Efficient Low-Resource Machine Translation: English to Maithili

Kuber Shahi
kshahi@ucsd.edu

1 Introduction

Most advancements in AI and NLP have primarily focused on the English language, leaving speakers of low-resource languages like Maithili underserved and unable to reap the benefit of the AI revolution fully. Despite being spoken by over 34 million people (Capitalist, 2024), Maithili lacks sufficient digital resources and machine learning models, resulting in a significant information gap.

This project explores approaches for building an English-to-Maithili machine translation model to bridge this divide. Such a model would empower Maithili speakers with access to critical resources in education, healthcare, and government services in their language, contributing to the broader mission of democratizing AI and fostering digital inclusion.

The steps planned for this project, along with their completion status, are as follows:

- Find or build a parallel English-Maithili dataset and preprocess it. Also, find if there are benchmark datasets for the translation tasks. **Done**
- Find SOTA (State-of-the-art) models for English-Maithili machine translation tasks and examine their performance on benchmark datasets. **Done**
- Explore and build smaller models keeping computational restraints in mind for efficient translation. **Done.** However, the chrf++ score for the translation task can be improved by further fine-tuning, which ultimately depends on the availability of computational resources.

2 Related Work

Research on the English-to-Maithili machine translation task is limited, with only a few notable studies published. These studies were primarily conducted as part of broader initiatives to develop multilingual machine translation models, where Maithili was included as one of the target languages. However, there is a noticeable lack of standalone research dedicated exclusively to English-to-Maithili translation.

The first notable work is the No Language Left Behind (NLLB) project (Costa-jussà et al., 2022), which aimed to develop a universal machine translation system encompassing 200 global languages, including Maithili. This project sought to address the global language divide by creating a robust multilingual model. As part of the effort, the NLLB team introduced seed training data and the FLORES-200 benchmark dataset, enabling evaluations of English-to-Maithili translation performance.

The second significant contribution is the IndicTrans2 project (Gala et al., 2023), which builds upon its predecessor IndicTrans (Ramesh et al., 2023). IndicTrans2 expanded multilingual translation capabilities to cover the 22 scheduled languages of India (Beelinguapp, 2023), including Maithili. This project introduced a comprehensive parallel training dataset, BPCC (Bharat Parallel Corpus Collection), and the IN22 benchmark dataset, specifically supporting English-to-Maithili translation tasks.

3 Your dataset

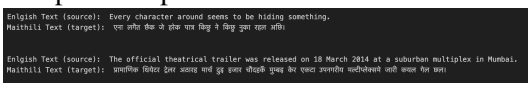
As discussed in the previous section, this project primarily focused on English-Maithili parallel texts from two multilingual datasets: NLLB (Costa-jussà et al., 2022) and BPCC (Gala et al., 2023). The NLLB dataset contains approximately 4.4 million parallel texts for English-to-Maithili translations, sourced from various online platforms. However, the NLLB dataset has not under-

gone thorough pre-processing and contains several issues, including inconsistent examples, untranslated segments, incorrect alignments, and low-quality or incomplete data entries.

In contrast, the BPCC dataset (Gala et al., 2023) improves upon NLLB by applying rigorous processing and quality checks. After filtering, the dataset was reduced to approximately 62,000 high-quality examples. Additionally, 24,000 examples were added from domains previously overlooked by NLLB, sourced from Wikipedia and daily conversational texts. This enhanced the dataset, making it more comprehensive for training and testing the English-to-Maithili translation model. Following further filtering to remove toxic examples, the final dataset consisted of approximately 68,000 high-quality, appropriate examples, which were used for training in this project. For testing, 1,024 English-Maithili parallel texts from the IN22 Gen benchmark dataset (Gala et al., 2023) were utilized. Further details on the training and testing datasets are provided below:

3.1 BPCC English-Maithili Training Dataset

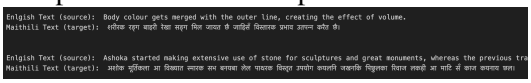
- Each training example is a pair of (English, Maithili) sentences. So, the total # of examples is 67658 pairs of sentences.
- The training dataset is divided into train-dev-split of (60892, 3383, 3383) respectively.
- Sample examples:



- The average number of words in an English source sentence is 16 and the target Maithili sentence is 15.
- As shown in the sample examples above, the translation model will take English sentences as input and output corresponding Maithili-translated sentences which will be compared against Maithili reference sentences for training.

3.2 IN22 Gen English-Maithili Benchmark Dataset

- Each benchmark example is a pair of (English, Maithili) sentences. So, the total # of examples is 1024 with an average of 24 words in both source and target sentences.
- Sample benchmark examples:



- For benchmarking, the translation model takes English sentences as input and outputs the corresponding Maithili-translated sentences which will be compared against the Maithili reference sentences.

3.3 Data Processing and annotation

Upon verifying the quality and alignment of the English-Maithili parallel texts from BPCC and IN22, no further processing or annotation was necessary. The IndicTrans2 project (Gala et al., 2023) had already performed extensive pre-processing, filtering, and quality checks.

4 Baselines

In this project, IndicTrans2 (Gala et al., 2023), the SOTA model for English-to-Maithili translation, was used as a baseline for comparison. IndicTrans2 is a multilingual machine translation model with 1.12 billion parameters, supporting translation in 22 official Indian languages, including Maithili. This project focuses specifically on its performance for the English-to-Maithili translation task.

As mentioned in the IndicTrans2 paper, chrF++ (Popović, 2017) scores were chosen as the evaluation metric due to their suitability for assessing translation quality in Indian and Nepali languages, which feature complex morphology and inflection. The chrF++ scores for IndicTrans2 on the IN22 Gen dataset for different languages are: To clarify, the goal of this project is to build

Languages	chrF++ score
English-Nepali	49.0
English-Hindi	56.7
English-Maithili	48.7

Table 1: chrF++ scored on IN22 Gen

a smaller and more efficient English-to-Maithili translational model that maintains comparable accuracy and precision to larger models such as IndicTrans2, rather than surpassing their performance. Improving upon the translation task itself is beyond the scope of this project due to computational constraints.

5 Approaches and Implementation

5.1 Different Approaches

At the start of this project, several approaches to building a small and efficient English-to-Maithili translation model were considered, as outlined in the project proposal. The pros and cons of each approach were evaluated, taking into account the computational constraints of the project and the challenges of Maithili being a low-resource language. Based on this analysis, a final approach was selected. The details of each approach, along with the reasoning for the chosen method, are outlined below:

- **Training an English-Maithili Model From Scratch**: This approach focused solely on English-Maithili translation rather than leveraging a multilingual model. This would allow the model to learn Maithili syntax and semantics without influence from other languages. However, the computational constraints required a compact model (4-5 million parameters), and no suitable Encoder-Decoder model was found within this range. While tiny-mBART (Shleifer, 2024) was considered, its 500k parameters were insufficient for the complexity of machine translation.

- **Pivot Approach Using English-Hindi and Hindi-Maithili Translation:** This method leveraged existing resources for Hindi by combining an English-Hindi translation model with a freshly trained Hindi-Maithili translation model. While the BPCC dataset provided seed data for training the Hindi-Maithili translation model, the approach faced the same challenge as previous methods — the unavailability of a compact model (4-5 million parameters) for the translation task.

- **Finetune English-Hindi or English-Nepali Model:** This approach leveraged the linguistic similarity between Maithili, Hindi, and Nepali as they all belong to the same family of Indo-Aryan language and use the same Devanagari script. This approach seemed most feasible and promising as the learning from English-Hindi or English-Nepali could be transferred. Consequently, we proceeded with this approach. We identified the Helsinki-NLP English-Hindi model (Helsinki-NLP, 2024), which has 76 million parameters, for fine-tuning on the English-Maithili BPCC dataset.

5.2 Implementation and Results:

5.2.1 Finetuning using LoRA:

Given the Helsinki-NLP English-Hindi model's 76 million parameters, fine-tuning the entire model on a local system was not feasible. To address this, we employed Low-Rank Adaptation (LoRA), a parameter-efficient fine-tuning technique, to adapt the Helsinki-NLP model for the English-Mathili Translation task on our local system. The implementation details, along with training and testing results, are provided below:

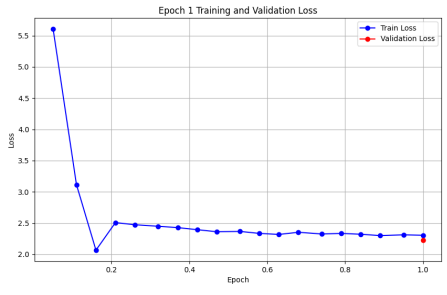
Epoch 1:

- LoRA configuration: LoraConfig (rank = 8, lora_alpha = 16, lora_dropout = 0.01, task_type = SEQ_2_SEQ_LM, bias = none, target_modules = ['q_proj', 'v_proj')

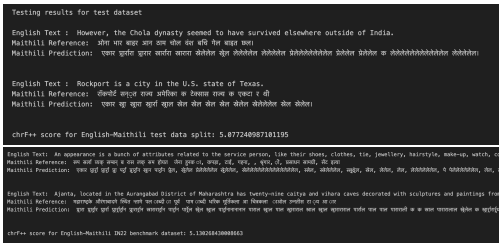
- Parameter Size (Before LoRA): 76381184, After LoRA: 76676096, Trainable Parameters: 294912 (0.38 %)

- Training arguments: the model was trained with a learning rate of 5e-5, batch size of 16, and gradient accumulation step as 2 for one epoch. The training time for one epoch was about 1 hour.

- Training Results:



- Testing Results: It took more than one hour each to predict to test on test data split (1200 random examples) and IN22 Gen (1024 examples) benchmark.



Dataset	chrF++ score
Test Split	5.07
IN22 Gen	5.13

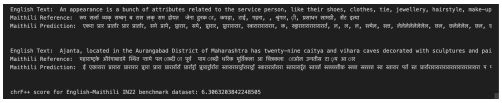
Table 2: Epoch 1 chrF++ scores

- The finetuned model can be found here.

Epoch 2:

- LoRA configuration: LoraConfig (rank = 16, lora_alpha = 32, lora_dropout = 0.1, task_type = SEQ_2_SEQ_LM, bias = "lora_only", target_modules=['q_proj', 'k_proj', 'v_proj')
- Parameter Size (Before LoRA): 76381184, After LoRA: 77265920, Trainable Parameters: 912384 (1,18 %)
- Training arguments: the model was trained with a learning rate of 5e-5, batch size of 16, and gradient accumulation step as 2 for two epochs. The training time for two epochs was more than 2 hours.
- Training Results:

- Testing Results: It took more than one hour each to predict to test on test data split (1200 random examples) and IN22 Gen (1024 examples) benchmark.



Dataset	chrF++ score
Test Split	6.71
IN22 Gen	6.30

Table 3: Epoch 2 chrF++ scores

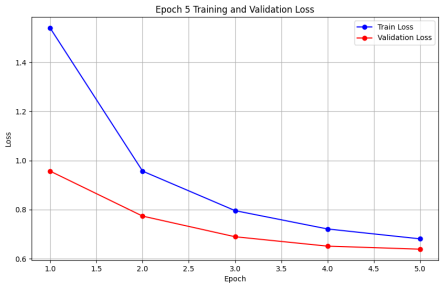
- The finetuned model can be found here.

5.2.2 Finetuning Whole Model:

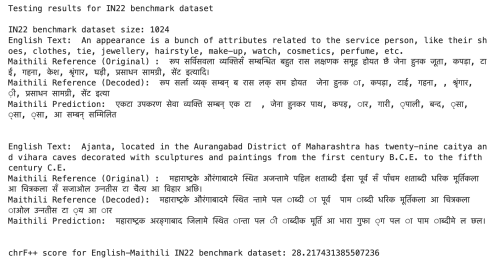
As finetuning the Helsiniki-NLP English-Hindi translation model using LoRA wasn't giving quality results, we chose to finetune the whole model on Kaggle's free-tier GPUs. The implementation details, along with the training and testing results, are provided below:

Epoch 5:

- Training arguments: the model was trained with a learning rate of 5e-5, batch size of 32, for five epochs. The training time for the whole model for five epochs was around 1.5 hours.
- Training Results:



- Testing Results: It took about 5 mins to test on test data split (1200 random examples) and about 3 mins on IN22 Gen benchmark (1024 examples).



Dataset	chrF++ score
Test Split	34.76
IN22 Gen	28.22

Table 4: Epoch 5 chrF++ scores

- The finetuned model can be found here.

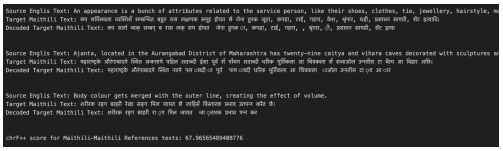
5.2.3 Code Submission:

Jupyter notebooks local_lora_finetuning, local_lora_testing, and kaggle_finetuning contain the relevant implementation code for this section.

6 Error analysis:

During finetuning, we recognized the following errors that gave us poor results with LoRA finetuning and decent results with whole model finetuning:

- **Faulty Tokenizer:** The tokenizer pretrained for the English-Hindi translation task may not be entirely suitable for English-Maithili translation, despite the linguistic and scriptural similarities between the two languages. During both training and testing, the English-Hindi tokenizer failed to fully and correctly tokenize the Maithili sentences. To investigate this, I encoded Maithili IN22 Gen benchmark sentences and then decoded them using the English-Hindi tokenizer.



As shown in the figure above, the target Maithili text and the decoded target Maithili text are not equivalent. Some characters are being ignored during the tokenization or encoding process. The chrF++ score for Maithili to decoded Maithili reference is only 68 when it should ideally be 100 (a perfect match).

- **More Robust English-Hindi Pretrained Model:** The pretrained English-Hindi model used for fine-tuning was not as effective as the state-of-the-art (SOTA) model, which may have impacted the final English-Maithili translation results. The SOTA model, IndicTrans2, achieved a chrF++ score of 56.7 on the IN22 Gen benchmark, as shown in Table 1. In contrast, the Helsinki-NLP English-Hindi translation model only scored 27.32 on chrF++. A more accurate and robust pretrained English-Hindi model could yield better results for the English-Maithili translation task.

- **More experimentation with LoRA**: As mentioned in class, LoRA does not generalize as well as whole model fine-tuning for complex tasks. Machine translation is a particularly complex task, requiring the model to learn the syntax and semantics of two languages. Achieving high performance on such tasks using LoRA fine-tuning alone may not be feasible.

  Further experimentation is needed to validate this. In this project, we fine-tuned the Helsinki-NLP English-Hindi model by applying LoRA to the query, key, and value projection layers. However, other trainable layers, such as the output layer and feed-forward layers, could also be fine-tuned using LoRA. Exploring different combinations of these layers would require additional computational resources.

- **More fine-tuning for the whole model:** While Kaggle's free-tier GPUs were sufficient for initial fine-tuning and provided decent preliminary results, we believe that with more fine-tuning and experimentation with training parameters, the chrF++ scores could be further improved with additional computational resources.

- **Code Submission:** Jupyter notebooks local_lora_finetuning and tokenizer_analysis contain the relevant analysis code for this section.

7 Conclusion

Our key takeaway from this project is that machine translation remains a complex task in the

NLP domain, particularly for low-resource languages. Fine-tuning the models proved to be a bottleneck in achieving better results for us. Moving forward, we plan to focus on developing a more robust tokenizer, either by training it on a Maithili monolingual corpus or extending the existing English-Hindi tokenizers. Additionally, we aim to explore more robust English-Hindi models and continue fine-tuning and experimentation with increased computational resources.

8 Acknowledgements

ChatGPT and GitHub Co-pilot were used as assistance to complete this project. Specifically:

- ChatGPT, along with other online resources, was used to gather information and better understand LoRA and its functionality.
- While writing code, mainly code completions and suggestions, and editing code, GitHub Pilot was used.
- Sections 1, 2, 5, and 6 of this report were initially drafted by a human and later edited and rewritten by ChatGPT to improve fluency and clarity.

References

Beelinguapp (2023). What are the 22 official languages spoken in india? Accessed: 2024-12-03.

Capitalist, V. (2024). 100 most spoken languages. Accessed: 2024-11-08.

Costa-jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., Sun, A., Wang, S., Wenzek, G., Youngblood, A., Akula, B., Barrault, L., Gonzalez, G. M., Hansanti, P., Hoffman, J., Jarrett, S., Sadagopan, K. R., Rowe, D., Spruit, S., Tran, C., Andrews, P., Ayan, N. F., Bhosale, S., Edunov, S., Fan, A., Gao, C., Goswami, V., Guzmán, F., Koehn, P., Mourachko, A., Ropers, C., Saleem, S., Schwenk, H., and Wang, J. (2022). No language left behind: Scaling human-centered machine translation.

Gala, J., Chitale, P. A., AK, R., Gumma, V., Doddapaneni, S., Kumar, A., Nawale, J., Sujatha, A., Puduppully, R., Raghavan, V., Kumar, P., Khapra, M. M., Dabre, R., and Kunchukuttan, A. (2023). Indictrans2: Towards high-quality and accessible machine translation models for all 22 scheduled indian languages.

Helsinki-NLP (2024). Helsinki-nlp/opus-mt-en-hi. https://huggingface.co/Helsinki-NLP/opus-mt-en-hi. Accessed: 2024-12-05.

Popović, M. (2017). chrF++: words helping character n-grams. In Bojar, O., Buck, C., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., and Kreutzer, J., editors, *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Ramesh, G., Doddapaneni, S., Bheemaraj, A., Jobanputra, M., AK, R., Sharma, A., Sahoo, S., Diddee, H., J, M., Kakwani, D., Kumar, N., Pradeep, A., Nagaraj, S., Deepak, K., Raghavan, V., Kunchukuttan, A., Kumar, P., and Khapra, M. S. (2023). Samanantar: The largest publicly available parallel corpora collection for 11 indic languages.

Shleifer, S. (2024). tiny-mbart: Hugging face model. https://huggingface.co/sshleifer/tiny-mbart. Accessed: 2024-12-05.