these steps. Notably, for being consistent with the composition of the test set, we picked out the data that the source side is real while the target side is manual from the previous valid set. In this way, we found that iterative fine-tuning can promote news title translation quality.

2.6 Reranking

For JA↔EN tasks, we followed the Ng et al. (2019), using a neural language model, and a reverse translation model. Different from the last year, we used several length penalties to generate more candidates.

2.7 Post Editing

For tasks to the English side, we only confirmed the numbers whether to generate correctly by designing a rule-based script which generated two lists for source and target sentences separately. For EN→ZH, the strategy was the same as the last year Li et al. (2019) and particularly dealt with the name's translation by using rules to delete the English name copy in Chinese sentences. For EN→JA task, we transferred English punctuation to Japanese pattern.

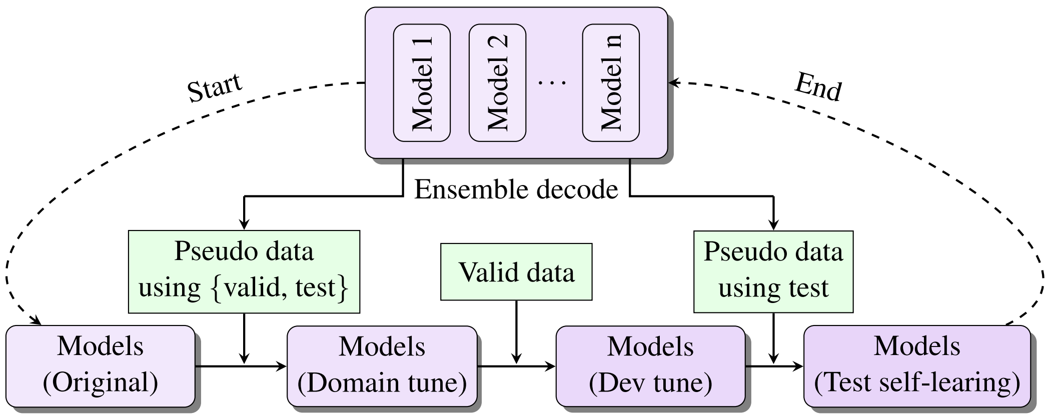

Figure 1: Iterative fine-tuning process

3 Experiment

3.1 Experiment Settings

For all tasks, we implemented the Transformer-Base as our baseline and all of our architectures were pre-normalize Wang et al. (2019) for stable training except Transformer-Big. We implemented models based on Fairseq (Ott et al., 2019) and trained on eight 2080Ti GPUs. We used Adam optimizer (Kingma and Ba, 2014) during training, $\beta_1 = 0.9$, $\beta_2 = 0.997$ for pre-normalize architectures and training batch was 2048 token while we accumulated gradient 4 times for achieving bigger batch size. We shuffled the training data before generate training batch and the training batch each epoch, so we didn't consider the document information. The max learning rate and warmup-steps we set were 0.002 and 8000 separately for deep models but 0.0016 and 16000 for deep and wide models. During training, we used fp16 to accelerate training with few performance damage. Training 15 epoch was enough for most Tasks, while 20 epoch was better for EN$\rightarrow$ZH task and we implemented Li et al. (2020)'s methods to accelerate training. To get more robust models, the last 5 checkpoints were saved every 5000 steps for EN$\rightarrow$ZH and JA$\leftrightarrow$EN tasks but every epoch for TA$\rightarrow$EN,IU$\rightarrow$EN tasks were average ensemble. During back-translation, we followed Hu et al. (2020)'s approaches to accelerate decoding when generating pseudo data.

3.2 JA$\leftrightarrow$EN Results

For JA$\leftrightarrow$EN tasks, we chose ParaCrawl v5.1, New Commentary v15, WikiMatrix, Japanese-English Subtitle Corpus, The Kyoto Free Translation Task Corpus, TED Talks total six parallel data corpus about 14.35 million and News crawl, News Commentary, Common Crawl , TED Talks 4 Japanese monolingual data corpus about 1.7 billion. After the data filter, 12 million parallel data was left and 11 million selected by the neural language model was used as training data. Cleaning several billion low-quality monolingual data will cost too much time, so here we shuffled all the data then split it into dozens of parts, one of which was 20 million. Finally we used total eight of them, each piece was carefully cleaned. Before we also used BPE (Sennrich et al., 2016) models with 32,000 merge operations for both sides to reduce UNK size in vocabulary.

We implemented back-translation two times, the first was beam search while the second was Nucleus Sampling to generate translations. Each time we selected 12 million mono data sampled from all the remaining data. Tough the second time didn't increase significantly compared with the first time, the performance was further improved with the increase of the model parameters. Considering the training time, we finally chose 35 million training data on both sides. Notably, as the official stated that the test target side only consists of manual translations, so the back-translation didn't bring too many improvements, only +0.55 and +2.1 BLEU separately in two tasks.

In order to get more diverse models for ensemble and achieve better results, we trained total 10 models including that eight with different architectures which have been shown in Table 1 and other two with different training data which consisted of 11 million bilingual data and 12 million pseudo data produced by the first back-translation. Then we searched from all the models to find the best combination of 6 out of 10 models. And Table 2 showed that the ensemble is still a robust and effective way

	JA→EN		EN→JA	
System	**Valid**	**Test**	**Valid**	**Test**
Baseline	19.9	20.4	33.2	34.8
+ 12M Beam	21.0	20.8	36.5	36.8
+ 12M Nucleus	21.2	21.0	36.7	36.9
Deep15-filter-768-RPR	23.2	22.9	39.1	39.3
+ Iterative KD	24.4	24.6	39.8	40.1
+ Iterative fine-tuning	25.6	26.2	40.7	41.6
+ Ensemble	25.8	26.5	41.1	41.9
+ Post Edit	26.4	26.6	42.1	42.8

Table 2: BLEU scores on JA↔EN tasks

to boost translation quality.

We implemented iterative KD process twice and each time chose 0.3 million monolingual data using ensemble model to decode then trained 3 to 5 epoch according to the dev PPL. Then we iteratively fine-tuned the models three and two times for JA→EN and EN→JA separately. And interestingly in some real case, the translation of the news titles was significantly improved after iteratively fine-tuning.

As Table 2 shows, iterative KD and fine-tuning strategies could significantly increase the BLEU on the test set.

We used the reranking model like Ng et al. (2019), though it could boost 0.3 BLEU on dev set, it didn't get benefits on the test set. During post edit, we mainly checked the number according to the source side, it also could on EN→JA task.

3.3 EN→ZH Results

In EN→ZH task, we employed News Commentary v15, UN Parallel Corpus V1.0, Back-translated news, CCMT Corpus total four corpora, and after data filter, 10 million data were sampled to train out baseline model. We set wmt18 and wmt19 test as the valid set and mainly referred wmt19 set. In the back-translation, 10 million mono data were sampled from News crawl, News Commentary and Common Crawl three corpora then used the baseline model decode by beam search strategy during the first time. During the second time, we still utilized the same amount of pseudo data while topk sampling which the k is 10 were used to translation mono sentences. From Table 3, we could find that back-translations didn't perform well. Finally 30 million data in total were used to train 10 models, different from other tasks, here we searched the best two combinations of 4 out of 10 models for iterative KD strategy to ensure the diversity of models.

System	**news2019**	**news2020**
Baseline	35.4	40.8
+ 10M Beam	36.3	41.6
+ 10M TopK	36.1	41.5
Dlcl25-RPR	38.7	44.2
+ Iterative KD	39.4	45.4
+ Iterative fine-tuning	39.8	45.9
+ Ensemble	40.1	46.7
+ Post Edit	40.3	47.3

Table 3: BLEU (%) scores on EN→ZH task

Then we implemented three times iterative KD and each time sampled 10 million in-domain source data. Table 3 showed that it's a very effective method to get 0.8 improvements. Furthermore, we fine-tuned models iteratively three times to domain adaptation and improved +0.5 BLEU. Due to implementing two ensemble combinations to decode sentences, at last model ensemble was still effective to gain 0.8 improvement. According to the WMT19 test, we adjusted the name's translations pattern during the post edit step then resulting in a 0.6 BLEU performance increase.

3.4 IU→EN Results

In IU→EN task, we only used Nunavut Hansard Inuktitut-English Parallel Corpus 3.0 total 1.3 million sentences. After the data filter, 1.1 million data was left to build the baseline model. Though romanization Inuktitut data directly was not effective, it performed better than baseline when build a multilingual system by adding 0.3 million Russia data which has the most similar semantic with Inuktitut. After that, we implemented data augmentation

System	Valid	Test
Baseline	29.6	21.3
+ Romanization	29.3	21.0
Multilingual baseline	30.2	21.6
+ 1.3M Beam	30.6	22.2
+ 1.3M Sampling	30.9	22.2
Deep15-filter-768-RPR	32.5	23.5
+ Knowledge Distillation	32.5	25.4
+ Iterative fine-tuning	49.3	28.4
+ Ensemble	49.3	28.6
+ Post Edit	49.7	29.1

Table 4: BLEU (%) scores on IU→EN task

by using the multilingual model to back-translate mono data iteratively twice and each time using 1.1 million data which equaled the true training sentences. Interestingly, the bigger and wider models improved translation quality distinctly proving it's a robust way whether the training data is rich or not.

Then we first fine-tuned the multilingual model to the target language by using 1.1 million bilingual data several epochs. According to the valid source set, ensemble models were used to decode monolingual in-domain 0.1 million data which was chosen by XenC and gained 1.85 BLEU improvement. Then fine-tuned models only once, because different from the bilingual model, the multilingual model didn't perform very well during the fine-tuning stage.

Finally we selected four models to ensemble and gained 0.18 increase, because different models were too similar after fine-tuning. And we fixed the punctuation and the score improved 0.52 BLEU. During the post process, we fixed the number and punctuation translation.

3.5 TA→EN Results

The Ta→EN task is similar to IU→EN but more complicated, because more data corpus and language can be used to build the multilingual system. Specifically, we total used {Hindi (HI), Kannada (KN), Malayalam (ML), Punjabi (PA), Telugu (TE), Urdu (UR)}→EN total six other languages, 17 million sentences according to Kudugunta et al. (2019)'s work showed similar languages with TA. From Table 5, it can be seen that using similar languages to build a multilingual system can indeed improve the performance. Also, using iterative back-translation is still an effective way but couldn't add too much pseudo language data because this will make the real target language data account for the whole data was too small, which leaded to performance damage. During the back-translation process, due to too many languages in one model, we followed Johnson et al. (2017)'s approach to build a reverse model to ensure translation quality.

System	Valid	Test
Baseline	12.8	13.2
Multilingual baseline	14.2	15.1
+ 0.5M Beam	19.2	15.7
+ 1M Beam	20.9	16.6
Deep15-filter-768-RPR	22.8	19.0
+ Knowledge Distillation	23.4	20.6
+ Iterative fine-tuning	23.6	20.7
+ Ensemble	23.8	21.0

Table 5: BLEU (%) scores on TA→EN task

For the model architectures we used, the wide and deep model was still very effective and improved 2.33 BLEU comparing with the base model. Also it performed better than simple deepen model layers. After finishing KD and fine-tuning, finally gain 1.92 improvements.

4 Conclusions

This paper introduced our submissions on WMT20 five tasks and our main exploration is using more diversified architectures, improving a iterative fine-tuning strategy and utilizing several similar languages to build a multilingual model on low-resource tasks. And we experimented with iterative back-translation by different decoding strategies, using pre-trained embeddings in multilingual models. On the whole, all of our systems performed competitively and ranked 1st on JA↔EN both sides.

Acknowledgments

This work was supported in part by the National Science Foundation of China (Nos.61876035 and 61732005) and the National Key R&D Program of China (No.2019QY1801).

References

Rachel Bawden, Nikolay Bogoychev, Ulrich Germann, Roman Grundkiewicz, Faheem Kirefu, Antonio Valerio Miceli Barone, and Alexandra Birch. 2019.

The University of Edinburgh's submissions to the WMT19 news translation task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 103–115, Florence, Italy. Association for Computational Linguistics.

Nikolay Bogoychev and Rico Sennrich. 2019. Domain, translationese and noise in synthetic data for neural machine translation. *arXiv preprint arXiv:1911.03362*.

Sergey Edunov, Myle Ott, Marc'Aurelio Ranzato, and Michael Auli. 2019. On the evaluation of machine translation systems trained with back-translation. *arXiv preprint arXiv:1908.05204*.

Ari Holtzman, Jan Buys, Leo Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *ICLR 2020 : Eighth International Conference on Learning Representations*.

Chi Hu, Bei Li, Yinqiao Li, Ye Lin, Yanyang Li, Chenglong Wang, Tong Xiao, and Jingbo Zhu. 2020. The NiuTrans system for WNGT 2020 efficiency task. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 204–210, Online. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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.

Sneha Kudugunta, Ankur Bapna, Isaac Caswell, and Orhan Firat. 2019. Investigating multilingual NMT representations at scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1565–1575, Hong Kong, China. Association for Computational Linguistics.

Bei Li, Yinqiao Li, Chen Xu, Ye Lin, Jiqiang Liu, Hui Liu, Ziyang Wang, Yuhao Zhang, Nuo Xu, Zeyang Wang, Kai Feng, Hexuan Chen, Tengbo Liu, Yanyang Li, Qiang Wang, Tong Xiao, and Jingbo Zhu. 2019. The niutrans machine translation systems for wmt19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 257–266.

Bei Li, Ziyang Wang, Hui Liu, Yufan Jiang, Quan Du, Tong Xiao, Huizhen Wang, and Jingbo Zhu. 2020. Shallow-to-deep training for neural machine translation. *arXiv preprint arXiv:2010.03737*.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook fair's wmt19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319.

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-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 48–53.

Anthony Rousseau. 2013. Xenc: An open-source tool for data selection in natural language processing. *The Prague Bulletin of Mathematical Linguistics*, (100):73–82.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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)*, volume 1, pages 1715–1725.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Meng Sun, Bojian Jiang, Hao Xiong, Zhongjun He, Hua Wu, and Haifeng Wang. 2019. Baidu neural machine translation systems for wmt19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 374–381.

Kazuma Takaoka, Sorami Hisamoto, Noriko Kawahara, Miho Sakamoto, Yoshitaka Uchida, and Yuji Matsumoto. 2018. Sudachi: a japanese tokenizer for business. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Paris, France. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International*

Conference on Neural Information Processing Systems, pages 5998–6008.

Qiang Wang, Bei Li, Jiqiang Liu, Bojian Jiang, Zheyang Zhang, Yinqiao Li, Ye Lin, Tong Xiao, and Jingbo Zhu. 2018. The niutrans machine translation system for wmt18. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 528–534.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy. Association for Computational Linguistics.

Tong Xiao, Jingbo Zhu, Hao Zhang, and Qiang Li. 2012. NiuTrans: An open source toolkit for phrase-based and syntax-based machine translation. In *Proceedings of the ACL 2012 System Demonstrations*, pages 19–24, Jeju Island, Korea. Association for Computational Linguistics.