Description and Findings of OPPO's Machine Translation Systems for CCMT 2020

Tingxun Shi, Qian Zhang, Xiaoxue Wang, Xiaopu Li,
Zhengshan Xue, and Jie Hao

Manifold Lab, OPPO Research Institute, Beijing, China
{shitingxun, zhangqian666, wangxiaoxue, lixiaopu,
xuezhengshan, haojie}@oppo.com

Abstract. This paper demonstrates our machine translation systems for the CCMT 2020, which is composed of four parts. The last three parts report our results in the contest, each respectively focuses on English-Chinese bi-direction translation, Japanese-Chinese-English multi-lingual translation (patent domain), and Chinese minority languages to Mandarin Chinese translation. In each part, we will demonstrate our work on data pre-processing, model training as well as the application of general techniques, such as back-translation, ensemble and reranking. Besides, during our experiments, we surprisingly found that simply applying different Chinese word segmentation tools on low-resource corpora could bring obvious benefit across different tasks, and we will separate an independent section to discuss this finding. Among the 7 directions we participated in, we ranked the first in 6 tasks [1] and the second for the rest.

Keywords: back-translation, ensemble, reranking, multi-task learning

1 Introduction

Machine translation has always been a popular research field in the Natural Language Processing (NLP) area. In recent years, Transformer[1]-based system has become the main-stream architecture for the neural machine translation tasks, brought the field to a new stage. Since the results generated by the model are more promised, and the system is generally end-to-end, no longer as complex as the systems in the statistical machine translation era, some new research problems have emerged, such as low-resource translation, multi-lingual translation, and so on. This report describes our (OPPO's) machine translation system designed for the 16th. China Conference on Machine Translation (CCMT 2020), including all the models we trained for nearly all tasks. These tasks could be further divided into three categories, which are:

– English $\leftrightarrow$ Chinese bi-directional translation. This task provides a great amount of parallel corpus which can be used to train a good enough Trans-

[1] For the corpus filtering task, we ranked first in the 500 million words sub-task

former model. We combined rule-based and model-based preprocessing methods to clean the corpus and further experimented with some other well-known techniques, such as back-translation, domain adaptation, knowledge distillation, and reranking, and the results show that they can all generally improve the translation quality more or less. The models trained in this task are also utilized in the parallel corpus filtering task to score the sentence pairs.
- Japanese → English translation in the patent domain. This is the second year we participate in this task and different from the one we proposed in the last year [2], in this paper we will show a new solution, which is based on multi-lingual translation.
- China's Minority Languages (including Uighur, Tibetan and Mongolian (in traditional form). Written as "minority languages" below for short) to Mandarin Chinese (written as "Mandarin" below for short). All of these three tasks are resource limited, but experiments show that both the model architecture and the extra boosting techniques applied in the English ↔ Chinese section are also applicable in the low-resourced tasks. Furthermore, we surprisingly found a simple extra preprocessing to the corpus can bring a big gain for the model. We will give a brief introduction to this preprocessing method in the corresponding section, and consider to publish an individual paper to explore its application scope.

As this report introduces multiple different systems together, and most of them share a similar data processing way, model architecture and improving techniques, to avoid duplicated words, we will demonstrate the common, general skills firstly, i.e. in the English ↔ Chinese translation system in Section Three (English → Chinese corpus filtering task is also described here). Section Four shows the Japanese → English patent translation system and in Section Five this paper introduces our system for the China's minority languages to Mandarin translation task. The Final Section will summary this report and list our further work. What's more, we will make a space for our finding during the contest, show how did we combine different word segmentation results for Chinese in low-resource translation tasks to improve the system.

2 Applying Multiple Word Segmentation Tools

As written Mandarin does not have explicit word boundaries, research on the segmentation methods of Mandarin has been always an active field in Chinese NLP [21] [22]. This also leads to the development of some well-known Chinese segmentation tools such as *jieba*, *pkuseg*, and so on. Generally, people use a single segmentation tool in their experiments, and besides this mainstream way, other works like [12] argue that translating from pure character-based data can also reach a SOTA result.

Different from the current practices, inspired by the concept of multiple tasks transfer learning, in this paper we propose a new way to handle how the Chinese

words should be segmented. For a given sentence pair, we segmented the Chinese side by two different tools, *jieba* [2] and *pkuseg* [13], then combined the segmented results together with another result that is from simply splitting the Chinese sentence into characters. Since after such a process one sentence pair becomes three, we added a tag in front of both the source side and the target side, indicating for the current pair which segmentation tool is applied. In this way, the size of parallel corpus is augmented to three times bigger (for Uighur → Mandarin task, we additionally segmented the Mandarin corpus using *scws* [3], so the corpus is four times bigger).

Furthermore, as Mandarin doesn't have explicit word boundaries, we decided to remove the BPE suffices "@@" for all subwords. The reason is in some cases, the subword generated by BPE tools actually share the same meaning from the corresponding independent word, the only literal difference is the former one has an extra BPE suffix. For example, suppose a segmentation tool sees "国际贸易" (international trade, "国际" means "international" and "贸易" means "trade") as a whole word, and this word is divided into "国际 @@ 贸易" by BPE tools. In this case, the two different tokens "国际 @@" and "国际" actually have the exactly same meaning, using different tokens to distinguish them is unnecessary, and the extra introduced token enlarges the vocabulary, makes the network bigger, thus is easier to be overfit in the low-resource tasks.

After having removed the BPE suffices, we iterated all subwords again: for a given subword which has been removed the suffix, if there isn't a same full word existing in the vocabulary, then we shatter it again into characters. The intuition behind such a decision is we think in this way model can learn more information from the character-level corpus.

Table 1 shows the effect after having applied multiple word segmentation tools on Tibetan → Mandarin task. The different techniques we listed above are all introduced step by step, thus this table can be seen as the result of ablation analysis. From the table we can see all the introduced techniques helps to boost the system, so the root cause that brings the improvement would not be simple augmenting the dataset. A further analysis would be carried on in our future work.

As a sentence can be translated into different results according to the label in the very beginning, we can extract different results by the segmentation tags, compare there BLEU scores on both the validation set and the online test platform. For Tibetan task, we found using a statistical language model (*kenlm* model) to evaluate the candidates, picking out the one has the highest score, can achieve more gains (For Uighur and Mongolian this method fails. Nevertheless we submit our final results according to the reranking system, so this does not matter much)

We also tried to apply this method on the English → Chinese task, which has abundant training corpus, but did not affect the system (neither improved

[2] https://github.com/fxsjy/jieba
[3] http://www.xunsearch.com/scws/

Method	Validation set BLEU	Online test BLEU
Baseline model (Character-based Mandarin)	44.2	54.74
+ *pkuseg* segmented Mandarin	45.4	(not tested)
+ Multiple segmentation, without segmentation tag	45.7	(not tested)
+ segmentation tag & keeping BPE symbol	46.1	(not tested)
+ removing BPE symbol	46.2	55.90
+ selected by *kenlm*	**46.7**	**56.69**

Table 1: Improvement achieved by using multiple segmentation tools on Tibetan → Mandarin tasks. Here the online test data is actually the CCMT 2019 test dataset. Scores reported on the validation set are calculated by SacreBLEU (character-level BLEU4), on the online test are calculated by the official evaluation suite (character-level BLEU5-SBP)

nor harmed the system). This results shows that our proposed method is more applicable in the low-resource scenarios.

3 English ↔ Chinese Machine Translation Task

In the neural machine translation era, models become much bigger, contains more parameters, therefore generally requires more data for training. The CCMT 2020 English ↔ Chinese task provides the biggest parallel dataset across all translation tasks held in CCMT 2020, contains roughly 28 million parallel sentence pairs and another 20 million official released forward-translated data [4], such a big data amount gave us confidence to train an applicable model and experiment with some other techniques. What's more, as models generally need high quality data to generate more promising results, we also designed a data preprocessing and filtering pipeline to clean the data. This pipeline was also reused in the other tasks that will be presented later.

3.1 Data Preprocessing

Our data preprocessing procedure can be divided into two parts: In the **preprocessing** part, sentences are normalized, generally including symbol normalization, tokenization, word segmentation (for the languages that don't have explicit words boundaries, such as Chinese), and true casing (for the languages of which the letters have different cases). In this part, sentences are only converted but not dropped. The concrete steps are:

– Simplifying Chinese characters. Traditional Chinese characters are converted into their corresponding simplified forms.

[4] Including datasets released by WMT 2020, which are allowed in CCMT 2020

- Punctuation normalization. E.g. all different hyphens are converted into the standard one. For Chinese, this step includes an extra function to convert all full-width symbols (not only punctuations, but also numbers and Latin letters) into half-width, except commonly used punctuations such as full stops, commas, question marks and exclamation marks.
- Word segmentation for Chinese. We used *pkuseg* [13] as the segmentation tool.
- Tokenization. We used *Moses* [5] as the tokenization tool.
- True casing for English. The true casing tool is also from the Moses suites

In the **filtering** part, sentence pairs that have low qualities are removed. We apply two kinds of methods to filter the corpus: For the **heuristic** methods, we set up some rules and thresholds, including

- Remove the sentence pairs that contain too many non-sense symbols.
- Remove the sentence pairs that contain too long sentences (have more than 160 words).
- Remove the sentence pairs that the count difference between numbers in Chinese side and numbers in English side is greater than or equal to 3.
- Remove the sentence pairs that the count difference between punctuations in Chinese side and punctuations in English side is greater than or equal to 5.
- Sentence pairs that have abnormal length ratio, here "length" is the count of words of a sentence. We set the upper bound of words count ratio between English and Chinese to 2.2 and the corresponding lower bound is 0.7.
- Deduplication.

The rest corpus is then filtered by **alignment** information. We used *fast_align* [6] [17] to calculate the alignment information between Chinese corpus and English. Both sentence-level and word-level alignment scores are referred to. For sentence-level information, we calculated scores from English to Chinese (noted as `enzh` below for short) and scores in the reversed direction (noted as `zhen` below for short), then averaged these two scores. For word-level information, we first averaged the two scores by the word counts of each other. Our threshold for the sentence-level alignment information is -16, and for the word-level is -2.5.

After the steps we listed above, about 17 million parallel sentence pairs and 14 million official forward-translated pairs were left. We also cleaned some official provided English & Chinese monolingual corpus for back-translation and forward-translation later, data sources and corresponding dataset size after cleaning are:

- Chinese: 10.55 million, including 7.5 million LDC data and 3 million Newscrawl data.

[5] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl

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

– English: 57 million, including 20 million Newscrawl data, 20 million LDC data and 17 million News-discuss data.

We strictly followed the requirements, built constrained systems for all the shared tasks in the CCMT 2020.

3.2 Model Training

Cleaned corpus is then used to divide words into subwords. We merged the English side and the Chinese side of the parallel corpus to train BPE separations, the count of BPE merge operations is 32K, then we built vocabulary lists for two languages independently. We applied 8 heads Transformer-Big [1] architecture to train our models, using *fairseq* [5]. For `zhen` task, we tried different hyperparameters to train several models for getting ensemble model: learning rates ranged from 0.0003 to 0.0008, warmup steps fixed at 16,000, dropout ranged from 0.2 to 0.3. For `enzh` task, the hyperparameters are all fixed (but tried different random seeds): learning rate was 0.0003, warmup steps was 15,000, feedforward network dimension was 15,000. In all the CCMT 2020 tasks we used Adam optimizer [4] to optimize the models.

During training we tried the following techniques:

– Back-translation and forward-translation. For back-translation [6] we first trained models on parallel corpus, then used these models to translate monolingual corpus of the target language, and combined the synthetic pseudo parallel corpus with the original one. Although [8] indicates that adding some noises by using sampling-based decoding can improve the results, in this task we found argmax-based beam search still performs the best. Later we found adding *forward-translation*, i.e., using models to translate monolingual corpus of the source language can also boost system's performance. We borrowed ideas from [9], used ensemble model to again back & forward translate both monolingual corpus and parallel corpus, and as [7] we did such processes for three rounds.
– Domain adaptation. We found some sentence pairs in the parallel dataset are somehow away from the test dataset: test dataset is in the news domain, however the parallel dataset provides some examples from UN conferences. Besides, style of synthetic data is also different from that of the parallel one (i.e. the "translationese issue"). To address such a problem we introduced a two-phases fine-tuning method: After having trained the model on a big, natural and synthetic dataset mixture, we first fine-tuned the model on the official provided parallel corpus only, and then fine-tuned the model again only using a tiny dataset, `newstest2017`, since the newstest dataset always has higher quality and fits the domain well.
– Ensemble. As presented we always train several models for the same task, using different hyperparameters and/or random seeds, to get an ensemble model. Experiments show that ensemble model in most cases can improve the result.

– Reranking. We generated several best candidates (generally 10) from the ensemble model, and scored them by several small models. The scorers include forward-translation models (e.g. the models used to compose the ensemble model), backward-translation models (e.g. the models used to generate back-translation results) and language models. For each kind of model, we also trained its right-to-left counterpart (i.e. reverse both the source sentences and target sentences then train a model) to enrich the choice of score models. We applied K-batched MIRA [11] to rerank the candidates.

The two tables below show our results achieved in both `zhen` and `enzh` tasks, with the techniques listed above.

System	BLEU	Absolute improvement	Relative improvement
Baseline (trained by parallel corpus only)	28.8	-	-
+ back-translation	29.8	+1.0	+1.0
+ forward-translation	34.5	+5.7	+4.7
+ fine-tuned by newstest2017	36.7	+7.9	+2.2
+ ensemble & reranking	38.3	+9.5	+1.6

Table 2: Our systems for `zhen` translation task. Scores are reported on the `newstest2019` dataset and evaluated by SacreBLEU [20] . Scorers for reranking are composed of 3 forward left-to-right (l2r) models, 3 forward right-to-left (r2l) models, 3 backward r2l models and 2 l2r Transformer language models.

System	BLEU	Absolute improvement	Relative improvement
Baseline (trained by parallel corpus only)	38.6	-	-
+ back-translation	39.1	+0.5	+0.5
+ fine-tuned by parallel corpus	40.6	+2.0	+1.5
+ fine-tuned by newstest2017	41.3	+2.7	+0.7
+ forward-translation	41.9	+3.3	+2.8
+ ensemble	42.7	+4.1	+0.8
+ reranking	43.2	+4.6	+0.5

Table 3: Our systems for `enzh` translation task. Scores are reported on the `newstest2019` dataset and evaluated by SacreBLEU. We also tried the two-phases fine-tuning on the models trained by adding forward-translation, but no gains observed (So the "relative improvement" given in the "forward-translation" row is calculated based on the "back-translation" row). Scorers for reranking are composed of 5 forward l2r models, 3 forward r2l models, 3 backward r2l models and 3 l2r Transformer language models.

3.3 Corpus Filtering Task

This year we also participated in the Chinese ↔ English corpus filtering task. This task requires us to score every sentence pair in a given parallel corpus. We used the models trained for the Chinese ↔ English translation task to score the sentence pairs, averaged the score got by the forward model and backward model, then sorted them according to the averaging score. In the contrast system, we selected out the sentence pairs that contain traditional Chinese characters, and put all of them in the end of the submission.

4 Japanese → English Translation Task (Patent Domain)

The Japanese (`ja`) → English (`en`) translation task in the patent domain is designed as a multi-lingual, zero-shot, domain-specific task. Official data doesn't contain any `jaen` parallel dataset, but is composed of two independent datasets: one is Japanese ↔ Chinese (`jazh`), the other is English ↔ Chinese (`enzh`). Based on the system we proposed in the last year [2], we made some further improvements, including introducing a step inspired by the mainstream multi-lingual translation solutions.

4.1 Data Preprocessing

We adopted exactly the same data preprocessing and filtering as we described in the `enzh` section, including the hyperparameter settings, with an extra step to convert all CJK characters in the Japanese corpus to Japanese *kanji* forms. Both of the officially provided datasets contain 3 million sentence pairs, after cleaning `jazh` corpus had 2.9 million pairs left, and `enzh` had 2.8 million. We used *pkuseg* to segment Chinese sentences, *mecab* [7] to segment Japanese sentences, and *Moses-tokenizer* to tokenize sentences. All source and target side corpora are mixed to train the BPE subword segmentation, merging operations were applied for 32K steps but we did not share the vocabulary among the source and the target.

4.2 Model Training

A direct solution from the given data is to train two systems, one is from Japanese to Chinese, the other is from Chinese to English. The key defect of this system is, all the source sentences will be translated by two models consecutively. As currently models are not able to guarantee the quality of the generated results, each step has a probability to make mistakes. What's the worse, consecutive translating may even have the risk to augment the wrong signals.

In the solution we proposed last year [2], we built a `zhja` system, and translated the Chinese sentences in `zhen` corpus into Japanese, therefore we can get a pseudo Japanese → English parallel dataset which contains 2.8 million pairs

[7] https://taku910.github.io/mecab/

of data. This year, as we saw a success of forward-translation in English ↔ Chinese task, we applied the same processing way here, built a `zhen` system, thus made another 2.9 million synthetic pairs. Combined the two synthetic datasets together we can get a corpus of which the size is 5.7 million. We first trained a Transformer-Big model on this "raw" dataset, then followed the model-based filtering process described in [3], used the same model to score all the sentence pairs and further removed 100k sentences away. We trained several different checkpoints, mainly different in the learning rate (range from 0.0003 to 0.0008). The warmup steps was fixed at 16,000. Among the checkpoints we got, the best model's BLEU on the validation set is 39.5 (evaluated by SacreBLEU [8], reported on the character-level), and the ensemble model's score is 41.0.

After having built the synthetic `jaen` corpus, we tried a multi-lingual machine translation method inspired by [16]: We combined this synthetic dataset along with the other two officially released corpora, added labels at the beginning of the sentences to indicate the concrete translation directions, then trained several models on this mixed multi-lingual dataset. This multi-lingual system improved the single model for one point, from 39.5 to 40.5. However, the ensemble model only got a 0.1 point gain. Furthermore, reranking by K-Batched MIRA also brought a 0.4 point improvement.

The overview of our system for patent domain multi-lingual `jaen` translation task is listed in table 4. We also tried some fine-tune methods but didn't see any positive results. Sometimes the score on the validation set was extremely high but by analyzing the generated results we found the model actually overfitted severely, one concrete phenomenon we observed is the fine-tuned model (after decades of epoch) always add an extra "the" before countries' names (e.g. "the China"), which obviously breaks grammar rules of English. As the corpus we used to fine-tune models are selected from validation dataset according to test dataset by fda algorithm [19], we doubt there exists some gap between the validation set and test set.

System	BLEU	Absolute improvement	Relative improvement
Baseline (no forward-translation)	37.8	-	-
+ forward-translation	39.5	+1.7	+1.7
+ multi-lingual processing	40.5	+2.7	+1.0
+ ensemble	41.1	+3.3	+0.6
+ reranking	41.5	+3.7	+0.4

Table 4: Our systems for patent domain multi-lingual `jaen` translation task. Scores are reported on the validation set and evaluated by SacreBLEU. Baseline score is from the system we designed in the last year, differs from the current system in two aspects: 1. The evaluator applied in the last year is multi-bleu, 2. The Chinese segmentation used last year is *jieba*

[8] https://github.com/mjpost/sacrebleu

5 Minority Languages → Mandarin Translation Task

Comparing with the English ↔ Chinese translation task, datasets released for the minority languages → Mandarin translation task are relatively much smaller (Details can be found in table 5). Training a good deep neural network model on such low-resource datasets brings a bigger challenge to us, therefore some extra processing steps are introduced.

Language pairs	# Sentence pairs	# Tokens in the source side	# Characters in the target side
Uighur → Mandarin	169,525	3,114,647	17,244,943
Tibetan → Mandarin	162,096	32,158,312	7,839,757
Mongolian → Mandarin	261,454	5,993,512	24,579,256

Table 5: Corpora sizes in the China's minority languages → Mandarin translation task. The statistics information is collected from the very original, raw **training** data, so all of the sentences in the source side are not tokenized. As Tibetan does not show the word boundary explicitly neither (as Mandarin), in the corresponding row we count the characters amount for Tibetan

5.1 Data Preprocessing

Small data amount is a double-edged sword: From one side it makes training a good model more difficult, but from the other side it allows us to do a more careful cleaning. As the dataset is small, model is more vulnerable, easier to be disturbed by noises, so a careful cleaning is necessary and even more important. For each language pair, we list all the characters in the raw training dataset, and according to the character list we further design ad-hoc rules to modify low-frequency, irregular characters. The rules can be roughly divided into below categories:

1. Symbol forms unification: For a given symbol/punctuation, map its all variations to the most popular one (in most cases, to the corresponding ASCII form). For example, "EM Dash" (Unicode `0x2014`) is mapped to "dash" (Unicode `0x002e`).
2. Conversion between full width symbols and half width symbols. In the source side (i.e. for all minority languages) full width symbols are changed to half width symbols, and in the target side (i.e. for all Mandarin corpus) some common half width symbols are converted to their full width counterparts (such as full stops, commas).
3. Removal of the invalid/invisible/unnecessary characters. For example, Unicode `0xe5e7` is removed.

Special process for Tibetan. In the given three different source languages, Tibetan, similar to Mandarin Chinese, doesn't have explicit word boundaries, which is different from Uighur and Mongolian. Since we didn't find an ideal Tibetan word segmentation tool released by the domestic team, we chose to train a "character"-based model [9] for Tibetan → Mandarin task. The extra process for Tibetan contains

- Remove all *initial yig mgo mdun ma*s (Unicode `0x0f04`).
- Replace all morpheme delimiters (*tseg*, Unicode `0x0f0b`) to spaces.
- Add spaces around all full stops (*tshig-grub*s, Unicode (`0x0f0d`)) and roof over brackets (both *ang khang g.yon*s and *ang khang g.yas*es, Unicode `0x0f3c` and `0x0f3d`).

After this character-based cleaning, the following preprocessing steps are similar to the ones described in section 2, which contains two stages: In the first stage some normalization methods are applied, such as space normalization, punctuation normalization, symbol unescaping, and (for Mandarin) simplifying traditional Chinese characters. The second stage involves some rule-based filtering steps, like deduplication, language identification, statistical information based filtering (including count of words/characters, source-target sentences length ratio, ratio of letters for each sentence, ratio between count of characters and count of words for each sentence), and alignment based filtering. We again used *fast_align* to get the alignment information of each training dataset [10]. Notice here that in the statistical information based filtering and alignment based filtering, for each item we didn't manually hard-code the concrete thresholds, but indicated percentiles respectively. The most frequently used percentiles are 0.1% and 99.9%, since we want to keep as many data pairs as possible, meanwhile also need to get rid of the real abnormal ones (for example, too long sentences). For Mandarin, we take an extra filtering step: use the 3,500 common used Chinese characters list and a kenlm language model [10] trained on officially provided monolingual Mandarin corpus to filter out sentences that contain too many irregular codes.

Table 6 shows the data processing results after the filtering. As [15] indicated, we also found the official validation set for Tibetan task has a low quality, so we fully discarded it and sampled 1,440 sentences from the training set as new validation set. BPE subwords are applied to all the three tasks and are all trained separately. For Uighur and Tibetan the BPE merge operations are 32K and for Mongolian it is 16K. As we showed in section 2, we applied multiple word segmentation tools on all the three tasks.

[9] More accurately, morpheme-based model

[10] For Uighur → Mandarin task we didn't filter the corpus according to alignment information, since we find sometimes a Mandarin word can be a long phrase in Uighur. e.g. "法治" (rule of law) is officially translated to *"qanun arqiliq idare qilish"*. (Uighur here is transliterated by Uighur Latin alphabet (ULY)). For Tibetan, alignment information is calculated on a character-level corpus, means not only the Tibetan data is segmented by morpheme, but also the Mandarin data is split into characters

Language pairs	# Raw sentence pairs	# Kept sentence pairs	Retention rate
Uighur → Mandarin	169,525	163,762	96.60%
Tibetan → Mandarin	162,096	147,440	90.96%
Mongolian → Mandarin	261,454	228,225	96.18%

Table 6: Corpus filtering information for Minority → Mandarin tasks

5.2 Model Training

Before discovering the multiple word segmentations method (described in section 2), the baseline models in this section (for Uighur and Tibetan tasks) were trained by *marian*. We adopted the Transformer-Big architecture described in [1]. The optimizer we used is Adam, learning rate was set to 0.0001, warmup was set to 16,000, gradient norm clipping was set to 5. We also applied label smoothing, the corresponding parameter is 0.1. When evaluating the model on the validation set, the beam search size was 24 and the normalization is 1.5. We set the early stopping validation counts to 50. We followed [15], didn't average the checkpoints, but used a smoothing averaging method with the factor set to 10^{-4}.

Since multiple segmentation method is validated to be effective, we applied it on all the three Minority language tasks, using *fairseq* to train the models. The reason we switched the framework is that we found when dataset becomes larger, models trained by *fairseq* with our frequently used configuration is slightly better than those trained by marian. The changes for our *fairseq* configuration are:

- gradient norm clipping set to 0.1
- gradient update frequency set to 8
- dropout set to 0.3 (ReLU dropout and attention dropout are kept as 0.1)
- warmup initial learning rate set to 10^{-7}
- beam search for decoding in validation is set to 5, and length penalty is set to 2

What's more, we didn't adopt the smoothing averaging checkpoints as we did when using marian. Here we did not specify what learning rate and warmup steps we applied, because they actually vary across different tasks, and even in the same task we tried different configurations to get different checkpoints to further compose the ensemble model. Generally, the most common used combination is learning rate 0.001 and warmup step 16,000, but per our experiences learning rate can range from 0.0008 to 0.002, and warmup can range from 8,000 to 32,000. Sometimes bad combination can lead to gradient explosion, but mostly if the training converges, the result could be acceptable (after averaging checkpoints, the score difference between the best one and the worst one is less than 1 BLEU.)

As discussed in the previous section, we found using synthetic data generated by back-translation can improve the system a lot. In the Minority → Mandarin tasks, the same process is also applied in all the three directions. The Mandarin

corpus is again segmented in three different ways (character, *jieba* and *pkuseg*. For Uighur again plus *scws*). For the back-translation model, we cannot guarantee the same sentence segmented in different ways can be translated into the same results, but we did not process the back-translated results, leaving the divergence and hoped this could be helpful noise for the model. However, it could be better to take some extra experiments to see how unified version of back-translated results can affect the system.

Table 7 shows our three Minority → Mandarin systems results, showing the gains brought by each technique. Besides the multiple segmentation methods, we applied roughly the similar techniques we described in the previous English ↔ Chinese section, including back-translation, domain adaptation, ensemble and reranking. It should be noted that for the Minority → Mandarin tasks, monolingual data is only available for Mandarin, so we were not able to do forward-translation using our trained systems, but we followed [9] to translate back-translated source corpus using ensemble model again (ensemble knowledge distillation, ensemble KD), to augment the dataset. For back-translated data, we added a special tag `<bt>` in front of both source side and target side. For the translated results from the original data, we added a special tag `<kd>` and for the results from back-translated data, the corresponding tag is `<btkd>`. We didn't follow [15] to fine-tune our systems on knowledge distilled data, but mixed the knowledge distilled data with the original parallel corpus and back-translated data together, and trained models from the scratch.

System	Uighur	Tibetan	Mongolian
Baseline	38.6	46.7	61.4
+ Back-translation & Ensemble KD	48.6 (+10, +10)	47.9 (+1.2, +1.2)	63.9 (+2.5, +2.5)
+ Fine-tune on original parallel corpus	49.0 (+10.4, +0.4)	50.0 (+3.3, +2.1)	66.9 (+5.5, +3.0)
+ Model ensemble	49.4 (+10.8, +0.4)	53.0 (+6.3, +3.0)	69.5 (+8.1, +2.6)
+ Reranking	49.5 (+10.9, +0.1)	53.0 (+6.3, +0.0)	73.0 (+11.6, +3.5)

Table 7: Overall for the minority languages → Mandarin systems. Every score is character-level BLEU calculated on the validation dataset by SacreBLEU (for Tibetan we used a part separated from the training data which contains 1440 examples, not the official validation set). Baseline for the Uighur task was trained without applying multiple segmentation tools. Reranking for the Mongolian system followed noisy channel reranking [18], the other two used K-batched MIRA. For the Uighur system, scorers contain 22 forward l2r models, 3 backward l2r models, 3 forward r2l models, 7 forward l2r Transformer language models, 7 backward r2l Transformer language models, 1 l2r *kenlm* language model and 1 r2l *kenlm* language model. For the Tibetan system, the count of forward l2r models in the scorers pack is 16, other options have the same amount as we used for the Uighur task.

6 Conclusion and Future Work

In this report we summarized all the systems we designed for CCMT 2020. We found applying forward-translation together with traditional back-translation can bring other gains, and verified the effect of multi-lingual model training methods in the zero-shot multi-lingual task. Fine-tuning (domain adaptation) is proved to be effective if the domain of test data mismatches the domain of training data (This is again verified in the minority languages tasks, in which the Tibetan corpus is in the government domain, the Mongolian corpus is in the daily domain. However, official provided Chinese corpus is in the news domain, so fine-tune on their each parallel corpus can improve a lot). Our systems generally achieved good results: among the 7 directions we participated in, we ranked 2nd in the Mongolian → Mandarin direction with a gap of 1.3 BLEU, and 1st in the rest.

During preparing the final systems for the competition we found applying multiple Chinese segmentation tools on the low-resource dataset can boost the models' performance, we'll research on this topic further to verify whether the same idea can be also useful for other languages which have the similar feature (i.e. no explicit word boundaries), e.g. Japanese, Vietnamese, Thai and so on, and try to find a way to extend such method to languages that have explicit word boundaries. Besides, we are also interested in how to design a more effective multi-lingual translation system in the zero-shot/few-shot scenario.

References

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems (NeurIPS 2017)* (pp. 5998-6008).
2. Xue, Z., Zhang, Q., Li, X., Dang, D., Zhang, G. & Hao, J. (2019). OPPO Machine Translation System. In *Proceedings of the 15th. China Conference on Machine Translation (CCMT 2019)* (in Chinese)
3. Zhang, Q., Li, X., Dang, D., Shi, T., Ai, D., Xue, Z., & Hao, J. (2020, July). OPPO' s Machine Translation System for the IWSLT 2020 Open Domain Translation Task. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT 2020)* (pp. 114-121).
4. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
5. Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., ... & Auli, M. (2019, June). fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)* (pp. 48-53).
6. Sennrich, R., Haddow, B., & Birch, A. (2016, August). Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2016)* (pp. 86-96).
7. Hoang, V. C. D., Koehn, P., Haffari, G., & Cohn, T. (2018, July). Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation (NMT 2018)* (pp. 18-24).

8. Edunov, S., Ott, M., Auli, M., & Grangier, D. (2018). Understanding Back-Translation at Scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)* (pp. 489-500).
9. Freitag, M., Al-Onaizan, Y., & Sankaran, B. (2017). Ensemble distillation for neural machine translation. *arXiv preprint arXiv:1702.01802*.
10. Heafield, K., Pouzyrevsky, I., Clark, J. H., & Koehn, P. (2013, August). Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013) (Volume 2: Short Papers)* (pp. 690-696).
11. Cherry, C., & Foster, G. (2012, June). Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2012)* (pp. 427-436).
12. Li, X., Meng, Y., Sun, X., Han, Q., Yuan, A., & Li, J. (2019, July). Is Word Segmentation Necessary for Deep Learning of Chinese Representations?. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)* (pp. 3242-3252).
13. Luo, R., Xu, J., Zhang, Y., Ren, X., & Sun, X. (2019). PKUSEG: A Toolkit for multi-domain Chinese word segmentation. *arXiv preprint arXiv:1906.11455*.
14. Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., ... & Martins, A. F. (2018, July). Marian: Fast Neural Machine Translation in C++. In *Proceedings of ACL 2018, System Demonstrations* (pp. 116-121).
15. Hu, B., Han, A., Zhang, Z., Huang, S., & Ju, Q. (2019, September). Tencent Minority-Mandarin Translation System. In *China Conference on Machine Translation* (pp. 93-104). Springer, Singapore.
16. Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., ... & Hughes, M. (2017). Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics (TACL)*, 5, 339-351.
17. Dyer, C., Chahuneau, V., & Smith, N. A. (2013, June). A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)* (pp. 644-648).
18. Yee, K., Dauphin, Y., & Auli, M. (2019, November). Simple and Effective Noisy Channel Modeling for Neural Machine Translation. 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 2019)* (pp. 5700-5705).
19. Biçici, E., & Yuret, D. (2011, July). Instance selection for machine translation using feature decay algorithms. In *Proceedings of the Sixth Workshop on Statistical Machine Translation (WMT 2011)* (pp. 272-283).
20. Post, M. (2018, October). A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers (WMT 2018)* (pp. 186-191).
21. Huang, C. & Zhao, H. (2007). Chinese Word Segmentation：A Decade Review. In *Journal of Chinese Information Processing*, 21(3) (pp. 8-19).
22. Zhao, H., Cai, D., Huang, C., & Kit, C. (2019). Chinese word segmentation: Another decade review (2007-2017). *arXiv preprint* arXiv:1901.06079.