Volctrans Parallel Corpus Filtering System for WMT 2020

Runxin Xu[1,3]**, Zhuo Zhi**[2,3]**, Jun Cao**[3]**, Mingxuan Wang**[3]**, Lei Li**[3]

[1] Institute of Computational Linguistic, Peking University
[2] Department of Computer Science and Engineering, University of California San Diego
[3] ByteDance AI Lab, Shanghai, China
runxinxu@gmail.com
zzhi@ucsd.edu
{caojun.sh, wangmingxuan.89, lileilab}@bytedance.com

Abstract

In this paper, we describe our submissions to the WMT20 shared task on parallel corpus filtering and alignment for low-resource conditions. The task requires the participants to align potential parallel sentence pairs out of the given document pairs, and score them so that low-quality pairs can be filtered. Our system, Volctrans, is made of two modules, i.e., a mining module and a scoring module. Based on the word alignment model, the mining module adopts an iterative mining strategy to extract latent parallel sentences. In the scoring module, an XLM-based scorer provides scores, followed by reranking mechanisms and ensemble. Our submissions outperform the baseline by 3.x/2.x and 2.x/2.x for km-en and ps-en on From Scratch/Fine-Tune conditions.

1 Introduction

With the rapid development of machine translation, especially **N**eural **M**achine **Translation** (NMT) (Vaswani et al., 2017; Ott et al., 2018; Zhu et al., 2020), parallel corpus in high quality and large quantity is in urgent demand. These parallel corpora can be used to train and build robust machine translation models. However, for some language pairs on low-resource conditions, few parallel resources are available. Since it is much easier to obtain quantities of monolingual data, it may help if we can extract parallel sentences from monolingual data through alignment and filtering.

The WMT19 shared task on parallel corpus filtering for low-resource conditions (Koehn et al., 2019) provides noisy parallel corpora in Sinhala-English and Nepali-Englis crawled from the web. Participants are asked to score sentence pairs so that low-quality sentences are filtered. In this year, the WMT20 shared task on parallel Corpus filtering and alignment for low-resource conditions is very similar, except that the language pairs become

Khmer-English and Pashto-English, and the provided raw data are documents in pair, which require sentence-level alignment. Besides, no data in similar languages are provided for this year.

The participants are required to align sentences within documents in different languages and provide a score for each sentence pair. To evaluate the quality of the extracted sentence pairs, they are subsampled to 5 million English words and used to train a neural machine translation model. Finally, the BLEU score of the machine translation system is used to reflect the quality of the sentence pairs.

In this paper, we propose the Volctrans filtering system, which consists of a **mining module** and a **scoring module**. First, the mining module extracts and aligns potential parallel sentence pairs within documents in different languages. In particular, we introduce an iterative mining strategy to boost mining performance. We keep adding newly aligned high-quality parallel sentences to train the word alignment model, which is essential for the mining module. Second, the scoring module is based on XLM (Conneau and Lample, 2019), and responsible for providing scores for each sentence pair. Several reranking mechanisms are also used in this module. We conduct experiments to tune the hyper-parameters for the best filtering performance, and four systems are ensembled to achieve the final results.

2 System Architecture

2.1 Data Introduction

In detail, as is shown in Table 1, the WMT20 shared task provides:

- Document pairs, including $391,250$ Khmer-English and $45,312$ Pashto-English document pairs;

- Sentence-aligned corpora extracted from the

above document pairs, using Hunalign (DBL, 2008) and LASER (Artetxe and Schwenk, 2019), including $4,169,574$ Khmer-English and $1,022,883$ Pashto-English sentence pairs;

- Parallel data which can be used to build filtering and alignment system, including $290,051$ Khmer-English and $123,198$ Pashto-English parallel sentences;

- Monolingual data, including approximately 1.9 billion English, 14.0 million Khmer, and 6.6 million Pashto sentences.

Table 1: Statistics of Provided Data Scale

	en-km	en-ps
Document Pairs	391K	45K
Extracted Sentence Pairs	4.2M	1.0M
Parallel Sentences	290K	123K

2.2 Mining Module

Besides the given aligned sentences, we believe that there are still more potential parallel sentences that can be mined. Thus we choose to extract our own set of sentence pairs from the provided document pairs, and design a mining module aiming to gather as many parallel sentence candidates as possible. We then elaborate on our mining procedure and mining module, shown in Figure 1, in detail.
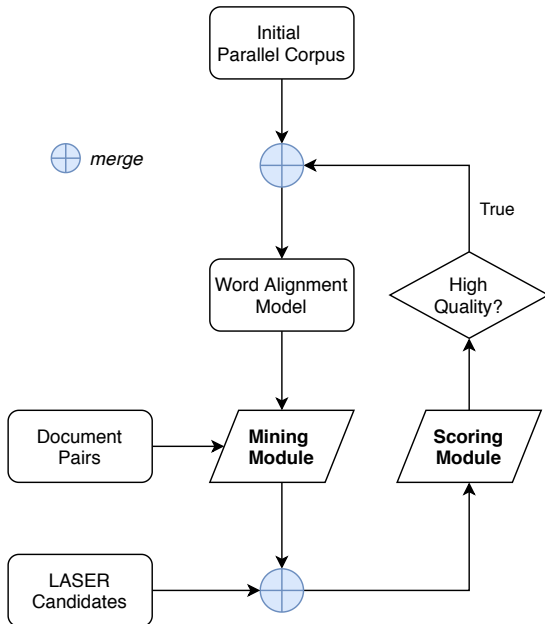

Figure 1: Mining Procedure

2.2.1 Word Alignment

We trained the word alignment model on the provided clean parallel corpus by using the *fast-align* toolkit (Dyer et al., 2013), and get the forward and reverse word translation probability tables. It's worth mentioning that both of Pashto and Khmer corpus are tokenized before word alignment model training for accuracy consideration. We separate Pashto words by moses tokenizer[1]. For Khmer, we use the character (\u200B in Unicode) as separator when it's available and otherwise use a dictionary-based tokenizer by maximizing the word sequence probability.

2.2.2 Mining Parallel Sentences

This step is operated by our mining module. With the bilingual word translation probability tables, the mining module evaluates the translation quality of bilingual sentence pairs by YiSi-2 (Lo, 2019), which involves both lexical weight and lexical similarity. The Document pairs are first segmented on each language side using Polyglot [2]. This initial segmentation is represented as:

$$e = \overline{e}_1 \overline{e}_2 \cdots \overline{e}_a = \overline{e}_1^a \qquad (1)$$

$$f = \overline{f}_1 \overline{f}_2 \cdots \overline{f}_b = \overline{f}_1^b \qquad (2)$$

where $\overline{e}_k$ ($\overline{f}_k$) is a segment of consecutive words of document e (f). Then we compute the sentence similarity (translation quality) by iteration from the initial segment ($\overline{e}_1, \overline{f}_1$). If the similarity reaches the preset threshold for ($\overline{e}_i, \overline{f}_j$), we pick the segment pair as parallel sentence candidate, and continue the computation from ($\overline{e}_{i+1}, \overline{f}_{j+1}$).

We notice that the inconsistency of segmentation in the document pairs can lead to the results: a sentence in one language contains information only part of a sentence in the other language, or two sentences (in different languages) both contain part of their information in common. These resulting sentence pairs may have low similarity scores.

In order to alleviate this problem, we also incorporate a parallel segmentation method in our mining module. We follow the basic idea proposed in (Nevado et al., 2003) where the parallel segmentation finding problem is treated as an optimization problem and a dynamic programming scheme is

---

[1] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl
[2] https://github.com/aboSamoor/polyglot

used to search for the best segmentation. We then briefly introduce our method. [3]

After obtaining the monolingual initial segmentation $(\overline{e}_1^a, \overline{f}_1^b)$, a parallel segmentation is represented as:

$$s \equiv ([\overline{e}_1^{k_1}, \overline{f}_1^{j_1}], [\overline{e}_{k_1+1}^{k_2}, \overline{f}_{j_1+1}^{j_2}],$$
$$\cdots, [\overline{e}_{k_{|s|-1}+1}^{k_{|s|}}, \overline{f}_{j_{|s|-1}+1}^{j_{|s|}}]) \quad (3)$$

where $|s|$ is the number of segments for the parallel segmentation s and $\overline{e}_{i_1}^{i_2}$ ($\overline{f}_{i_1}^{i_2}$) are consecutive segments $\overline{e}_{i_1}\overline{e}_{i_1+1}\cdots\overline{e}_{i_2}$ ($\overline{f}_{i_1}\overline{f}_{i_1+1}\cdots\overline{f}_{i_2}$). In this setting, all initial segments will be included in the parallel segmentation, and the order of the initial segments cannot be inverse. Therefore, the alignment is monotone.

Then we search for the best parallel segmentation using the objective function:

$$\max_{S \in s} C \cdot \prod_{n=1}^{|s|} P(\overline{f}_{j_{n-1}+1}^{j_n} | \overline{e}_{k_{n-1}+1}^{k_n}) \quad (4)$$

where $P(\overline{f}_{j_{n-1}+1}^{j_n} | \overline{e}_{k_{n-1}+1}^{k_n})$ is the translation quality of the pair $(\overline{f}_{j_{n-1}+1}^{j_n}, \overline{e}_{k_{n-1}+1}^{k_n})$.

Next, we use a dynamic programming algorithm to compute the best segmentation S where we have a restriction that no more than 3 initial segments can be joined.

Finally, this set of parallel segments are combined with the set of the extracted initial segment pairs through global deduplication to serve as the output of our mining module.

It is worth noting that the sequence can be very long in the process of translation quality computation because several segments can be joined together. Therefore, while computing the similarity of a segment pair, our method based on word translation probability tables can be more time-efficient than LASER, as LASER is based on sentence embeddings and can be very slow when its LSTM encoder is fed with long sequence. Thus we do not consider using LASER to compute translation quality in our mining procedure.

2.2.3 Iterative Mining Strategy

The quantity and quality of the mined data are basically dependent on the word alignment model. Besides, more high-quality parallel corpus is used, the word alignment model would be more accurate and robust. Therefore, we propose an iterative

mining strategy. For the first time, all the provided parallel data by the task are used to train the word alignment model. But we keep mining data for several times. We iteratively add new high-quality sentence pairs to the parallel corpus and train the word alignment model again to improve the word translation probability tables, thus boosting the mining cycle.

2.3 Scoring Module

The scoring module consists of three parts. First, we make use of both the parallel and monolingual data to train an XLM-based classifier to score each sentence pair. Secondly, different reranking mechanisms are used to adjust the scores. Finally, we ensemble four different models to improve the performance of our systems.

2.3.1 XLM-based Scorer

Recently, pre-trained transformer-based models play an important role in a variety of NLP tasks, such as question answering, relation extraction, etc.. Pre-trained models are often trained from scratch with self-supervised objective, and then fine-tuned to adapt to the downstream tasks. In our system, we choose the XLM (Conneau and Lample, 2019) as our main model. The reason are as follows: a) Similar to BERT (Devlin et al., 2019; Yang et al., 2019), XLM has **M**asked **L**anguage **M**odel (MLM) objective, which enables us to make the most use of the provided monolingual corpora; b) XLM also has **T**ranslation **L**anguage **M**odel (TLM) objective. Taking two parallel sentences as input, it predicts the randomly masked tokens. In this way, cross-lingual features can be captured; c) With a large amount of training corpus in different languages, XLM can provide powerful cross-lingual representation for downstream tasks, which is very suitable for parallel corpus filtering situations.

We follow the instructions [4] to prepare the training data and train the XLM model. In detail, we use Moses tokenizer to tokenize the text [5], and fastbpe [6] to learn and apply Byte-Pair Encoding. We use 50K BPE codes on the concatenation of all the training data. After applying BPE codes to the training data, we obtain a large vocabulary con-

---

[3]More details can be found in (Nevado et al., 2003)

[4]https://github.com/facebookresearch/XLM

[5]We do not use the character-/dictionary- based method introduced in Section 2.2.1 to tokenize Khmer here. Performance may be improved with that method, but we have run out of time, unfortunately.

[6]https://github.com/glample/fastBPE

taining around 100K tokens. Therefore, we only keep the top frequent $100,000$ tokens to form the vocabulary and train the XLM.

We use monolingual data in MLM objective and parallel data in TLM objective. In detail, the monolingual data we use are as follows:

- Khmer: all the 14M provided sentences.

- Pashto: all the 6.6M provided sentences.

- English: because the number of english monolingual sentences are so large, we subsample 25M sentences to keep a balance.

All the available parallel sentence pairs (29K en-km and 12K en-ps) are used in TLM objective. For each objective, we hold out 5K sentences or sentence pairs for validation and 5K for the test.

We pre-train the XLM using two different settings on 8 Tesla-V100 GPU: a) **Standard**: The embedding size is 1024, with 12 layers and 64 batch size. b) **Small**: The embedding size is 512, with 6 layers and 32 batch size. The other values of hyperparameters are all set to the default values. The two pre-trained XLM model is then fine-tuned in downstream task and further ensembled.

To score sentence pairs according to their parallelism, classification models are usually used (Xu and Koehn, 2017; Bernier-Colborne and Lo, 2019). In the training phrase, it is formulated as a binary classification problem, whether the sentence pair is semantically similar to each other or not. In the inference phrase, the probability of the positive class is considered as the score of the sentence pair. Therefore, we use the provided parallel sentence pairs as the positive instances, and construct negative instances taking advantage of such positive instances similar to Bernier-Colborne and Lo (2019). Specifically, we generate negative examples in the following ways:

- Shuffle the sentences in source language and target language respectively, and randomly align them.

- Randomly truncate the length of the source sentences or/and target sentences to 3.

- Randomly shuffle the order of the source sentences or/and target sentences.

- Simply swap the source and target sentences. Or replace the source/target sentences with target/source sentences, such that the two sentences are exactly the same.

We only add a linear or convolutional layer on top of the pre-trained XLM model and predict through a sigmoid function. The input of the model is the concatenation of a sentence pair, separated by one [SEP] token. Besides, to tackle the problem that some sentences may be too long, we simply truncate each sentence such that the maximum length of sentence is 128. The dropout rate is set to 0.5.

2.3.2 Reranking

We apply some reranking mechanisms in order to compensate for the latent bias in the XLM-based scorer, and aim to boost the quality of the whole corpus rather than each sentence pair independently.

The first reranking mechanism is based on **language identification**. For some sentences, they may include many tokens that do not belong to the corresponding language, and therefore damage the performance of the machine translation system. This phenomenon is rather common in Khmer-English corpus in particular. We utilize *pycld2* tools [7] to identify the language of the sentences. The scores of those which cannot be identified as the corresponding language are reranked by a discount of α. α is a hyperparameter.

The second reranking mechanism is based on **n-gram coverage**. Because the sentence pairs are scored independently, redundancy may exist in those high-score sentences. To enhance the diversity of the selected corpus, we first sort the sentence pairs in the descending order based on their scores. Next we maintain a n-gram pool for source sentences, and scan the source sentences from the top down. Those sentences that have no n-gram different from those in the pool will receive a discount of β, and both n and β are hyperparameters.

Note that before reranking, we always normalize the score according to their rankings, so that scores provided by different models can be unified. The score of the i-th sentence pair is:

$$score_i = 1 - \frac{rank_i}{N} \quad (5)$$

where i-th pair ranks $rank_i$ in all the sentence pairs and N denotes total number of pairs.

We also try to rerank through language models, but it does not bring improvements. Thus we do not use this reranking mechanism in our submissions.

---

[7] https://github.com/aboSamoor/pycld2

2.3.3 Ensemble

Different models may capture different features during training and inference. To make use of group wisdom and improve the final performance, we ensemble the following four models by averaging scores:

- Model 1: Standard XLM + Linear Layer. The learning rate of XLM and linear layer are $1e^{-8}$ and $1e^{-5}$ respectively.

- Model 2: Standard XLM + Linear Layer. The learning rate of XLM and linear layer are $1e^{-7}$ and $1e^{-4}$ respectively.

- Model 3: Standard XLM + Convolutional Layer. The learning rate of XLM and linear layer are $5e^{-7}$ and $5e^{-4}$ respectively.

- Model 4: Small XLM + Linear Layer. The learning rate of XLM and linear layer are $5e^{-7}$ and $5e^{-4}$ respectively.

All the models use 16 batch size per GPU.

3 Experiments

We conduct various experiments to evaluate the performance of different models, and select the most proper hyperparameters for both Khmer-English and Pashto-English. Note that **FS** and **FT** denote **From Sratch** and **Fine-Tune** respectively.

Firstly, we conduct the experiments with both the provided aligned sentence pairs (denoted as **Baseline**) and our mined data at the first iteration of the mining module. It shows that our system can outperform the baseline remarkably and the ensemble of four different models can further improve the performance. As Table 2 illustrates, Model 1-4 outperform baseline by about $1 \sim 2$ BLEU in both km-en and ps-en . Besides, the ensemble model performs the best in general.

Table 2: BLEU Scores of Difference Models

Model	km-en		ps-en	
	FS	FT	FS	FT
Baseline	7.28	10.24	9.81	11.37
Model 1	8.33	11.43	11.21	13.11
Model 2	8.96	11.38	**11.43**	13.18
Model 3	8.72	11.29	11.26	12.74
Model 4	9.01	11.27	11.36	13.09
Ensemble	**9.22**	**11.51**	11.28	**13.52**

Next, to verify the effectiveness of the iterative mining strategy in the mining module, we compare the performance of the same ensemble model with different mined data. In our paper, we iteratively mine data for three times, and combine them with the provided sentence-aligned corpus. Table 3 reveals the mining scale each time. As table 4 shows, iteration 3 works best for km-en and iteration 2 for ps-en respectively.

Table 3: The Number of Mined Sentence Pairs

Data	km-en	ps-en
Data 1	238K	200K
Data 2	330K	120K
Data 3	660K	20K

Table 4: BLEU Scores with Different Mined Data

Data	km-en		ps-en	
	FS	FT	FS	FT
+ Data 1	9.22	11.51	11.28	13.52
+ Data 1+2	9.47	11.56	**12.17**	**13.19**
+ Data 1+2+3	**9.84**	**11.62**	12.14	12.69

Finally, by introducing the reranking mechanism, we can further improve the performance, which is shown by Table 5 and 6. Note that $\alpha = 0$ or $\beta = 0$ means it does not have any discount. We select $\alpha = 0.2, n = 2, \beta = 0.2$ and $\alpha = 0, n = 1, \beta = 0.1$ for km-en and ps-en for our submissions.

Table 5: BLEU Scores with Reranking for km-en

	FS	FT
$\alpha = 0, \beta = 0$	9.84	11.62
$\alpha = 0.2, n = 2, \beta = 0.05$	10.40	12.25
$\alpha = 0.2, n = 2, \beta = 0.1$	10.38	11.87
$\alpha = 0.2, n = 2, \beta = 0.2$	**10.50**	**12.45**
$\alpha = 0.2, n = 3, \beta = 0.1$	10.09	12.09
$\alpha = 0.3, n = 2, \beta = 0.05$	10.40	12.25

4 Conclusion

In this paper, we present our submissions to the WMT20 shared task on parallel Corpus filtering and alignment for low-resource conditions. Our Volctrans system consists of two modules: a) Mining module is responsible for mining potential parallel sentence pairs out of the provided document pairs. Word alignment model is utilized and an iterative mining strategy is further taken to boost

Table 6: BLEU Scores with Reranking for ps-en

	FS	FT
$\alpha = 0, \beta = 0$	12.17	13.19
$\alpha = 0, n = 1, \beta = 0.1$	**12.28**	13.34
$\alpha = 0.2, n = 1, \beta = 0.1$	12.15	13.06
$\alpha = 0, n = 2, \beta = 0.1$	12.20	**13.38**
$\alpha = 0, n = 2, \beta = 0.2$	12.20	13.31

the mining performance. b) Scoring module aims to evaluate sentence pairs quality according to their parallelism and fluency properties, by exploiting an XLM-based scorer. We further tune the output score with different reranking mechanism, by considering language detection confidence and n-gram vocabulary coverage. Finally, four models are ensembled to improve the final performance. We also make some analysis through a variety of experiments.

References

2008. *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2008, 26 May - 1 June 2008, Marrakech, Morocco*. European Language Resources Association.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Trans. Assoc. Comput. Linguistics*, 7:597–610.

Gabriel Bernier-Colborne and Chi-kiu Lo. 2019. NRC parallel corpus filtering system for WMT 2019. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 252–260, Florence, Italy. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 7057–7067.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. 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*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Philipp Koehn, Francisco Guzmán, Vishrav Chaudhary, and Juan Pino. 2019. Findings of the WMT 2019 shared task on parallel corpus filtering for low-resource conditions. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 54–72, Florence, Italy. Association for Computational Linguistics.

Chi-kiu Lo. 2019. YiSi - a unified semantic MT quality evaluation and estimation metric for languages with different levels of available resources. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 507–513, Florence, Italy. Association for Computational Linguistics.

Francisco Nevado, Francisco Casacuberta, and Enrique Vidal. 2003. Parallel corpora segmentation using anchor words. In *Proceedings of the 7th International EAMT workshop on MT and other language technology tools, Improving MT through other language technology tools, Resource and tools for building MT at EACL 2003*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Belgium, Brussels. Association for Computational Linguistics.

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 *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Hainan Xu and Philipp Koehn. 2017. Zipporah: a fast and scalable data cleaning system for noisy web-crawled parallel corpora. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2945–2950, Copenhagen, Denmark. Association for Computational Linguistics.

Jiacheng Yang, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Yong Yu, Weinan Zhang, and Lei Li. 2019. Towards making the most of bert in neural machine translation. *arXiv preprint arXiv:1908.05672*.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating BERT into neural machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.