LUL's WMT22 Automatic Post-Editing Shared Task Submission

Xiaoying Huang[1,2] and **Xingrui Lou**[1,2] and **Fan Zhang**[1] and **Mei Tu**[1]
[1]Samsung Research China – Beijing (SRC-B)
[2]State Key Laboratory of Media Convergence and Communication, Communication University of China
{huenhoying, louxingrui}@cuc.edu.cn, {zhang.fan, mei.tu}@samsung.com

Abstract

By learning the human post-edits, the automatic post-editing (APE) models are often used to modify the output of the machine translation (MT) system to make it as close as possible to human translation. We introduce the system used in our submission of WMT'22 Automatic Post-Editing (APE) English-Marathi (En-Mr) shared task. In this task, we first train the MT system of En-Mr to generate additional machine-translation sentences. Then we use the additional triple to bulid our APE model and use APE dataset to further fine-tuning. Inspired by the mixture of experts (MoE), we use GMM algorithm to roughly divide the text of APE dataset into three categories. After that, the experts are added to the APE model and different domain data are sent to different experts. Finally, we ensemble the models to get better performance. Our APE system significantly improves the translations of provided MT results by -2.848 and +3.74 on the development dataset in terms of TER and BLEU, respectively. Finally, the TER and BLEU scores are improved by -1.22 and +2.41 respectively on the blind test set.[1]

1 Introduction

Automatic Post-Editing (APE) is the task of automatically editing the translations of MT system. By using APE models, we can transfer MT system from general domain to specific domain and then reduce the workload of human post-edits. WMT has been holding APE task competitions in different languages and fields since 2015. Now the APE models are often based on transformer and improved on this basis.

WMT 2022's Automatic Post-Editing task focused on English-Marathi language pairs. The difference from the previous competition is that the target language is changed to Marathi. Besides, two new fields, medical care and tourism, are added. Participants are provided a training set with 18000 instances, a development set and a test set with 1000 instances respectively. Each dataset consists of source, machine-translation and post-edit triplets. The source sentences in English come from the healthcare, tourism, and general/news domains. The MT outputs are automatic translations to Marathi. The post-edits are human revisions of the target elements. This synthetic training data is prepared as a part of the 2022 APE shared task. The data is created by taking a parallel corpus, where the source data is translated using an MT system, and the references are considered as post-edits. Participants are also allowed to use any additional data for systems training.

Last year's research mainly focused on transfer learning and data augmentation. Sharma et al. (2021) utilizes the most advanced En-De machine translation model and further fine tune the APE dataset on this basis. We adopted the same strategy to train our baseline model with transfer learning and data augmentation. Due to the lack of a ready-made machine translation model of En-Mr as the basis of the APE model, we trained an APE model by using synthetic data and additional data. The APE model is then further fine-tuned with the APE dataset, which is data enhanced. In order to make use of the domain information in the training dataset, we use the mixture of experts structure and add adapter modules in the transformer, so that different adapters can learn the distribution of different domain information, thus improving the translation performance. The contributions of this work are as follows. (1) Data augmentation. We trained an external MT to obtain more data sets consistent with ape tasks. At the same time, we use Google translation to back translate the post-edits in the training set. The dataset is composed as follows: back translation <s> machine translation as input and post-edits as reference output. On the other hand, we take source <s> post-edits as

[1] Work performed during internship in Samsung Research China - Beijing

Proceedings of the Seventh Conference on Machine Translation (WMT), pages 689 - 693
December 7-8, 2022 ©2022 Association for Computational Linguistics

input and post-edits as reference output. (2) Mixture of adaptors. We implement the mixture of experts structure to deal with inputs from different domains, in which we use lightweight adapters as experts and introduce a classifier for expert routing. Considering the effect of directly initializing the adapter for training is not good enough, so we first set an adapter in the model and pre-train the model to obtain the adapter weight W_0. Since the training set comes from three different fields, we add two additional adapters, which will read W_0 as a parameter for initialization. We freeze all model parameters when training the model, and only fine tune the weights of the three adapters.

2 Related Work

Last year's WMT'21 APE shared task proved that both transfer learning and data augmentation were very effective. Facebook Fair's WMT19 news translation model was used in Shinhyeok's system (Oh et al., 2021). By continuously adding different levels of datasets, the model gradually understood APE tasks. For further improvement, Oh et al. (2021) used a multi-task learning strategy with dynamic weight average. By adding related subtasks, the model can learn unified representation. In addition, they also used the data set provided by ape shared task in previous years. Finally, their TER and BLEU scores were 17.28 and 71.55, respectively. Oh et al. (2021) used the most advanced machine translation model as the pre-trained model. The WikiMatrix dataset was uesd to make the model distribution tend to match the field. After that, APE samples from former years were added for fine adjustment. Finally, their model's TER and BLEU scores were 17.85 and 70.5, respectively.

Considering the experience of previous competitions, we used the existing data to train an En-Mr translation model as a data augmentation method due to the lack of advanced En-Mr translation model. Inspired by the MoE, the built-in adapter module enables the model to learn three data distributions at the same time to improve the performance of translation.

3 Dataset

3.1 Data Source

We used the WMT22 official English-Marathi APE dataset which consisted of a training and development set. We also used synthetic training data, which was prepared as a part of the 2022 APE shared task. In addition, we collected LoResMT2021 Shared Task (mac) data, CVIT PIBv1.3 (Philip et al., 2021), bible-uedin (Christodouloupoulos and Steedman, 2015) as some additional data to train our models. The LoResMT2021 Shared Task focused on machine translation of COVID-19 data for both low-resource spoken and sign languages. The LoResMT2021 dataset contains three parts: English-Irish, English-Marathi, and Taiwanese Sign language-Traditional Chinese. We only use its English-Marathi parallel corpora. CVIT PIBv1.3 is used in this work as a source for articles published in several Indian Languages to extract a multiparallel corpus. Sentences in CVIT PIBv1.3 aligned parallel corpus between 11 Indian languages, crawling and extracting from the press information bureau website. Bible-uedin is a multilingual parallel corpus created from translations of the Bible compiled by Christos Christodoulopoulos and Mark Steedman. The summary of the corpora used is provided in Table 1.

3.2 Data augmentation

As shown in Table 1, we have collected lots of parallel corpus but these corpus lack the MT part (LoResMT2021, CVIT PIBv1.3, Bible-uedin, and some synthetic training data). Following the method of generating synthetic training data, we first train a machine translation system, and then use this system to translate the source data. To generate translation similar to synthetic training data as much as possible, we did not use *src-pe* pairs but *src-mt* pairs when training MT models. We use all parallel corpus to train MT model. We are able to achieve a BLEU score of 25.3 with our MT model. Finally, we translate the sources we collected by our MT model and achieve approximately 2500000 triplets.

Yang et al. (2020) utilized data augmentation with external MT to generate the external translated sentence, which could help generate the post-editing sentence. We take a similar line of approach by leveraging external MT to generate the external translated sentence but the result is not satisfactory. So we utilize external MT to generate the external back-translations. We'd like to use back-translations to add a set of parallel corpora for the model to learn the rules of post-edits. In addition,

Source	pairs	type
APE dataset	18k	*src-mt-pe*
Synthetic training data	2.57m	*src-mt-pe*
LoResMT2021	21k	*src-pe*
CVIT PIBv1.3	117k	*src-pe*
Bible-uedin	60k	*src-pe*

Table 1: Publicly available corpuses for Indian languages.

we also use sentence X that contains a source sentence (*src*) and a post-editing sentence (*pe*) as input. We assume that the model can learn the invariance in post-editing rules by leaking some information of *pe*.

In this paper, we use D_{ape} for [*src*, <*s*>, *mt*], D_{bac} for [*src'*, <*s*>, *mt*] and D_{pe} for [*src* and <*s*>, *pe*].

4 Model

We describe our baseline model followed by the details of domain and task adaptation in this section.

4.1 Fine-tuned Transformer

Compared with previous APE tasks, this task focuses on English-Marathi language pairs. It is impossible to fine tune APE dataset on the basis of MT model. We decided to solve the APE task as NMT alike task. To adapt this idea with Transformer, we use a special token <*s*> to concatenate *src* and *mt* to generate input sentence: [*src*, <*s*>, *mt*]. We first trained the APE model with the standard Transformer (Vaswani et al., 2017) structure using synthetic training data and additional data. In order to fix the mismatch between the APE model training data and the distribution in our task, we further fine-tuned the APE model on the APE dataset.

To further solve the problem of limited data, we use the data collected in the Data section to adopt three data augmentation methods. First, we use Google translation system to create the *src'* from the provided *pe* text. We simply concatenate the *src'* with *mt* to form the new input: [*src'*, <*s*>, *mt*]. After this, the model input consists of [*src*, <*s*>, *mt*] and [*src'*, <*s*>, *mt*], which contains 36000 triplets. The second method is to add [*src*, <*s*>, *pe*] as the input on the basis of the original input and the third method is to add the first two as input at the same time. In the first way, we'd like to add a group of parallel corpora for the rules in editing after model learning. The second way is to think that by adding PE, the model can learn the rules of human post editing from SRC, MT and PE. The purpose to adopt the third method is to combine the first two methods for a better model performance.

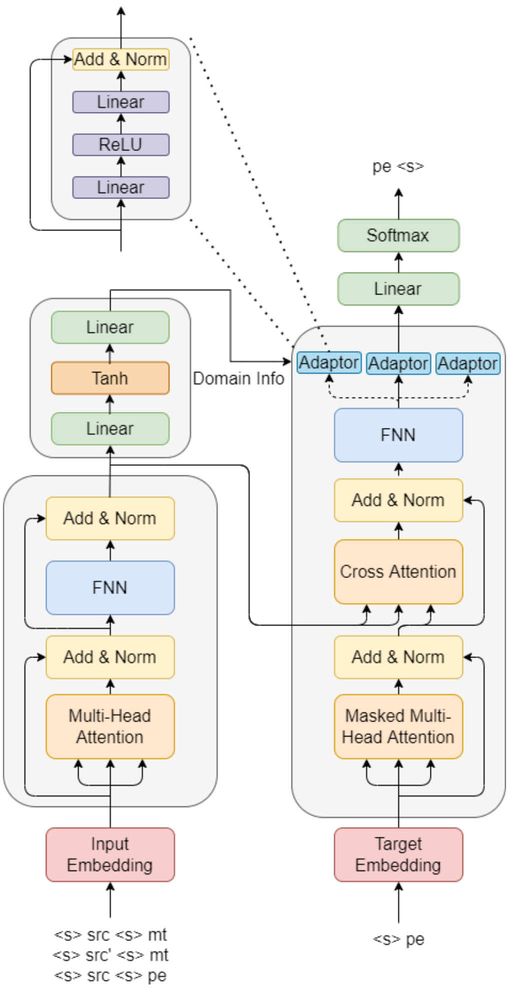

Figure 1: Adapter overall framework.

4.2 Adapter

We found that the APE dataset contains medical, tourism and general/news data. Inspired by the mixture of experts (Jacobs et al., 1991), we introduce adapters (Bapna and Firat, 2019; Pham et al., 2020) to handle different domains. We suppose that different adapters can process different domain data, so as to keep other parameters unchanged to improve the translation performance of the model for each domain, thus improving the overall TER and BLEU.

The structural diagram of the adapter is shown in Figure 1, which is similar to the FFN layer in transformer, but has a low dimensional hidden layer for nonlinear activation. In the experiment, we add the adapter layer after the FFN layer for each block in the decoder. Each adapter layer consists of three adapters. A classifier is introduced after the encoder to generate domain information (Domain Info) and it decides which adapter is activated during inference. Overall, in the inference phase, the classifier first generates domain information (Domain Info) by the encoder output, and then the corresponding adapter in each decoder is activated by the domain information. Finally, same with a general NMT model, the output is generated with an auto-regressive process.

The adapter model is trained with a pipe-line training process. First, an APE model is trained as the base model. Different from the original baseline model, an adapter is injected to each decoder layer and will be used to initialize the other two adapters in the same decoder layer. Then the classifier is trained by using a multi-classification task. Finally, with all other parameters are frozen, the parameters of the adapters are optimized by using the NMT task.

5 Experiment and Results

5.1 Experimental Settings

Both our En-Mr MT model and APE model are implemented with Fairseq framework (Ott et al., 2019). The Transformer model used for both models is Transformer-base with 6 encoders and 6 decoders, and the hidden size is 2048 for FFN layers and 512 for all other layers. The adapter used in our model is also modified to have a larger parameter size, where the hidden size of the inner layer is set to 2048.

Because we lacked the MT model, we learned the vocabulary of En and Mr by BPE. Specially, for English, we use token first and then BPE, while for Marathi, we directly conduct BPE. We believe that if token is used for Marathi before BPE, the model cannot learn the rules for punctuation after manual post-edits. The thing we should notice that the vocabulary of the En-Mr model cannot be shared which contains 31K and 31K sub-tokens for En and Mr respectively. Since the input of APE model contains En and Mr, the joint vocabulary of APE should be the total number of tokens in both languages, about 58K sub-tokens. All models were trained on NVIDIA Tesla V100. We use Adam optimizer to optimize with a fixed learning rate of 5e-4. The max tokens are set to 4096, about 64 batch sizes.

System	BLEU	TER
baseline	64.62	19.93
+Fine-tuning (D_{ape})	66.19	18.71
+Fine-tuning (D_{ape}+D_{bac})	66.44	18.56
AVG_FT (D_{ape}+D_{bac})	66.94	18.06
+Fine-tuning (D_{ape}+D_{pe})	67.24	18.09
AVG_FT (D_{ape}+D_{pe})	**67.37**	**17.91**
+Fine-tuning (D_{ape}+D_{bac}+D_{pe})	66.93	18.30
AVG_FT (D_{ape}+D_{bac}+D_{pe})	67.22	17.93

Table 2: This is the experimental result of fine-tune. AVG represents the weighted average of the model.

System	BLEU	TER
baseline	64.62	19.93
Adpt (D_{ape}+D_{bac})	66.89	18.34
AVG_Adpt (D_{ape}+D_{bac})	66.84	18.36
Adpt (D_{ape}+D_{pe})	67.55	17.90
AVG_Adpt (D_{ape}+D_{pe})	67.55	17.89
Adpt (D_{ape}+D_{bac}+D_{pe})	**67.71**	**17.85**
AVG_Adpt (D_{ape}+D_{bac}+D_{pe})	67.67	17.89

Table 3: This is the experimental result of adapter. AVG represents the weighted average of the model.

5.2 Fine-tuned Transformer

Table 2 shows the experimental results of APE fine-tune model, where the baseline result is produced by directly calculating scores between the provided *mt* and *pe*. The first experiment is performed by fine-tuning all parameters of the pre-trained Transformer on the official training set. The TER and BLEU on the 2022 dev set were 18.71 and 66.91, which were -1.2 and + 1.57 better than baseline. This demonstrates that fine-tuning the pre-trained NMT model on the limited dataset can be useful.

The experiment of training model on D_{ape}+D_{bac} and D_{ape}+D_{pe} for data augmentation shows significant improvements on the performance. However, after performing experiments with different checkpoints of APE model, we find that the best checkpoint is not the best saved checkpoint for translation, which motivates us to average model weight parameters near the best checkpoint. The avg model results show averaging the model weight parameters near the best checkpoint can help the model to be closer to the convergence point locally. The performance of the model is improved well.

5.3 Adapter

Table 3 shows the experimental results of APE adapter model. For D_{ape}+D_{bac} dataset, adding the

System	BLEU	TER
baseline	67.55	20.28
Finetune_PRIMARY	69.66	19.36
Adapter_CONTRASTIVE	69.96	19.06

Table 4: Results on test dataset.Finetune_PRIMARY ensembles AVG_FT (D_{ape}+D_{bac}) and AVG_FT (D_{ape}+D_{pe}). Adapter_CONTRASTIVE ensembles AVG_Adpt (D_{ape}+D_{bac}) and AVG_Adpt (D_{ape}+D_{pe}).

adapter does not improve the APE performance of the model, but for D_{ape}+D_{pe} dataset, adding adapter makes the model reduce TER and improve BLEU, reaching the lowest TER and the highest BLEU respectively. The experimental results show that the rough classification of data and the learning of their respective distributions are more conducive to the better APE performance of the model.

5.4 Results on Test set

Table 4 shows the official results of our proposed methods on WMT22 test dataset with a baseline scores of 20.28 and 67.55, which is higher than the development dataset with 19.93 and 64.62 in terms of TER and BLEU. Despite its high quality, our proposed methods show effectiveness on this test dataset. We find that there are some Arabic numerals and Devanagari numerals in post-edits. However, because we are not familiar with Marathi, we do not know the number modification rules. Therefore, we replace all Arabic numerals in test results with Devanagari numerals to get the final post-edits.

6 Conclusion

In this paper, we first use the data augmentation method to build the *src' <s> mt* and *src <s> pe* as two additional training datasets. We suppose that the enhanced datasets can effectively improve the performance of the APE model. The experimental results show that the data augmentation method we used is effective. At the same time, it also shows that adding *pe* can make the model automatically learn the rules of human post-edits. After that, we draw lessons from mixture of experts. We add adapters in the APE baseline model. And we let the training data be sent to different adapters through the trained classifier so that the model can further learn the post-editing rules in different translations. The experimental results confirm that our system can modify the output of MT system with high efficiency and quality. Compared with baseline, the TER and BLEU scores are improved by -1.22 and + 2.41 respectively.

References

machinetranslate.org. https://machinetranslate.org/. Accessed: 2022-05-12.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation 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)*, pages 1538–1548.

Christos Christodouloupoulos and Mark Steedman. 2015. A massively parallel corpus: the bible in 100 languages. *Language resources and evaluation*, 49(2):375–395.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Shinhyeok Oh, Sion Jang, Hu Xu, Shounan An, and Insoo Oh. 2021. Netmarble ai center's wmt21 automatic post-editing shared task submission. In *Proceedings of the Sixth Conference on Machine Translation*, pages 307–314.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Minh Quang Pham, Josep-Maria Crego, François Yvon, and Jean Senellart. 2020. A study of residual adapters for multi-domain neural machine translation. In *Conference on Machine Translation*.

Jerin Philip, Shashank Siripragada, Vinay P Namboodiri, and CV Jawahar. 2021. Revisiting low resource status of indian languages in machine translation. In *8th ACM IKDD CODS and 26th COMAD*, pages 178–187.

Abhishek Sharma, Prabhakar Gupta, and Anil Nelakanti. 2021. Adapting neural machine translation for automatic post-editing. In *Proceedings of the Sixth Conference on Machine Translation*, pages 315–319.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Hao Yang, Minghan Wang, Daimeng Wei, Hengchao Shang, Jiaxin Guo, Zongyao Li, Lizhi Lei, Ying Qin, Shimin Tao, Shiliang Sun, et al. 2020. Hw-tsc's participation at wmt 2020 automatic post editing shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 797–802.