

DiDi’s Machine Translation System for WMT2020

**Tanfang Chen, Weiwei Wang, Wenyang Wei,
Xing Shi, Xiangang Li, Jieping Ye, Kevin Knight**

AI Labs, DidiChuxing

{chentanfang, wangweiweiwill, weiwenyang,
xingshi, lixiangang, yejieping, kevinknight}@didiglobal.com

Abstract

This paper describes DiDi AI Labs’ submission to the WMT2020 news translation shared task. We participate in the translation direction of Chinese→English. In this direction, we use the Transformer as our baseline model, and integrate several techniques for model enhancement, including data filtering, data selection, back-translation, fine-tuning, model ensembling, and re-ranking. As a result, our submission achieves a BLEU score of 36.6 in Chinese→English.

1 Introduction

We participate in the WMT2020 news translation shared tasks in Chinese → English direction. For this translation direction, we train several variants of Transformer (Vaswani et al., 2017) models on the provided parallel data enlarged with synthetic data from monolingual data. We experiment with several techniques proposed in the past translation tasks and adopt effective ones as components of our system.

Our data preparation pipeline consists of data filtering, data augmentation, and data selection. For data filtering, we filter sentence pairs based on language model scoring, alignment model scoring, etc. For data augmentation, we experiment with iterative back-translation (Sennrich et al., 2016; Edunov et al., 2018) methods and iterative knowledge distillation (Freitag et al., 2017) methods. We leverage source-side monolingual data by applying iterative knowledge distillation, and target-side monolingual data by back-translation methods, including greedy search, beam search, and noised beam search. For data selection, we select an in-domain corpus with N-grams language models and binary classifiers. A tri-gram token-level language model and a bi-gram character-level language model are introduced for English and Chinese respectively. Out-of-domain sentences which

have similar scores as in-domain sentences are chosen. We also treat data selection as a text classification problem, and use BERT (Devlin et al., 2019) as the basic classifier. In this way, we collect a corpus of high-quality in-domain training data, which improves translation performance significantly.

To enhance a single model, we use several variants of Transformer, including Transformer with relative position attention (Shaw et al., 2018), Transformer with larger feedforward inner (FFN) size (8, 192 or 15,000), and Transformer with reversed source. We then ensemble these models with adequate model diversity and data diversity to further improve the performance.

Domain conflicts influence the translation performance significantly. For example, there exist differences between written English and spoken English. Usually, a model cannot do the best in all domains due to the conflicts. In this work, we propose to obtain domain information with unsupervised clustering and exploit this information for translation. Specifically, we partition the training data, dev data, and test data into different clusters, and translate each cluster part of the test set with the model fine-tuned on the corresponding training set. Exploiting domain information helps improve the translation significantly. Details will be discussed in Section 3.

This paper is structured as follows: Section 2 describes variants of Transformer we used in the competition. In Section 3, we introduce several techniques for model enhancement, including data filtering, back-translation, fine-tuning, model ensembling. Section 4 presents experimental settings, results and analysis. Finally, in Section 5 we draw a brief conclusion of our work in the WMT2020.

2 Model

2.1 Transformer

The Transformer adopts a sequence-to-sequence structure, using stacked encoder and decoder layers of self-attention. Encoder layers consist of a self-attention layer followed by a feed-forward layer. Decoder layers consist of a masked self-attention layer, an encoder-decoder attention layer, and a feed-forward layer to incorporate source information and generate texts. The residual connections (He et al., 2016) and layer normalization (Ba et al., 2016) are introduced in the encoder and decoder layers for better convergence. In contrast to recurrent neural networks, the Transformer implicitly leverages relative and absolute position information in its structure. The Transformer introduces position encoding based on sinusoids in its inputs to incorporate position information.

In the competition we use Transformer Big as the baseline model, in which both the encoder and decoder have 6 layers, the number of heads is 16, the hidden size is 1,024, and the feedforward inner (FFN) size is 4,096.

2.2 Transformer with Relative Position Attention

The original Transformer leverages position information by taking absolute positional embeddings as inputs and does not explicitly capture the information in its structure. Thus the original Transformer cannot leverage position information efficiently. Here we used relative positional embeddings in the self-attention mechanism proposed in Shaw et al. (2018) for the encoder layers and decoder layers. We do an ablation study and find that the model with relative positional embedding has faster convergence and better performance than Transformer Big. We adopt Transformer with relative position attention as a basic architecture in the final ensemble model.

2.3 Transformer with Larger FFN Size

Since increasing the model size can help improve the performance on the NMT tasks, we experiment with Transformer with a larger embedding dimension, FFN size, number of heads, and number of layers. We find that using a larger FFN size (8, 192 or 15,000) gives a reasonable improvement in the performance while maintaining a manageable network size. We adopt a Transformer with FFN size of 8,192 and a Transformer with FFN size

of 15,000 as basic models in the final ensemble model, which has a larger inner dimension of feed-forward network than Transformer Big. Since Transformer with a larger FFN size is more likely to overfit, we set the dropout rate from 0.1 to 0.3 and use a label smoothing rate of 0.2.

2.4 Transformer with Reversed Source

We reverse the source sentences of the bilingual corpus and train a Transformer with source reversed. In this way, the model can learn a different meaning of the positional embeddings, which helps capture the source sentences from a different perspective. Viewing source in a reversed order provides another kind of model diversity and data diversity and presents positive effects in the final model ensemble.

3 System Overview

3.1 Data Filtering

Previous works (Sun et al., 2019; Xia et al., 2019; Guo et al., 2019) show that the translation performance improves as the quality of parallel corpus improves. We filter the training bilingual corpus with the following schemes:

- Normalize punctuation with Moses scripts
- Filter out the sentences longer than 120 words or sentences including a single word more than 40 characters.
- Filter out the sentences which contain HTML tags or duplicated translations.
- Filter out the sentences whose languages detected by fastText¹ (Joulin et al., 2017) are not identical to the translation direction.
- Filter out the sentences whose alignment scores obtained by fast-align² (Dyer et al., 2013) are low.
- Filter out the sentences whose n-gram scores from language models are low.
- Filter out the sentences whose length ratio between the source and target are not in range of 1 : 3 and 3 : 1

¹<https://github.com/facebookresearch/fastText>

²https://github.com/clab/fast_align

In this paper, we also filter out noisy sentence pairs with the translation acceptability filter proposed in (Zhang et al., 2020). Specifically, we feed the sentence pair (s, t) into multilingual BERT, which accepts two-sentence input due to its next-sentence prediction objective. Instead of using the [CLS] token representation, we use a Convolutional Neural Network (CNN) layer that takes the BERT output and generates the final representation of the pair. Our experiments show that using CNN layer pooling achieves marginal gains over [CLS] pooling. We use the softmax probability as the degree of parallelism and filter the sentences. The translation quality of the model boosts with the data filtering strategies.

3.2 Large-scale Back-Translation

The provided monolingual data contains a certain amount of noise, in which noise may affect the translation quality implicitly. Therefore, we adopt the data filtering schemes described in Section 3.1.

Previous work (Edunov et al., 2018) shows that leveraging the back-translation mechanism on the large-scale monolingual corpus can help improve the translation quality. Edunov et al. (2018) investigates several methods to generate synthetic source sentences, including greedy search, beam search, sampling top-K outputs, adding noise to beam search output, and adding noise to input sentences.

- Both greedy search and beam search are approximate algorithms to identify the maximum a-posteriori (MAP) output, i.e. the sentence candidate with the largest estimated probability given an input. This leads to less rich translations and is particularly problematic for text generation tasks such as back-translation.
- Sampling top-K method selects the k most likely tokens from the output distribution, renormalizes, and samples from this restricted set. This method is a trade-off between MAP and unrestricted sampling.
- Adding noise to input sentences or beam search outputs can help improve the quality and robustness of the translation.

We experiment with the above methods and observe that language pairs with abundant parallel

corpus like Chinese → English obtain obvious improvement with beam search and adding noise. In our back-translation scheme, we add noise to input sentences, and use a beam search to produce the synthetic sentences. In particular, we delete words, replace words by a filler token and swap words according to a random permutation with the probability of 0.05.

Zhang et al. (2018) proposed an iterative joint training of the source-to-target model and target-to-source model for the better quality of synthetic data. Specifically, in each iteration, the target-to-source model is responsible for generating synthetic parallel training data for the source-to-target model using the target-side monolingual data. At the same time, the source-to-target model is employed for generating synthetic bilingual training data for the target-to-source model using the source-side monolingual data. The performance of both the target-to-source and source-to-target model can be further improved iteratively. We stop the iteration when we can not achieve further improvement.

Since there are amounts of genres in both parallel and synthetic data, we adopt a language model to divide data into a coarse domain-specific corpus. We train multiple language models on different types of monolingual data (News crawl, Gigaword, etc.), and score the sentences with the language models. We select the top 600K sentences for each domain. In the final submission, we adopt an iterative joint training scheme and train models on both bilingual and synthetic data of different genres to improve translation quality.

3.3 Knowledge Distillation

Alternate knowledge distillation (Hinton et al., 2015; Freitag et al., 2017) and ensemble iteratively is adopted in the competition to further boost the performance of a single model. We simply use an ensemble model as the teacher model and boost the single student model by data augmentation. In our experiments, we use Transformer Big, Transformer with relative position, Transformer with larger FFN size, and Transformer with reversed source as basic models. For each model type, we ensemble other model types as the teacher model to boost the model performance. For example, the ensemble model of a Transformer with relative position, a Transformer with larger FFN size, and a Transformer with re-

versed source are adopted as a teacher model to improve the performance of a Transformer Big.

Considering that distillation from a poor-quality teacher model is likely to hurt the student network and thus results in an inferior performance, we selectively use distillation in the training process. In our experiments, we filter out data according to the sentence-level BLEU scores whose English translations lower than 28.

3.4 In-domain Data Selection and Fine-tuning

Domain adaptation plays an important role in improving the performance towards given test data. A practical method for domain adaptation is training on the large-scale data and then fine-tuning on the in-domain data (Luong and Manning, 2015). We select the small in-domain corpus with several approaches, including N-grams language model similarity and binary classification.

N-grams: We adopt the algorithm proposed in Duh et al. (2013); Axelrod et al. (2011), which selects sentence pairs from the large out-of-domain corpus that are similar to the in-domain data. In our work, we train a tri-grams token-level language model for English and a bi-grams character-level language model for Chinese. We use the parallel texts as the out-of-domain corpus and all available test sets in the past WMT tasks and News Commentary as the in-domain corpus. We score the sentence pairs with bilingual cross-entropy differences as follows:

$$CE(H_{I-SRC}, H_{O-SRC}) + CE(H_{I-TGT}, H_{O-TGT}) \quad (1)$$

where we denote out-of-domain corpus as O , in-domain corpus as I . H_{I-SRC} denotes language models over the source side and H_{I-TGT} denotes language models over the target side on in-domain data. H_{O-SRC} denotes language models over the source side and H_{O-TGT} denotes language models over the target side on out-of-domain data. CE denotes the cross-entropy function which evaluates the differences between distributions.

Finally, we sort all sentence pairs and select the top 600K sentences with the lowest scores to fine-tuning the parameter of the model.

Binary Classification: We also treat in-domain data selection as a text categorization problem. There are two categories: in-domain (1) and out-of-domain (0). We use the pre-trained language

model BERT as the basic classifier. For the fine-tuning data, all available newstest data and News Commentary are regarded as positive data, and randomly sampled data from the large-scale corpus are regarded as negative data. Then BERT is exploited to score the sentence pairs. We sort all sentence pairs and select the top 600K sentences with the highest scores as fine-tuning data.

All the in-domain data obtained by the above methods are adopted to fine-tuning the single model and provide about a 2 BLEU scores improvement.

3.5 Model Ensemble

Ensemble learning is a widely used technique in the real-world tasks, which provides performance improvement by taking advantages of multiple single models. In neural machine translation, a practical way of the model ensemble is to combine the full probability distribution over the target vocabulary of different models at each step during sequence prediction. We experiment with the max, avg, and log-avg strategies, and find the log-avg strategy achieves the best performance. We implement a model ensemble module in OpenNMT³ (Klein et al., 2017). In our experiments, we observe that simply enlarging the size of ensemble models does not necessarily improve translation performance. However, brute-force search of all models is prohibitively expensive and unrealistic. As the number of models increases, the decoding of the ensemble will take more time than a single model and exceed the limits of computer resource capacity. Therefore, we adopt a greedy model ensemble algorithm (Li et al., 2019) as shown in Algorithm 1.

Since model and data diversity are important factors for an ensemble system, we train diverse models with different initialization seeds, different parameters, different architectures, and different training data sets. All the models are fine-tuned to achieve superior performance.

3.6 Domain Style Translation

Translation performance differs in different topic domains. For intuitive explanation, we take native style and translation style as an example, and our topic domains are generated by using unsupervised clustering, not limited to these two styles. Native style and translation style are much differ-

³<https://github.com/OpenNMT/OpenNMT-tf>

Algorithm 1: An simple ensemble algorithm based on greedy search

Input: a model list Ω_{cand} sorted by the scores on development data.

Output: a final model list Φ_{final}

- 1 **for** all combination of 2 models that $model \in top-8\ models$ **do**
- 2 | obtain translation by ensemble decoding and evaluate with BLEU score;
- 3 **end**
- 4 Choose the best 2 model combination as the initial Φ_{final} ;
- 5 **while** there is tiny improvement as the model number increases **do**
- 6 | choose one single model from the rest of Ω_{cand} to the Φ_{final} which performs better when combined with Φ_{final} ;
- 7 **end**

ent. A single model cannot do the best in both styles. For the Chinese → English task in WMT 2017 and 2018, the source side of both dev set and test set are composed of two parts: documents created originally in Chinese (translation style) and documents created originally in English (native style). For the Chinese→English task, if the Chinese sentences are created from native Chinese corpus, then the corresponding English sentences are in translation style, so the model fine-tuned on these parallel sentences helps with translation style. Similarly, if the English sentences are created from native English corpus, the model fine-tuned on these sentences helps with native style. Previous work (Sun et al., 2019) shows exploiting translation style and native style achieves much better performance. In our work, we classify sentences into different topic categories (not limited to translation style and native style), and translate each specific part of the test set with the model fine-tuned on the corresponding training set.

Domain Label: We use pre-trained BERT models to extract [CLS] vector as the sentence embedding and obtain two clusters by K-Means clustering. We use the cluster id as the domain label.

Domain Classification: Pre-trained BERT models are fine-tuned as a text classification task, based on the source and target side with the domain label we defined above. In this way, we can

select several fine-tuning data w.r.t. different topic domains.

Decoding Stage: Since the test data is 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.

3.7 Re-ranking

We obtain n-best hypotheses with an ensemble model and then train a re-ranker using k-best MIRA (Cherry and Foster, 2012) on the validation set. K-best MIRA works with a batch tuning to learn a re-ranker for the n-best hypotheses. The features we use for re-ranking are:

- Length Features: length ratio and length difference between the source sentences and hypotheses
- NMT Features: scores from the ensemble model
- Language Model Features: scores from multiple n-gram language models

4 Experiments and Results

4.1 Experiment Setup

Our implementation of the Transformer models is based on the version 2.3.0 of OpenNMT-tf. We use Transformer Big as a basic model. Transformer Big has 6 layers in both encoder and decoder respectively, where each layer consists of a multi-head attention sublayer with 16 heads and a feed-forward sublayer with inner dimension 4096. The word embedding dimensions and the hidden state dimensions are set to 1024 for both encoder and decoder. In the training phase, the dropout rate $P_{dropout}$ is set to 0.1. Variants of Transformer described in Section 2 are adopted in the competition.

In the training phase, we use cross entropy as the loss function and apply label smoothing of 0.1. We use Adam (Kingma and Ba, 2014) as our optimizer, with parameters settings $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-8}$. The initial learning rate is set

	Transformer Big	Transformer with relative position attention	Transformer with larger FFN size	Transformer with reversed source
baseline	26.01	26.23	26.12	26.08
+ data augmentation	27.02	27.03	27.13	26.69
+ In-domain data finetuning	29.33	29.49	29.62	29.18
+ model ensemble		29.72		
+ domain style weighted		31.77		
+ reranking*		31.86		

Table 1: BLEU evaluation results on the WMT 2018 Chinese → English test set (* denotes the submitted system)

	newstest19
baseline	26.19
+ data augmentation	27.45
+ In-domain data finetuning	37.23
+ model ensemble	37.64
+ domain style weighted	38.59
+ reranking	38.99

Table 2: BLEU evaluation results on the WMT 2019 Chinese → English test set

	newstest18	newstest19
NEU (Li et al., 2019)	30.9	34.2
MSRA (Xia et al., 2019)	30.9	39.3
Baidu (Sun et al., 2019)	31.83	38
ours	31.86	38.99

Table 3: Comparison with related work on the WMT 2018 and 2019 Chinese → English test set

to 10^{-4} for training and 10^{-5} for fine-tuning. The models are trained on 4 GPUs for about 500,000 steps. Each model learns from data randomly sampled from the whole corpus, including bilingual data, synthetic data from back-translation, and synthetic data from knowledge distillation. Models used in iterative back-translation and knowledge distillation are trained for 200,000 steps. We validate the model every 1,000 steps on the development data and save the checkpoints with the best BLEU scores. After training, we average the last 10 checkpoints for every single model of the general domain.

In the fine-tuning phase, we use the averaged model obtained in the training phase as pre-train weights for domain models, and train with in-domain data selected as in Section 3.4 for 10,000 steps without early stop. After fine-tuning, we average the last 10 checkpoints for every single model of the specific domain.

For evaluation, we adopt the cased BLEU scores calculated with SacreBLEU (Post, 2018).

4.2 Pre-processing and Post-processing

In pre-processing, we conduct data filtering, tokenization, subword encoding. For Chinese sen-

tences, we use the DiDi tokenizer for tokenization. For English data, we do punctuation normalization and use Spacy⁴ tokenizer for tokenization. We filter parallel sentences as described in Section 3.1. Finally, we collect a preprocessed bilingual training data consisting of 10M parallel sentences and 20M synthetic sentences. We adopt subword encoding for Chinese → English. Specifically, we learn a BPE with 40K merge operations, in which 37.8K and 27.8K subword tokens are adopted as Chinese and English vocabularies separately.

In the post-processing phase, we conduct unknown (UNK) words replacement, de-tokenization, punctuation, and numerals normalization. UNK words are simply removed in the sentences. We use the Moses scripts to true-case and de-tokenize the English translations.

4.3 Chinese → English

We adopt methods in Section 3 for Chinese → English task. Firstly we adopt techniques of iterative back-translation and knowledge distillation for generating synthetic parallel data based on monolingual data. We combine the synthetic data and bilingual data as the training data and randomly split training data into 6 portions and do experiments to obtain 3 most effective portions. We train several models with different initialization seeds, different training datasets, and different architectures with the sampled synthetic data and bilingual data. In this way, we obtain models with diversity. After that, we fine-tune the model with different in-domain data. Next, we do the model ensemble by exploiting the translation domain style and choose the best model on development data as the final submission. Here we use WMT 2018 test set and WMT 2019 test set as our development data. Finally, we adopt several re-ranking and post-processing methods to obtain the final submission.

Table 1 shows the results on WMT 2018 test

⁴<https://github.com/explosion/spaCy>

data of Chinese → English. As shown in the table, data augmentation with iterative back-translation and knowledge distillation consistently improve the BLEU score. Fine-tuning with selected in-domain corpus plays an important role in our system, which helps achieve improvement about more than a 2 BLEU score. We observe that ensemble with log-avg strategy achieves slight improvement, which may be caused by the conflicts between different topic domains. To alleviate domain conflicts, we incorporate the domain style information, which achieves 2.15 improvement over the best single model. We also observe a relatively slight improvement with re-ranking. The reason may be that we use the training data to train both the re-ranker and the NMT models, which produces similar scores while dealing with the same sentences. Similar conclusions can be drawn from Table 2.

Table 3 shows the BLEU comparisons with related works on the WMT 2018 and WMT 2019 test sets. From the table, we observe that our system achieves the best performance on the WMT 2018 test set and the second best performance on the WMT 2019 test set. This demonstrates the effectiveness of the proposed system.

In our final submission, the model is an ensemble of 6 models, including 2 Transformer, 1 Transformer with relative position attention, 2 Transformer with larger FFN size, and 1 Transformer with reversed source. We do translation with beam size=10 and length penalty=1.4. Finally, we achieve a cased BLEU score of 36.6 in WMT 2020 Chinese → English competition.

5 Conclusion

In this paper, we present our NMT systems for WMT2020 news translation shared tasks in Chinese → English translation direction. Our final system achieves substantial improvement over baseline systems by integrating the following techniques:

1. Data filtering
2. Data augmentation, including iterative back-translation, knowledge distillation, etc.
3. Fine-tuning with in-domain data
4. Model ensemble and leverage domain topic information

As a result, our submitted system achieves a 36.6 BLEU score in the Chinese → English direction of WMT 2020 news translation shared tasks.

References

- Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Colin Cherry and George Foster. 2012. 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*, pages 427–436. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Kevin Duh, Graham Neubig, Katsuhiro Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 678–683.
- 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.
- Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500.
- Markus Freitag, Yaser Al-Onaizan, and Baskaran Sankaran. 2017. Ensemble distillation for neural machine translation. *arXiv preprint arXiv:1702.01802*.
- Xinze Guo, Chang Liu, Xiaolong Li, Yiran Wang, Guoliang Li, Feng Wang, Zhitao Xu, Liuyi Yang, Li Ma, and Changliang Li. 2019. Kingsoft’s neural machine translation system for WMT19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 196–202.

- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.
- Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, System Demonstrations*.
- Bei Li, Yiniao Li, Chen Xu, Ye Lin, Jiqiang Liu, Hui Liu, Ziyang Wang, Yuhao Zhang, Nuo Xu, Zeyang Wang, et al. 2019. The Niutrans machine translation systems for WMT19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 257–266.
- Minh-Thang Luong and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 76–79.
- Matt Post. 2018. [A call for clarity in reporting BLEU scores](#). In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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)*, pages 86–96.
- Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468.
- Meng Sun, Bojian Jiang, Hao Xiong, Zhongjun He, Hua Wu, and Haifeng Wang. 2019. Baidu neural machine translation systems for WMT19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 374–381.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Yingce Xia, Xu Tan, Fei Tian, Fei Gao, Di He, Weicong Chen, Yang Fan, Linyuan Gong, Yichong Leng, Renqian Luo, et al. 2019. Microsoft Research Asia’s systems for WMT19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 424–433.
- Boliang Zhang, Ajay Nagesh, and Kevin Knight. 2020. Parallel corpus filtering via pre-trained language models. *arXiv preprint arXiv:2005.06166*.
- Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2018. Joint training for neural machine translation models with monolingual data. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

This figure "algo_greedy_based_ensemble.png" is available in "png" format from:

<http://arxiv.org/ps/2010.08185v1>