Maastricht University's Large-Scale Multilingual Machine Translation System for WMT 2021

Danni Liu, Jan Niehues
Department of Data Science and Knowledge Engineering, Maastricht University
{danni.liu,jan.niehues}@maastrichtuniversity.nl

Abstract

We present our development of the multilingual machine translation system for the large-scale multilingual machine translation task at WMT 2021. Starting form the provided baseline system, we investigated several techniques to improve the translation quality on the target subset of languages.

We were able to significantly improve the translation quality by adapting the system towards the target subset of languages and by generating synthetic data using the initial model. Techniques successfully applied in zero-shot multilingual machine translation (e.g. similarity regularizer) only had a minor effect on the final translation performance.

1 Introduction

This paper describes Maastricht University's participation in the large-scale multilingual machine translation task of WMT 2021. We participate in Small Track #2. In this track, the task is to build a translation system between English and 5 Southeast Asian languages. The evaluation is performed on all 30 possible translation directions between these languages. We are provided with parallel data extracted from Wikipedia and other sources for all language pairs, as well as a large-scale multilingual machine translation model pretrained on 124 languages (Goyal et al., 2021) including the languages in this task.

Starting from the provided baseline models, we investigate several directions in order to improve the performance on the 30 target language directions. As the first step, we focus on methods to adapt the model to these language directions. Specifically, we investigate different strategies to fine-tune the model on the proposed parallel training data.

Since the provided parallel data is extremely limited for several translation directions, we investigate the use of synthetic parallel data. We focus on using pivot languages in order to use well performing language direction to generate training data for worse performing language directions.

	jv	**id**	**ms**	**tl**	**ta**	**en**
su	44	243	137	440	108	904
jv		644	340	662	46	2,556
id			4,060	2,356	415	48,486
ms				1,174	297	12,023
tl					489	12,348
ta						1,864

Table 1: Number of sentences for each languages pair after preprocessing (in thousand sentences).

Finally, we investigate the usefulness of techniques to promote the similarity of representation between different languages. While these techniques were shown essential for models to perform zero-shot machine translation (Arivazhagan et al., 2019a; Pham et al., 2019; Liu et al., 2021), in our experiments the impact of these methods is only limited.

2 Data

We start by introducing the training data and preprocessing steps.

2.1 Languages

As required for the small tracks, we only use the data provided by the organizers. The covered languages are: Javanese (jv), Indonesian (id), Malay (ms), Tagalog (tl), Tamil (ta), English (en). Although Sundanese (su) is later excluded from the evaluations, we still include it in the training data because of its high relatedness to Javanese (jv), one of the lowest-resource languages in this track.

2.2 Preprocessing

After de-duplicating, we remove sentences with more than 50% punctuation marks or digits, and

Proceedings of the Sixth Conference on Machine Translation (WMT), pages 425–430
November 10–11, 2021. ©2021 Association for Computational Linguistics

sentence pairs of length ratios beyond 1:5 in character count. We also apply frequency cleaning following M2M-100 (Fan et al., 2021). An overview of the training data amount after preprocessing is shown in Table 1.

3 Techniques

Our efforts to improve upon the provided baselines can be categorized into three directions: finetuning (subsection 3.1), utilizing synthetic data (subsection 3.2), encouraging similarities between languages (subsection 3.3).

3.1 Language Adaptation

While the provided baseline systems are trained on over 100 languages, our target track focuses on 6 specific languages. Therefore, we investigate different fine-tuning methods to adapt the model to the target languages. In this framework, we initialize from the pretrained baseline model and continue training on different subsets of the given training data. First (§3.1.1), we finetune on each of the language pairs. Then (§3.1.2), we apply finetuning on all the languages jointly. Finally (§3.1.3), in an attempt to preserve model performance on all other directions, we use adapter layers dedicated to the languages of interest.

3.1.1 Language-Specific Adaptation

First, we adapt the model to each translation direction individually, resulting in 30 different translation systems. Each of the models is trained on a single language pair from the provided translation directions.

While this approach achieves good performance, the main disadvantage is that we will have 30 individual systems. Since all of the individual models are fine-tuned from the same baseline model, we hypothesize that the resulting models are relatively similar. Therefore, we investigate the possibility of checkpoint-averaging on all the models adapted to individual language pairs, as successfully done in previous evaluations of multilingual translation (Pham et al., 2017).

3.1.2 Language-Independent Adaptation

Similar to the motivation for the checkpoint-averaging described above, in order to preserve one single model, we adapt the baseline system to all the target language pairs jointly by continue training on all the provided training data.

3.1.3 Adapter Layers

The approaches above update all parameters of the pretrained baseline model during the fine-tuning stage. As a result, the models would lose performance on translation directions other than those in the training data. To avoid this catastrophic forgetting, we take inspiration from the adapters (Bapna and Firat, 2019) and insert feedfoward layers after each encoder/decoder layer. When finetuning, we only train these parameters while keeping the rest of the model frozen. At test time, the model keeps the adapter layers for the languages seen in training. When handling languages unseen in training, the model drops the adapters and falls back to the pretrained baseline.

A main difference to the multilingual adapters (Bapna and Firat, 2019) is that our adapter layers are not language-pair-specific. Instead, they are shared among all the directions. A main reason for sharing the adapters is that, when scaling to more languages, a quadratic set of adapters would be needed.

Due to resource constraints, in this work we only train one set of adapter layers for the translation directions in Small Track #2. Nevertheless, we believe this approach could remain applicable when scaling to more languages like in Large Track. This could, for instance, be achieved by multiple sets of adapter layers dedicated to different language families.

3.2 Synthetic Data

Motivated by the strong improvements from synthetic data in multilingual speech translation evaluation (Anastasopoulos et al., 2021), we investigate the creation of synthetic parallel data for all language direction with limited available parallel data. Based on the corpus statistics (Table 1), we select all languages pairs with less than n parallel sentences as low-resource directions. In the initial experiments, we choose a threshold of $n = 500$K sentences pairs. Following successful initial experiments, we increase n to 2M sentences. For all the language with $k < n$ parallel sentences, we generated $n - k$ synthetic sentences. Consequently, the final system is trained on at least n sentences for each language pair.

While monolingual data was provided, as the initial translation system performed poorly on low-resource directions, we chose not to directly generate synthetic data from the monolingual data. Instead, we create synthetic data based on parallel data between the source and a pivot language. Synthetic target-side data is created by translating out of the pivot language. When selecting the pivot language, we choose the source-pivot pair with the highest BLEU scores.[1]

In this above-described scenario, as the source sentences are human-generated and the synthetic target sentences are automatically generated, we hypothesize that the mistakes are concentrated on the target side. This is normally addressed by using back-translation and generating the translation in the inverse directions. Since we aim to use the generated languages for both direction (source to target and target to source) for the sake of efficiency, we generate half of the sentences in the one direction and the other half in the inverse direction.

3.3 Encouraging Similar Representations

As shown in the statistics in Table 1, our training data is highly unbalanced. Extreme low-resource pairs such as ta↔jv could be considered as few-shot directions. We therefore explore several techniques shown useful for zero-shot conditions, and investigate their usefulness in this current scenario.

3.3.1 Similarity Regularization

First shown in (Arivazhagan et al., 2019a; Pham et al., 2019), an auxiliary loss promoting similarities between source and target languages facilitate zero-shot translation. Given source sentence X and target sentence Y, besides the translation loss, we minimize the following auxiliary loss:

$$\mathcal{L}_{similarity} = \lambda \cdot \texttt{dist}(X, Y), \qquad (1)$$

where $\texttt{dist}(\cdot)$ is the Euclidean distance between two meanpooled sentence embeddings, and λ is the weight for this auxiliary loss.

3.3.2 Residual Removal

The residual removal approach was shown helpful to zero-shot translation by reducing the positional information from source sentences (Liu et al., 2021). Specifically, the residual connections of a middle encoder layer is removed to relax the strong positional correspondence between input tokens and encoder outputs.

4 Experimental Setup

4.1 Training Details

The provided M2M-100 models (Fan et al., 2021; Goyal et al., 2021) cover over 100 languages and have a vocabulary size of 256K. To accelerate training and reduce GPU memory usage, we trim away the word embedding of those tokens that do not occur in our training data. After vocabulary trimming, our vocabulary size is 165K. An exception where we do not trim the vocabulary is when training with adapters, since the goal is to preserve performance on all languages.

To counteract the data imbalance among the language pairs, following previous works (Arivazhagan et al., 2019b; Tang et al., 2020), we use a sampling temperature of 5.0 which upsamples the low-resource pairs.

For the model with similarity regularizer, we use weight of 0.1 on the auxiliary loss. For the model with residual removal, the residual layer is skipped after the third encoder layer. For the adapters, we use a bottleneck dimension of 256.

4.2 Decoding and Evaluation

When decoding we use a beam size of 5, and limit the maximum output length to 1.3 * source length + 5. We report translation performance on the FloRes-101 (Goyal et al., 2021) devtest set. The BLEU reported scores are the spBLEU (Goyal et al., 2021) variant based on sentencepiece (Kudo and Richardson, 2018) tokenization. The systems are also submitted to Dynabench (Kiela et al., 2021)[2] for evaluation on a blind test set.

5 Results

In this section, we report the results of the three directions we explored: finetuning on the focus languages (subsection 5.1), utilizing synthetic data (subsection 5.2), and encouraging similarities between languages (subsection 5.3).

5.1 Language Adaptation

The results of adapting the model towards different language pairs are shown in Table 2.

5.1.1 Language-Dependent and -Independent Adaptation

We start with the small baseline model for faster experiment iterations, with results summarized in

[1] In the later experiments, the best-performing source-pivot direction is always source-English.

[2] `https://dynabench.org/flores`

System	BLEU
baseline small	11.5
+ lang. dep. fine-tune	20.6
+ averaging	5.7
+ lang. indep. fine-tune	19.6
baseline big	15.4
+ lang. indep. fine-tune	27.4
+ shared adapter layers (rest frozen)	23.0

Table 2: Results of different fine-tuning approaches from the baseline models. Language-dependent fine-tuning achieves the strongest performance, but creates individual models for each translation direction. Averaging the models from the individual directions performs poorly. Fine-tuning on all directions falls slightly behind language-specific fine-tuning but preserves one single model.

the upper section of Table 2. When adapting to each language pair individually (lang. dep. fine-tune), we see large gains with average BLEU score increasing from 11.5 to 20.6. In contrast to previous work (Pham et al., 2017), we are not able to preserve this gain by averaging all the individual models into one single models. Instead, averaging the models results in a low average BLEU score of 5.7. This suggests the adapted individual models are relatively dissimilar and cannot be simply averaged.

Nevertheless, by fine-tuning on all 30 language directions together (lang. indep. fine-tune), we achieve a comparable gain in performance, results in a BLEU score of 19.6. Since this is achieved by a single model instead of 30 individual models, we continue with jointly training on all directions in the upcoming experiments on the big baseline model. Similar to findings on the small model, by fine-tuning on all the languages, we were able to improve the average BLEU score of the big baseline model from 15.4 to 27.4.

5.1.2 Adapters

As shown in the lower section of Table 2, by inserting adapters into the large baseline model and only training these modules, we achieve 23.0 BLEU on average. While the gain is less compared to full parameter tuning, the model preserves performance on the remaining tens of thousands directions.

As motivated previously (§3.1.3), the adapter layers are shared across the language directions rather than language-pair-specific. This could explain the performance gap to full parameter tuning.

System	jv-ta	ta-jv
baseline big	3.8	3.1
+ parallel data	8.5	7.9
+ syn. jv-ta data	10.0	8.2
+ syn. ta-jv data	15.0	10.7
+ both syn. data	15.9	9.7
+ both syn. data big	16.0	11.7

Table 3: Impact of synthetic data on jv↔ta, the lowest-resource language pair in this task.

5.2 Synthetic data

In the first set of experiments, we evaluate the influence of synthetic data only on the translations between Javanese (jv) and Tamil (ta), since this was the language pair with the least data (44K sentences). The synthetic data was always produced by the system fine-tuned on all the target language directions. The results are summarized Table 3. First, although the available parallel data is limited, we see a clear improvement of the baseline model when trained on the provided training data.

Adding the synthetic data (225K sentences for jv-ta and ta-jv each) does improve the performance compared to only using the parallel data. For both directions, the data generated from ta-jv was performing better than the other data. Since the combination of both directions performed the best for the jv-ta direction and reasonable good for the other direction and it is not clear how we should select the direction without perfoming test for each langauge pair, we continued the experiments by always using synthetic data generated by both directions.

By increasing the amount of synthetic data, so that the model is not trained on around 500k sentences by 2M sentence, we see additional gains to the best performance of 16.0 and 11.7 BLEU points for both directions. This is an improvement nearly by a factor of 3 compared to the baseline system.

Given the best system so far with 27.4 average BLEU, we continue fine-tuning with the additional synthetic data. This leads to an improvement to 27.9 BLEU on average. This improvement is significantly lower than expected, considering the general positive role of utilizing synthetic data. This has two potential reasons. First, as all other language directions have more data, the gains from the additional data could be reduced. Furthermore, the initial model is fine-tuned on the parallel data

Directions	# Sent.	Δ BLEU
jv↔ta	46K	+0.6
ms↔ta	297K	+0.1
jv↔ms	340K	+0.1
Overall	89M	+0.0

Table 4: The average change in BLEU after fine-tuning with residual removal. There is no gain in overall average BLEU, and limited gain in the top 3 lowest-resource directions.

of all the language pairs and therefore performing better.

5.3 Encouraging Similar Representations

Next we report the results of the approaches that promote language similarity as motivated in subsection 3.3.

5.3.1 Similarity Regularizer

Based on the best system trained with synthetic data (with 27.9 BLEU on average), we continue fine-tuning with the similarity regularizer described in §3.3.1. While we observe consistent increase in the similarity scores on the dev set, fine-tuning with the similarity regularizer alone does not improve the system further, achieving 27.7 BLEU on average. Nevertheless, we see gains when combining the similarity regularizer and the adapters described in §3.1.3. As adding the adapter layers expands the capacity of the existing model, we hypothesize the similarity regularizer could help combat overfitting. With this combination, we achieve an average of 28.1 BLEU.

5.3.2 Residual Removal

Based on the baseline big + fine-tune model (with 27.4 BLEU on average), we fine-tune once again using the residual-removal architecture described in §3.3.2. In Table 4, we summarize the average change in BLEU after this additional fine-tuning step. While there was no improvement in the overall average BLEU score, we observe some gain in the lowest-resource direction of jv↔ta which has 46K parallel data. However, the gain falls largely for the second and third lowest-resource directions.

5.4 Final System

System	BLEU
baseline big	15.4
+ fine-tune	27.4
+ synthetic data	27.9
+ sim. regularizer + adapter	28.1

Table 5: Average BLEU scores on FLoRes-101 devtest set on 30 directions of the final system.

The final system submitted to the evaluation is presented in Table 5. In a first step, we fine-tuned on the provided parallel data. Using this model, we created additional synthetic data. Fine-tuning the previous model on the parallel data and the synthetic data gave an additional improvement of 0.5 BLEU.

Finally, on top of the previous improvements, our best system uses the additional similarity regularization and adapters during training and further improves the average BLEU by 0.2 points to 28.1. The submitted system achieves 28.6 BLEU on average on the blind test set[3].

6 Conclusion

This paper summarizes our participation in the WMT 2021 large-scale multilingual translation task. We focus on Small Track #2 for English and 5 Southeast Asian languages. Building upon the provided baseline models, we achieved the largest gain from fine-tuning on the parallel data of all directions in this task. By further utilizing synthetic data and a combination of similarity regularization and adapters, we were able to further improve the system.

References

Antonios Anastasopoulos, Ondřej Bojar, Jacob Bremerman, Roldano Cattoni, Maha Elbayad, Marcello Federico, Xutai Ma, Satoshi Nakamura, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Sebastian Stüker, Katsuhito Sudoh, Marco Turchi, Alexander Waibel, Changhan Wang, and Matthew Wiesner. 2021. FINDINGS OF THE IWSLT 2021 EVALUATION CAMPAIGN. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 1–29, Bangkok, Thailand (online). Association for Computational Linguistics.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Roee Aharoni, Melvin Johnson, and Wolfgang Macherey. 2019a. The missing ingredient in zero-shot neural machine translation. *arXiv preprint arXiv:1903.07091*.

[3] https://dynabench.org/models/445

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. 2019b. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

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, Hong Kong, China. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzman, and Angela Fan. 2021. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. *arXiv preprint arXiv:2106.03193*.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

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.

Danni Liu, Jan Niehues, James Cross, Francisco Guzmán, and Xian Li. 2021. Improving zero-shot translation by disentangling positional information. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1259–1273, Online. Association for Computational Linguistics.

Ngoc-Quan Pham, Jan Niehues, Thanh-Le Ha, and Alexander Waibel. 2019. Improving zero-shot translation with language-independent constraints. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 13–23, Florence, Italy. Association for Computational Linguistics.

Ngoc-Quan Pham, Matthias Sperber, Elizabeth Salesky, Thanh-Le Ha, Jan Niehues, and Alexander Waibel. 2017. Kit's multilingual neural machine translation systems for iwslt 2017. In *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT 2017)*.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*.