DLUT-NLP Machine Translation Systems for WMT24 Low-Resource Indic Language Translation

Chenfei Ju[1], **Junpeng Liu**[1], **Kaiyu Huang**[2] and **Degen Huang**[1]
[1]Dalian University of Technology, Dalian, China
[2]Beijing Jiaotong University, Beijing, China
{845110184, liujunpeng_nlp}@mail.dlut.edu.cn, kyhuang@bjtu.edu.cn
huangdg@dlut.edu.cn

Abstract

This paper describes the submission systems of DLUT-NLP team for the WMT24 low-resource Indic language translation shared task. We participated in the translation task of four language pairs, including en↔as, en↔mz, en↔kha, en↔mni. We used a transformer-based neural network architecture to train the model. Our system used the following methods: First, data processing was performed, and then we used monolingual data for pre-training. Next, we used parallel data for fine-tuning to obtain a multilingual translation model, and then we used this model for back-translation. We merged the back-translated data with the official parallel data and used the upsampling method to train a multilingual translation model from scratch. In order to improve the translation ability of the model for each translation direction, we fine-tuned the model for each language pair and used model averaging to obtain the best model for each language pair. Finally, we used kNN-MT and established a datastore using the official parallel data to assist translation in the inference stage. Experimental results show that our method greatly improves the BLEU scores for translation of these four language pairs.

1 Introduction

This paper introduces our system for WMT24 low-resource Indic language translation shared task. We participated in 4 language pairs, including English↔Assamese (en↔as), English↔Mizo (en↔mz), English↔Khasi (en↔kha) and English↔Manipuri (en↔mni).

The main methods used by our system are denoising language model pre-training (Lample and Conneau, 2019; Song et al., 2019; Lewis et al., 2020), back-translation (Sennrich et al., 2016a) and kNN-MT (Khandelwal et al., 2020). Neural machine translation is the first choice for machine translation systems nowadays, but it requires a large amount of parallel data. Therefore, low-resource translation is a major challenge due to its lack of data. In this task, the organizers provided a large amount of monolingual data in addition to a small amount of parallel data. So we considered using some pre-training methods to improve the performance of the model. At the same time, back-translation is a commonly used method in the field of machine translation, which is effective in many scenarios. Therefore, we used the back translation method to obtain pseudo-parallel data to train a strong baseline model. To obtain the best model for each translation direction, we fine-tuned the baseline model for each language pair using the official parallel data. During this process, we used model averaging technology to improve the translation quality of the model. In addition to parametric methods, a large number of non-parametric methods have recently emerged to help models generate translations. We adopted the kNN-MT method and built a datastore for each translation direction to assist the model in the inference phase.

The rest of the paper is organized as follows: In Section 2 we describe our data processing methods; In Section 3 we describe the implementation process of our translation systems; In Section 4, we describe the experimental settings; In Section 5, we discuss about the results; Finally, in Section 6, the conclusion is drawn.

2 Data

For bilingual data, we only used official bilingual data. For monolingual data, in addition to the official monolingual data for Assamese, Mizo, Khasi and Manipuri (Pal et al., 2023; Pakray et al., 2024), we obtained English monolingual data from the WMT24 general task. Specifically, we used the English side of bilingual data (English↔German) in the WMT24 general task as English monolingual data.The statistics of the dataset is shown in Table 1.

Proceedings of the Ninth Conference on Machine Translation, pages 742–746
November 15-16, 2024

	as	kha	mni	mz	en
train (mono)	2.6M	0.2M	2.1M	1.9M	2.5M
train (para)	50k	24k	22k	50k	-
dev	2k	1k	1k	1.5k	-
test	2k	1k	1k	2k	-

Table 1: The number of sentences in the training, dev and test sets.

Since the quality of official data is relatively high, we did not perform additional preprocessing. For the English monolingual data, we performed some additional preprocessing steps. During preprocessing, we deleted sentences that were too long or repeated. And then we filtered out sentences in other languages by applying language identification. Finally we used an n-gram language model trained with KenLM (Heafield, 2011)[1] to calculate the perplexity of English monolingual data and removed sentences with high perplexity (>7,000). We used the Sentencepiece (Kudo and Richardson, 2018) tool to train a multilingual BPE (Sennrich et al., 2016b) model for subword segmentation. The training data includes all the parallel training data and monolingual data. The vocabulary size is set to 32,000.

3 System Overview

3.1 Pre-training

Using monolingual data for pre-training tasks is an effective solution for low-resource situations (Raffel et al., 2020). To this end, we first performed BART-style pre-training (Lewis et al., 2020) with all the available monolingual data and then fine-tuned the pretrained model with bilingual data. Following Lewis et al. (2020), we masked words with a probability of 0.15 and we randomly swapped words in the input sentences with a probability of 0.5.

After pre-training, we used all the bilingual data to fine-tune the pre-trained model. The bilingual data contains 4 language pairs in 8 translation directions.

3.2 Back-translation

To improve our translation pipeline, we explored the integration of back-translation as a potential enhancement. Back-translation involves using a trained model to translate from the target language back to the source language, effectively creating a synthetic parallel dataset. We used the approach inspired by Sennrich et al. (2016a) to generate pseudo-parallel corpus.

Specifically, we used the model fine-tuned in the pre-training phase. We used this model to translate all non-English monolingual data into English as pseudo-parallel data. Then we mixed all the pseudo-parallel data with the official bilingual data. We used this data to train a multilingual translation model from scratch. During training, we used upsampling method and the official parallel data was upsampled until it reached to a ratio of 1:1 with the synthetic data.

3.3 Language-specific Fine-tuning

Although multilingual translation models have made great progress, there is still the problem of inconsistent convergence of different language pairs in joint training (Wu et al., 2021; Huang et al., 2022). That is, different language pairs reach convergence in various training stages. We hope to get the best model for each language pair. Due to the low quality of pseudo-parallel data, we used the official bilingual data of each language pair to fine-tune the model trained using pseudo-parallel data.

During fine-tuning, we used the model averaging technology. Through model averaging, we combined the advantages of various models into a unified translation model. This process can not only improve the stability of the translation output, but also help improve the overall translation quality. We kept the three models with the lowest loss on the validation set for each language pair. We then used these three models to get the best model for each language pair.

3.4 kNN-MT

Non-parametric, k-nearest-neighbor algorithms have recently made inroads to assist generative models such as language models and machine translation decoders. Khandelwal et al. (2020) introduced k-nearest-neighbor machine translation

[1] https://github.com/kpu/kenlm

(kNN-MT): a simple non-parametric method for machine translation via nearest-neighbor retrievals was proposed and has been verified its effectiveness. According to his method, we constructed a datastore to store the translation examples to be accessed during decoding with the official parallel data. When decoding, we used the current translation context to retrieve the k-nearest-neighbors in the datastore. Let $\boldsymbol{x} = \left(x_1, \ldots, x_{|\boldsymbol{x}|}\right) \in \mathcal{V}_X^{|\boldsymbol{x}|}$ and $\boldsymbol{y} = \left(y_1, \ldots, y_{|\boldsymbol{y}|}\right) \in \mathcal{V}_Y^{|\boldsymbol{y}|}$ denote a source sentence and target sentence, respectively, where $|\cdot|$ represents the length of the sentence, and $\mathcal{V}_X$ and $\mathcal{V}_Y$ are the vocabularies of the source language and target language, respectively. Each target token y_t from the translation examples is stored in the datastore with a d-dimensional key ($\in \mathbb{R}^d$), which is the representation of the translation context $(\boldsymbol{x}, \boldsymbol{y}_{<t})$ obtained from the decoder of the pre-trained NMT model. The datastore $\mathcal{M} \subseteq \mathbb{R}^d \times \mathcal{V}_Y$ is formally defined as a set of tuples as follows:

$$\mathcal{M} = \{(f\left(\boldsymbol{x}, \boldsymbol{y}_{<t}\right), y_t) \,|(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}, 1 \leq t \leq |\,\boldsymbol{y}\,|\} \tag{1}$$

The size of the datastore for each translation direction is shown in Table 2. During decoding, kNN-MT retrieves the k-nearest-neighbor key–value pairs $\{(\boldsymbol{k}_i, v_i)\}_{i=1}^{k} \subseteq \mathbb{R}^d \times \mathcal{V}_Y$ from the datastore $\mathcal{M}$ using the query vector $f\left(\boldsymbol{x}, \boldsymbol{y}_{<t}\right)$ at timestep t. $f: \mathcal{V}_X^{|x|} \times \mathcal{V}_Y^{t-1} \rightarrow \mathbb{R}^d$ represents the intermediate representation of the final decoder layer from the source sentence and prefix target tokens. In our system, the value of k is set to 32 for all translation directions. In order to speed up the retrieval during translation, we used FAISS (Johnson et al., 2019). We then obtained the output probability for each token by interpolating the kNN-MT probability and the probability from the translation model. The formula for calculating the kNN-MT probability is:

$$\begin{aligned} & p_{k\mathrm{NN}}\left(y_t \mid \boldsymbol{x}, \boldsymbol{y}_{<t}\right) \\ & \propto \textstyle\sum_{i=1}^{k} \mathbb{1}_{y_t=v_i} \exp \frac{-\left\|\boldsymbol{k}_i - f\left(\boldsymbol{x}, \boldsymbol{y}_{<t}\right)\right\|_2^2}{\tau} \end{aligned} \tag{2}$$

The formula for calculating the output probability is as follows:

$$\begin{aligned} & P\left(y_t \mid \boldsymbol{x}, \boldsymbol{y}_{<t}\right) \\ & = \lambda p_{k\mathrm{NN}}\left(y_t \,|\, \boldsymbol{x}, \boldsymbol{y}_{<t}\right) + (1-\lambda) p_{\mathrm{NMT}}(y_t \,|\, \boldsymbol{x}, \boldsymbol{y}_{<t}). \end{aligned} \tag{3}$$

For all translation directions, we set λ = 0.3 and τ = 100 in the kNN-MT decoding.

datastore	size
en→as	1,212,711
en→kha	1,024,451
en→mni	574,142
en→mz	1,404,832
as→en	1,253,490
kha→en	878,620
mni→en	524,002
mz→en	1,263,000

Table 2: Datastore size for all translation directions.

4 Experiments

All of our translation models were implemented based on fairseq (Ott et al., 2019) and trained on 8 NVIDIA 3090 GPUs. All models use the same structure of 12 transformer layers (Vaswani et al., 2017). During training, we used the Adam (Kingma, 2014) optimizer with β_1 = 0.9, β_2 = 0.98, the learning rate scheduling strategy of inverse sqrt, the number of warmup step set to 4000, the maximum learning rate set to 0.0005 and FP16 to accelerate the training process. We trained our models till convergence with early stopping criteria with a patience of 5. The dropout ratio is set to 0.5. We used a fixed beam size of 4 and a length penalty of 0.8 when doing back-translation.

All experiments were evaluated using the sacrebleu (Post, 2018) tool to calculate BLEU (Papineni et al., 2002) scores on the official validation sets.

5 Results

As shown in Table 3, each method can bring certain improvements to the model. However, pre-training and back-translation did not bring much improvement. For example, pre-training leads to an improvement of 0.82 BLEU on average, while back-translation brings BLEU improvements of 0.41. In particular, back-translation has caused some damage to the performance of the model on some translation directions. The BLEU in en→mni direction dropped from 25.17 to 24.04. This may be caused by the low quality of pseudo-parallel data. We believe that fine-tuning the model separately using the data of each language pair is necessary for a multilingual translation model. And it achieves 1.03 BLEU improvement on average. Doing so can alleviate the problem of inconsistent convergence of different language pairs in joint training, although it does not benefit all translation directions. It can be seen that all translation directions

System	en→as	en→kha	en→mz	en→mni	as→en	kha→en	mz→en	mni→en
M2M Baseline	8.75	17.84	22.26	24.49	15.69	13.15	22.45	32.41
Pre-training	9.24	17.77	22.72	25.17	17.70	**14.05**	22.89	34.08
Back-translation	11.51	18.24	23.29	24.04	17.98	13.22	23.36	35.25
Fine-tuning	12.50	18.29	24.17	26.93	18.55	13.33	24.32	37.06
kNN-MT	**12.82**	**18.78**	**29.39**	**28.99**	**19.69**	13.82	**31.27**	**39.02**

Table 3: BLEU scores of all translation direction on validation sets

are further improved with kNN-MT (+2.33 BLEU). The four translation directions of the two language pairs en↔mni and en↔mz can even get an average improvement of 4.05 BLEU. This shows the great potential of kNN-MT in improving data utilization efficiency, inspiring more research on kNN-MT in low-resource scenarios. Finally, from the overall perspective, some translation directions do not benefit much from our methods. The translation performance of the model in these translation directions may be most limited by the size of the data. However, the results in most translation directions still achieve significant improvements over the baseline, which demonstrates the effectiveness of our approach for low-resource machine translation.

6 Conclusion

In this paper, we describe DLUT-NLP's submission to the WMT24 low-resource Indic language translation shared task. We participated in four subtasks with a total of eight translation directions. We leveraged methods ranging from pre-training, back-translation, language-specific fine-tuning and kNN-MT. Experimental results show that we achieved large improvements in all directions.

Limitations

We found that our system still has the following limitations:

- We did not perform effective filtering on the pseudo-parallel corpus, and we did not perform iterative back-translation. This may be the reason why our back-translation did not achieve the expected results.
- We believe that we have not made enough use of monolingual data. Next, we need to explore other ways to use monolingual data, such as using other pre-training tasks.
- We did not leverage any existing LLMs because we were not sure whether they were trained on languages other than English included in the task. This will also be a future exploration mission.

Acknowledgements

We sincerely thank the reviewers for their insightful comments and suggestions to improve the paper. This work was supported by the Key Research and Development Program of Yunnan Province (Grant No. 202203AA080004). This research was also partially supported by Global Tone Communication Technology (GTCOM).

References

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197.

Yichong Huang, Xiaocheng Feng, Xinwei Geng, and Bing Qin. 2022. Unifying the convergences in multilingual neural machine translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6822–6835, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Nearest neighbor machine translation. *arXiv preprint arXiv:2010.00710*.

DP Kingma. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Partha Pakray, Santanu Pal, Advaitha Vetagiri, Reddi Mohana Krishna, Arnab Kumar Maji, Sandeep Kumar Dash, Lenin Laitonjam, Lyngdoh Sarah, and Riyanka Manna. 2024. Findings of wmt 2024 shared task on low-resource indic languages translation. In *Proceedings of the Ninth Conference on Machine Translation (WMT)*.

Santanu Pal, Partha Pakray, Sahinur Rahman Laskar, Lenin Laitonjam, Vanlalmuansangi Khenglawt, Sunita Warjri, Pankaj Kundan Dadure, and Sandeep Kumar Dash. 2023. Findings of the wmt 2023 shared task on low-resource indic language translation. In *Proceedings of the Eighth Conference on Machine Translation*, pages 682–694.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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, Brussels, Belgium. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. 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, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Minghao Wu, Yitong Li, Meng Zhang, Liangyou Li, Gholamreza Haffari, and Qun Liu. 2021. Uncertainty-aware balancing for multilingual and multi-domain neural machine translation training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7291–7305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.