CASIA's System for IWSLT 2020 Open Domain Translation

Qian Wang[1,2], **Yuchen Liu**[1,2], **Cong Ma**[1,2], **Yu Lu**[1,2], **Yining Wang**[1,2], **Long Zhou**[1,2]
Yang Zhao[1,2], **Jiajun Zhang**[1,2], **Chengqing Zong**[1,2]
[1]National Laboratory of Pattern Recognition, CASIA, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

Abstract

This paper describes the CASIA's system for the IWSLT 2020 open domain translation task. This year we participate in both Chinese→Japanese and Japanese→Chinese translation tasks. Our system is neural machine translation system based on Transformer model. We augment the training data with knowledge distillation and back translation to improve the translation performance. Domain data classification and weighted domain model ensemble are introduced to generate the final translation result. We compare and analyze the performance on development data with different model settings and different data processing techniques.

1 Introduction

Neural machine translation(NMT) has been introduced and made great success during the past few years (Sutskever et al., 2014; Bahdanau et al., 2015; Luong et al., 2015; Wu et al., 2016; Gehring et al., 2017; Zhou et al., 2017; Vaswani et al., 2017). Among those different neural network architectures, the Transformer, which is based on self-attention mechanism, has further improved the translation quality due to the ability of feature extraction and word sense disambiguation (Tang et al., 2018a,b). In this paper, we describe our Transformer based neural machine translation system submitted to the IWSLT 2020 Chinese→Japanese and Japanese→Chinese open domain translation task (Ansari et al., 2020).

Our system is built upon Transformer neural machine translation architecture. We also adopt Relative Position (Shaw et al., 2018) and Dynamic Convolutions (Wu et al., 2019) to investigate the performances of advanced model variations. For the implementation, we extend the latest release of Fairseq[1] (Ott et al., 2019).

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

For data pre-processing, we use byte-pair encoding(BPE) segmentation (Sennrich et al., 2016b) for the source side and character level segmentation for the target side to improve the model performance on rare words. We also investigate the influence of different segmentation methods including word, BPE and character segmentation for both sides.

To further improve the translation quality, we utilize data augmentation techniques of back-translation with a sub-selected monolingual corpus to build additional pseudo parallel training data. Sentence level knowledge distillation is used to strengthen the performance of student model from multi-policy teacher models including left→right, right→left, source→target and target→source.

We also investigate the domain information of the large training data by using a Bert based domain classifier, which is a masked language model and has been shown effective in large scale text classification tasks (Devlin et al., 2019). With the in-domain data, we transfer the model of general domain to each specific domain, and use weighted domain model ensemble as decoding strategy.

2 System Description

Figure 1 depicts the whole process of our submission system, in which we pre-process the provided data and train our advanced Transformer models on the bilingual data together with synthetic corpora from back-translation and knowledge distillation. With domain classification and fine tuning techniques, we obtain multiple models for ensemble strategy and post-processing. In this section, we will introduce each process step in detail.

2.1 NMT Baseline

In this work, we build our model based on the powerful Transformer (Vaswani et al., 2017). The Transformer is a sequence-to-sequence neural

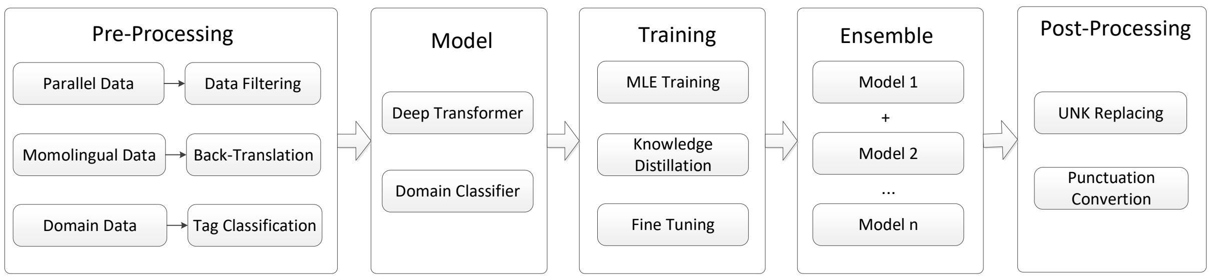

Figure 1: System process of our submissions.

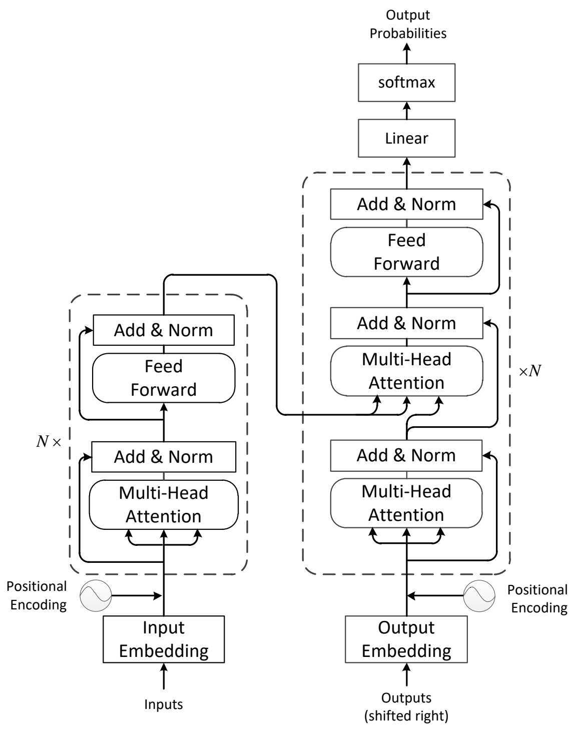

Figure 2: The Transformer model.

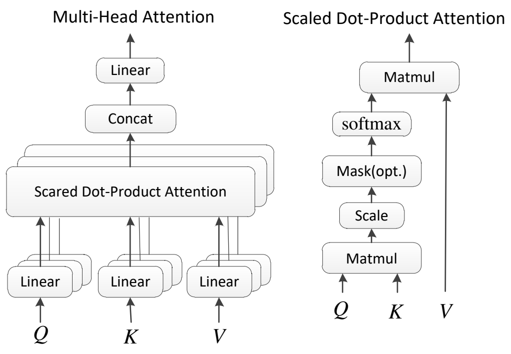

Figure 3: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention.

model that consists of two components: the encoder and the decoder, as shown in Figure 2. The encoder network transforms an input sequence of symbols into a sequence of continues representations. The decoder, on the other hand, produces the target word sequence by predicting the words using a combination of the previously predicted word and relevant parts of the input sequence representations. Particularly, relying entirely on the multi-head attention mechanism, the Transformer with beam search algorithm achieves the state-of-the-art results for machine translation.

Multi-Head Attention We use the multi-head attention with h heads, which allow the model to jointly attend to information from different representation subspaces at different positions. Formally, multi-head attention first obtains h different representations of (Q_i, K_i, V_i). Specifically, for each attention head i, we project the hidden state matrix into distinct query, key and value representations $Q_i{=}QW_i^Q$, $K_i{=}KW_i^K$, $V_i{=}VW_i^V$ respectively. Then we perform **scaled dot-product attention** for each representation, concatenate the results, and project the concatenation with a feed-forward layer.

$$\text{MultiHead}(Q,K,V) = \text{Concat}_i(head_i)W^O$$
$$head_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (1)$$

where W_i^Q, W_i^K, W_i^V and W^O are parameter matrices .

Scaled Dot-Product Attention An attention function can be described as a mapping from a query and a set of key-value pairs to an output. Specifically, we can multiply query Q_i by key K_i to obtain an attention weight matrix, which is then multiplied by value V_i for each token to obtain the self-attention token representation. As shown in Figure 3, we compute the matrix of outputs as:

$$\text{Attention}(Q,K,V) = \text{Softmax}(\frac{QK^T}{\sqrt{d_k}})V \quad (2)$$

where d_k is the dimension of the key. For the sake of brevity, we refer the reader to Vaswani et al. (2017) for more details.

2.2 Back-Translation

Back-translation is an effective and commonly used data augmentation technique to incorporate monolingual data into a translation system (Sennrich et al., 2016a; Zhang and Zong, 2016). Especially for low-resource language tasks, it is indispensable to augment the training data by mixing the pseudo corpus with the parallel part. Back-translation first trains an intermediate target-to-source system that is used to translate monolingual target data into additional synthetic parallel data. This data is used in conjunction with human translated bitext data to train the desired source-to-target system.

How to select the appropriate sentences from the abundant monolingual data is a crucial issue due to the limitation of equipment and huge overhead time. We trained a n-gram based language model on the target side of bilingual data to score the monolingual sentences for each translation direction.

Recent work (Edunov et al., 2018) has shown that different methods of generating pseudo corpus made discrepant influence on translation performance. Edunov et al. (2018) indicated that sampling or noisy synthetic data gives a much stronger training signal than data generated by beam search or greedy search. We adopt the back-translation script from fairseq[2] and generate back-translated data with sampling for both translation directions.

2.3 Knowledge Distillation

The goal of knowledge distillation is to deliver a student model that matches the accuracy of a teacher model (Kim and Rush, 2016). Prior work (Yang et al., 2018) demonstrates that student model can surpass the accuracy of the teacher model. In our experiments, we adopt sequence-level knowledge distillation method and investigate four different teacher models to boost the translation quality of student model.

S2T+L2R Teacher Model: We translate the source sentences of the parallel data into target language using our source-to-target (briefly, S2T) system described in Section 2.1 with left-to-right (briefly, L2R) manner.

S2T+R2L Teacher Model: We translate the source sentences of the parallel data into target language using our S2T system with right-to-left (briefly, R2L) manner.

[2]https://github.com/pytorch/fairseq/tree/master/examples/backtranslation

T2S+L2R Teacher Model: We translate the target sentences of the parallel data into source language using our target-to-source (briefly, T2S) system with L2R manner.

T2S+R2L Teacher Model: We translate the target sentences of the parallel data into source language using our T2S system with R2L manner.

In the final stage, we use the combination of the translated pseudo corpus to improve the student model. It is worth noting that we also mix the original bilingual sentences into these pseudo training corpus.

2.4 Model Ensemble and Reranking

Model ensemble is a method to integrate the probability distributions of multiple models before predicting next target word (Liu et al., 2018). We average the last 20 checkpoints for single model to avoid overfitting. One checkpoint is saved per 1000 steps. For model ensemble, we train six separate models. To achieve this, we fine-tune our student model described in Section 2.3 and back translation model described in Section 2.2 using corpus from three different domains (Spoken domain, Wiki domain and News domain). We use weighted ensemble to generate the translation result, in which the weights for each domain model is calculated from a Bert based domain classifier. The domain specific data for training the domain classifier and fine tuning the student translation model will be described in detail in Section 3.4.

For reranking, we rescore 50-best lists output from the ensemble model using a rescoring model, which includes the models we trained with different model sizes, different corpus portions and different token granularities.

3 Data Preparation

This section introduces the methods we employ to prepare the provided parallel data (18.9M *web_crawled* corpus and 1.9M *existing_parallel* sources) and monolingual sentences (*unaligned_web_crawled* data). We also describe how to prepare domain specific data to facilitate translation.

The provided parallel corpus *existing_parallel* for the two translation directions consists of around 1.9M sentence pairs with around 33.5M characters (Chinese side) in total. Furthermore, a large, noisy set of Japanese-Chinese segment pairs built from web data *web_crawled* is also provided, which consists of around 18.9M sentence pairs with around 493.9M characters (Chinese side) in total. We use the provided development dataset as the validation set during training, which consists of 5,304 sentence pairs. The average length and length ratio of the provided parallel corpus and development dataset is shown as in Table 1.

3.1 Pre-processing and Post-processing

In the open domain translation task both on Chinese→Japanese and Japanese→Chinese translation directions, we first implement pre-processing on training corpus and then filter it.

Before pre-processing, We remove illegal sentences in the provided Japanese-Chinese parallel corpus which include duplicate sentences and sentences in different languages other than source or target (filtered by our language detector tools).

Pre-processing steps include escape character transformation, text normalization, language filtering and word segmentation. There are lots of escape characters in the *existing_parallel* and *web_crawled* which do not occur in development set. As a result, we transform all these escape characters into corresponding marks with a well designed rule-based method to make it consistent between the training and evaluation.

Text normalization step mainly focuses on normalization of numbers and punctuation. Based on analysis on development set, we found that in Chinese, most of the punctuation are double byte characters (DBC), while most of the numbers are single byte characters (SBC). However, most of the numbers and punctuation in Japanese are double byte characters (DBC). Hence we normalize the numbers and punctuation format to make it the same as development set.

In word segmentation step, we apply Jieba[3] as our Chinese word segmentation tool for segmenting Chinese parallel data and monolingual data. For Japanese text, word segmentation is used Mecab (Toshinori Sato and Okumura, 2017). After pre-processing, we filter the training corpus as mentioned in section 3.2.

Finally, we apply Byte Pair Encoding (BPE) (Sennrich et al., 2016b) on both Chinese and Japanese text. Separate BPE models are trained for Chinese and Japanese respectively. Based on the comparison of BPE operations from 30k, 35k, 40k, 45k, 50k, we determine to use 40k BPE operations in source language since it has the best performance on preliminary machine translation experiments. For target side, we determine to use character granular because character level decoder could perform better in our preliminary experiments.

Post-processing steps are similar to pre-processing without filtering. We apply escape character transformation, text normalization and unknown words (UNK) processing steps on machine translation results. The same methods are used to implement escape character transformation and text normalization as pre-processing. For UNK processing, we find some of the numbers can not be well translated by model and we replace these UNKs with the numbers in source sentence. Otherwise, we remove the UNK symbols.

Length Statistic	**Train**		**Dev**	
	Ja	Zh	Ja	Zh
avg. length	17.56	14.52	10.12	7.82
avg. ratio	1.35		1.34	

Table 1: The average length and length ratio (Ja/Zh) of the provided parallel corpus and development dataset.

Filtering Methods	# of sentences
original	20,929,833
remove illegal	18,073,574
filter by length and ratio	15,708,757
filter by alignment	15,679,247

Table 2: The number of the remaining sentence pairs after each filtering operation.

3.2 Parallel Data Filtering

The following methods are applied to further filter the parallel sentence pairs.

We remove sentences longer than 50 and select the parallel sentences where the length ratio (Ja/Zh) is between 0.53 and 2.90. We then calculate word alignment of each sentence pair by using fast_align[4](Dyer et al., 2013). The percentage of aligned words and alignment perplexities are used as the metric where the thresholds are set as 0.4 and −30 respectively. Through the above filtering procedure, the number of the remaining data is reduced from 20.9M to 15.7M, as shown in Table 2.

3.3 Monolingual Data Filtering

It is proven that back-translation is a simple but effective approach to enhance the translation quality

[3]https://github.com/fxsjy/jieba

[4]https://github.com/clab/fast_align

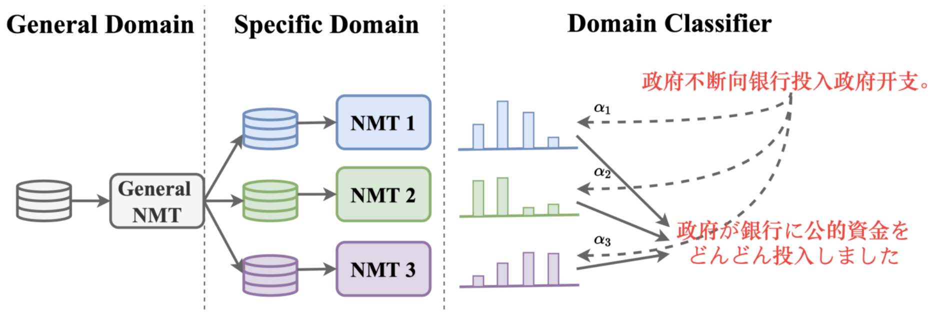

Figure 4: The domain data processing steps, including the NMT model trained on general domain data, the NMT models fine tuned on specific domain, the domain classification and weighted ensemble in the decoding stage.

Filtering Methods	Ja	Zh
original	941,297,925	928,670,666
remove illegal	10,078,827	32,644,917
filter by length	8,175,157	30,415,964
filter by LM	6,128,443	16,374,195

Table 3: The number of the remaining monolingual sentences for Japanese and Chinese after each filtering operation.

Domain	Existing	Web
Wiki	558,531	4,006,232
Spoken	1,290,796	9,534,754
News	21,661	2,444,884

Table 4: Statistics of domain data. Existing indicates *existing_parallel* which is used to train the domain classifier, while Web means *web_crawled_parallel* in which the domain labels are predicted by the classifier.

as described in Section 2.2. To achieve that, we extract the high-quality monolingual sentences from the provided *unaligned_web_crawled* data. After removing illegal sentences from *web_crawled* corpus, we limit the maximum sentence length as 50 and remove dirty data by a language model. Specially, we use KenLM[5] toolkit to train two language models with Japanese and Chinese monolingual data extracted from the provided parallel corpus *existing_parallel*. We then rank the sentences based on the perplexities calculated by the trained language models and filter by perplexity threshold of 4 for Chinese and 3 for Japanese. Note that the perplexities are normalized by sentence lengths. obtain 6.1M and 16.4M monolingual sentences for Japanese and Chinese separately. The filtering results are presented in Table 3.

The obtained monolingual sentences are fed to the trained model to generate pseudo parallel sentence pairs, which are employed to boost the performance of the model.

3.4 Domain Data Processing

Although the amount of provided training data is large enough, it is a noise set of web data built from multiple domain sources. Koehn and Knowles (2017) have demonstrated that the NMT model performs poorly when the test domain does not match the training data. Only the same or similar corpora are typically able to improve translation performance. Therefore, we apply domain adaptation methods in this task.

Adaptation methods for neural machine translation have attracted much attention in the research community (Britz et al., 2017; Wang et al., 2017; Chu and Wang, 2018; Zhang and Xiong, 2018; Wang et al., 2020). They can be roughly classified into two categories, namely data selection and model adaptation. The former focuses on selecting the similar training data from out-of-domain parallel corpora, while the latter focuses on the internal model to improve model performance. Following these two categories, our domain data processing takes the following steps, as shown in Figure 4.

Domain Label In this task, there are two kinds of domain labels provided: domains in *existing_parallel* and domains in *web_crawled_parallel*. Since the later is mainly source document index for each sentence pair, the former is more meaningful for domain classification. We categorize the domain label of *existing_parallel* data into three commonly used classes, namely *Wiki, Spoken, and News*. The domain *Wiki* includes *wiki_facebook*, *wiki_zh_ja_tallip2015* and *wiktionary*. The label *Spoken* includes *ted* and *opensubtitles*. The label *News* includes *global-voices*, *newscommentary* and *tatoeba*.

[5]`https://github.com/kpu/kenlm`

Domain classification Data selection can be conduct in supervised or unsupervised manners (Dou et al., 2019). Since there is a provided data source descriptive file in the *existing_parallel* data which can be regarded as domain labels, we choose the supervised way here. We use two BERT models pretrained on Chinese[6] and Japanese[7] data, respectively. Then the BERT models are fine tuned as a text classification task, based on the source and target side of *existing_parallel* with three domain label we defined. Since the domain data is uneven, we also adopt oversampling and use extra data to enlarge *News* domain For the remaining data in *web_crawled_parallel*, we use the classification model to classify the total data into the three different domains. The statistics of domain data we used is shown in Table 4.

Decoding Stage Considering the test set is also composed of a mixed-genre data, we first classify the domain of each sentence in the test set and obtain the probabilities corresponding to each domain. Then we apply a weighted ensemble method to integrate NMT models. Specifically, when computing the output probability of the next word, we multiply the output probability in each domain specific translation model with the corresponding domain probability of each sentence.

3.5 Other Data Resource

The task description says that the test data is a mixture of genres but the provided development set is mainly from spoken domain. Furthermore, we find that the domain distribution of the training data is severely unbalanced (as shown in Table 4). Especially, the data of *News* domain is quite limited. Due to above two reasons, we decided to crawl some data from other domains.

It is easy to find that *hujiangjp* [8] which is a website helping people to study foreign languages contains some parallel Chinese-Japanese sentences. Accordingly, we crawled all the available data in this website before test data release. The total amount of extra data consists of $12,665$ parallel sentences. We randomly select $4,877$ sentence pairs to build an extra development set. When training each domain model, all the extra data are used as part of *News* domain. We find that 383 Chinese$\rightarrow$Japanese pairs and 421 Japanese$\rightarrow$Chinese pairs in the crawled data are overlapped with the final test set. We just used the originally trained model to decode the test set and decided not to retrain our model since it will take much time and the organizers remind that models cannot be changed after the test set is released. Anyway, we also suggest to test the translation quality on the remaining test set excluding the overlapped sentences.

4 Experiment Settings and Results

4.1 Experiment Setup

Our implementation of Transformer model is based on the latest release of Fairseq. We use Transformer-Big as basic setting, which contains layers of $N = 6$ for both encoder and decoder. Each layer consists of a multi-head attention sublayer with heads $h = 16$ and a feed-forward sublayer with inner dimension $d_{ff} = 4096$. The word embedding dimensions for source and target and the hidden state dimensions d_{model} are set to 1024. In the training phase, the dropout rate P_{drop} is set to 0.1. In the fine tuning phase, the dropout rate is changed to 0.3 to prevent over-fitting.

We use cross entropy as loss function and apply label smoothing of value $\epsilon_{ls} = 0.1$. For the optimizer, we use Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-8}$. The initial learning rate is set to 10^{-4} for training and 10^{-5} for fine tuning.

The models with complete training data are trained on 4 GPUs for 100,000 steps. For the dataset with knowledge distillation or back-translation, the models are trained for 150,000 steps. We validate the model every 1,000 mini-batches on the development data and perform early stop when the best loss is stable on validation set for 10,000 steps. At the end of training phase, we average last 20 checkpoints for each single model of general domain. In fine tuning phase, we use the averaged model of general domain as starting point for initializing the domain model, and continue training on 1 GPU with domain data for 50,000 steps without early stop. The batch sizes in training and fine tuning are set to 32768 and 8192 respectively.

4.2 Result

Table 5 shows the result on development data of both Chinese$\rightarrow$Japanese (ZH$\rightarrow$JA) and

[6]https://github.com/ymcui/Chinese-BERT-wwm
[7]https://github.com/cl-tohoku/bert-japanese
[8]https://https://jp.hjenglish.com/new/

Settings	ZH→JA	JA→ZH
Single System		
Baseline	28.07	22.19
Complete Parallel Data	27.38	27.41
+Parallel data Filtering	33.46	27.69
+Back Translation	34.42	28.08
+Knowledge Distillation	34.00	29.50
+Domain Classification	34.96	30.14
System Combination		
Ensemble Baseline	34.79	30.32
+ Weighted Ensemble	**35.41**	**30.55**
+ Reranking	34.92	30.41

Table 5: The BLEU scores of both directions on development data.

Japanese→Chinese (JA→ZH) translation directions. We report the character BLEU score calculated with *multi-bleu-detok.perl* script. As shown in the result, filtering with complete parallel data plays an important role in our system. Techniques of back translation and knowledge distillation consistently improve the BLEU score. When applying domain classification, we classify each sentence using the Bert-based domain classifier and decode each sentence with corresponding domain model.

As for combination methods, we build six separate models with three domain (*Wiki*, *Spoken* and *News*) fine tuned on two large synthetic data (back translation and knowledge distillation). In ensemble baseline, all of these models share the same weight in predicting word distributions. Weighted ensemble indicates we apply different weights for the ensemble models, in which the weights are obtained by the domain classifier. With weighted domain ensemble, our system achieves the best performance on development data in terms of BLEU, and surpass the single baseline systems by 7.34 BLEU for Chinese→Japanese and 8.36 BLEU for Japanese→Chinese.

We also find a performance drop with reranking. The reason may be that we train the reranking models on the complete parallel data, which is from general domain and may assign lower score for domain specific translations. As a result, our submission is based on the weighted ensemble system, which performs best in our experiments.

4.3 Analysis

We compare the performance of different model variations and token granularities on Chinese→Japanese development data. The data we used to train the models is *existing parallel data*, which consists of 1.9M parallel sentences.

Model Architecture	BLEU
Dynamic Convolutions (Big)	27.13
Transformer (Base)	27.16
Relative Position (Big)	27.41
Transformer (Big)	**27.89**

Table 6: The BLEU scores of Chinese→Japanese on development data with different model settings and variations.

Token Granularities	BLEU
Word→Word	25.45
Character→Character	26.92
BPE→BPE	27.89
BPE→Character	**28.07**

Table 7: The BLEU scores of Chinese→Japanese on development data with different token granularities.

For the model variations, we compare Relative Position (Shaw et al., 2018), Dynamic Convolutions (Wu et al., 2019) and Transformer Base and Big settings (Vaswani et al., 2017). As shown in Table 6, The best result is produced by Transformer Big setting, which is used as default when training on large datasets.

For the token granularities, we report the result with four tokenization methods: Word→Word, Character→Character, BPE→BPE and BPE→Character. As shown in table 7, adopting BPE in source side and Character in target side performs better than other token granularities, which is used in our submission systems.

We notice that there exits a large divergence between the two translation directions when using complete parallel data and process with parallel data filtering. We have verified the result and the parallel data in depth. We find that the quality of Japanese data is lower. For example, there are sentences consist of punctuations only, which may harm the target side language model learned by the decoder. After parallel data filtering, the invalid sentences are removed and thus the translation quality of ZH-JA is improved.

We also find that the provided development data is mainly from spoken domain, and thus we use our collected data as extra development set from other domain to investigate the general performance of single model. The result is shown in table 8. We

Development data	ZH→JA	JA→ZH
Provided	34.00	29.50
Our collected	32.78	30.95

Table 8: The BLEU scores of best single model (with knowlegde distillation) on provided development data and our development data.

find there exists a small gap between provided development data and our collected data, which indicates that the domain information may further improve the translation quality, and thus leads us to utilize domain transfer and ensemble techniques. Note that the extra development set is only used in single models. When it comes to system combination, these data are added into News domain since the size of News domain data in parallel dataset is extremely smaller than other domains (Section 3.4).

5 Conclusion

We present the CASIA's neural machine translation system submitted to IWSLT 2020 Chinese→Japanese and Japanese→Chinese open domain translation task. Our system is built with Transformer architecture and incorporating the following techniques:

- Deliberate data pre-processing and filtering
- Back-translation of selected monolingual corpus
- Knowledge distillation from multi polity teacher models
- Domain classification and weighted domain model ensemble

As a result, our final system achieves substantial improvements over baseline system.

6 Acknowledgement

The research work described in this paper has been supported by the Natural Science Foundation of China under grant No.U1836221 and No.61673380.

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.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Denny Britz, Quoc Le, and Reid Pryzant. 2017. Effective domain mixing for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 118–126.

Chenhui Chu and Rui Wang. 2018. 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. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Zi-Yi Dou, Junjie Hu, Antonios Anastasopoulos, and Graham Neubig. 2019. Unsupervised domain adaptation for neural machine translation with domain-aware feature embeddings. In *2019 Conference on Empirical Methods in Natural Language Processing*, pages 1417–1422.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings*, pages 644–648.

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

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. pages 1317–1327. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016).*

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Yuchen Liu, Long Zhou, Yining Wang, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2018. A comparable study on model averaging, ensembling and reranking in nmt. In *Natural Language Processing and Chinese Computing*, pages 299–308, Cham. Springer International Publishing.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

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 *Proceedings of NAACL-HLT 2019: Demonstrations*.

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

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, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

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, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 464–468.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018a. Why self-attention? a targeted evaluation of neural machine translation architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272.

Gongbo Tang, Rico Sennrich, and Joakim Nivre. 2018b. An analysis of attention mechanism: The case of word sense disambiguation in neural machine translation. In *Third Conference on Machine Translation*, pages 26–35.

Taiichi Hashimoto Toshinori Sato and Manabu Okumura. 2017. Implementation of a word segmentation dictionary called mecab-ipadic-neologd and study on how to use it effectively for information retrieval (in japanese). In *Proceedings of the Twenty-three Annual Meeting of the Association for Natural Language Processing*, pages NLP2017–B6–1. The Association for Natural Language Processing.

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 I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Rui Wang, Andrew Finch, Masao Utiyama, and Eiichiro Sumita. 2017. Sentence embedding for neural machine translation domain adaptation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 560–566, Vancouver, Canada. Association for Computational Linguistics.

Yong Wang, Longyue Wang, Shuming Shi, Victor Li, and Zhaopeng Tu. 2020. Go from the general to the particular: Multi-domain translation with domain transformation networks. In *AAAI 2020 : The Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Felix Wu, Angela Fan, Alexei Baevski, Yann N. Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Chenglin Yang, Lingxi Xie, Siyuan Qiao, and Alan L. Yuille. 2018. Knowledge distillation in generations: More tolerant teachers educate better students. *arXiv preprint arXiv:1805.05551*.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. page 1535–1545.

Shiqi Zhang and Deyi Xiong. 2018. Sentence weighting for neural machine translation domain adaptation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3181–3190, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Long Zhou, Wenpeng Hu, Jiajun Zhang, and Chengqing Zong. 2017. Neural system combination for machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 378–384.