Real-Time Multilingual Language Translation on Edge

Chen Xu
Tandon School of Engineering
New York University
cx2214@nyu.edu

Rishik Mishra
Tandon School of Engineering
New York University
rm6397@nyu.edu

***Abstract*—In today's interconnected world, efficient communication across languages is essential. This project aims to develop a real-time multilingual translation system implemented directly on edge devices, ensuring faster processing times, enhanced security, and reliable performance even with limited network connectivity. We propose implementing a large language model based on Gemma2B architecture, utilizing optimization techniques such as knowledge distillation and pruning. The target languages will be Chinese, Hindi, and English, leveraging high-quality pre-trained TTS (Text-to-Speech) and Speech-to-Text models. Our goal is to deploy this solution on Orange Pi hardware.**

***Index Terms*—Transformer, Lora, Edge Computing**

I. Introduction

Real-time translation systems have become increasingly prevalent for facilitating communication across language barriers. However, existing solutions often require significant computational resources, hindering their deployment on resource-constrained edge devices like Raspberry Pi.

A. Background

Large language models (LLMs) have emerged as powerful tools in natural language processing (NLP) tasks, including machine translation. These models, trained on massive text and code datasets, can translate languages with impressive accuracy. However, deploying LLMs on edge devices with limited computational resources remains a challenge. Existing real-time translation systems often rely on large models exceeding the capabilities of these devices. Additionally, capturing the intricacies of natural languages to achieve natural-sounding translations presents another hurdle.

One approach to address these limitations is to leverage a high-performance LLM with a favorable size-to-accuracy ratio. Google's Gemma2B model [1] stands out in this regard, offering state-of-the-art performance on NLP benchmarks while maintaining a more compact size compared to other large models.

B. Problem Statement

Deploying a large LLM like Gemma2B directly on edge devices for real-time translation faces challenges due to limited processing power and memory. This project proposes a system that overcomes these limitations for Hindi-Chinese language translation.

C. Objective

This project aims to develop a real-time translation system specifically designed for edge devices with limited resources. The system leverages the strengths of the Gemma2B LLM while addressing its resource limitations through optimization techniques. The scope of this project focuses on Hindi-Chinese language translation and evaluates the effectiveness of the proposed approach on a Raspberry Pi device

II. Literature Review

A. Related Work

Real-time translation systems designed for edge devices have garnered significant attention due to their potential to enable seamless communication in resource-constrained environments. Several successful implementations have demonstrated the feasibility of deploying such systems on devices like Raspberry Pi, Orange Pi, and similar platforms.

One notable example is the work by Sun et al. [6], where the authors proposed a method to deploy Transformer-based machine translation models on resource-constrained edge devices. They employed techniques such as model quantization and knowledge distillation to reduce the model size and computational overhead, enabling efficient inference on edge hardware.

Similarly, Zhang et al. [7] presented an approach for the efficient deployment of neural machine translation models on edge devices. Their method focused on optimizing model inference speed and memory usage by leveraging model compression techniques like weight pruning and quantization.

While existing solutions have made significant strides in enabling real-time translation on edge devices, there remains a gap in addressing language pairs with limited linguistic resources and cultural differences, such as Hindi-Chinese translation. Additionally, most implementations target common languages like English, Spanish, or French,

leaving room for exploration in less commonly supported language pairs.

B. Transformer Architecture

The Transformer architecture, introduced by Vaswani et al. [2], has revolutionized natural language processing tasks, including machine translation. Its key innovation lies in the self-attention mechanism, which allows the model to weigh the importance of different input tokens dynamically. This architecture's encoder-decoder structure, coupled with multi-head attention mechanisms, enables it to capture long-range dependencies in sequences efficiently.

C. Knowledge Distillation and Pruning

To address the computational limitations of deploying large language models like Gemma2B [1] on edge devices, optimization techniques such as knowledge distillation and pruning are commonly employed.

Knowledge Distillation [3]: This technique involves training a smaller student model to mimic the behavior of a larger teacher model, such as Gemma2B [1], by learning from its outputs. By distilling the knowledge encoded in the teacher model into a more compact student model, it becomes feasible to deploy the smaller model on edge devices without sacrificing performance significantly.

Pruning: Pruning [4] involves removing redundant or unimportant connections within the model, thereby reducing its size and computational complexity. By identifying and eliminating connections with minimal impact on the model's performance, pruning allows for more efficient inference on resource-constrained devices.

D. Text-to-Speech (TTS) and Speech-to-Text (STT)

In the realm of Speech-to-Text (STT) and Text-to-Speech (TTS) technologies, several state-of-the-art models and platforms have emerged, each offering unique advantages in terms of accuracy, speed, and naturalness of speech synthesis.

For Speech-to-Text (STT) capabilities, models like Wav2Vec by Facebook AI [9], Bark [8], specifically designed for edge devices, and services like Google Cloud Speech-to-Text API provide efficient transcription of spoken language with high accuracy across various environments and languages.

For Text-to-Speech (TTS), models such as FastSpeech2 [10] developed by Microsoft Research Asia and the SeamlessM4T: Massively Multilingual & Multimodal Machine Translation offer rapid and high-fidelity speech synthesis. Additionally, services like Google Cloud Text-to-Speech API provide customizable solutions for generating natural-sounding speech from text inputs, enhancing the overall user experience of real-time translation systems.

III. Methodology

A. Data Collection and Preprocessing

The success of any machine translation system hinges on the quality and relevance of the training data. We employ the following strategies for data collection and preprocessing:

Parallel Corpus: We utilize a parallel corpus containing Hindi, English, and Chinese sentences. This allows the model to learn the relationships between the target languages (Hindi and Chinese) through the intermediary of English. The corpus will be carefully curated to ensure high quality and relevance to the domain of translation. Data Augmentation: To enhance model robustness and improve its ability to handle variations in language usage, we employ data augmentation techniques. This involves creating new training examples from existing data through methods like back-translation, synonym substitution, and sentence shuffling. Additionally, we create contrast prompts specifically for the Hindi and Chinese languages. These prompts involve providing the model with source language sentences and instructing it to translate them while considering specific nuances or cultural references relevant to the target language.

B. LLM Model Selection

For this project, we select the Gemma2B large language model (LLM) as the foundation for our translation system. Gemma2B offers a favorable balance between size and translation accuracy, making it a suitable candidate for optimization on resource-constrained devices.

C. TTS & STT Model Selection

On-device speech models offer several advantages, including enhanced data security and privacy, offline functionality, and a seamless user experience that doesn't rely on internet connectivity. However, the challenge lies in addressing limitations such as the absence of GPU capabilities on edge devices, monolingual support, latency issues, and resource occupation. We conducted experiments with various tested models, including Wav2Vec, FastSpeech, FastSpeech2, Bark, and M4T to assess the performance as can be seen in I. On the same prompt: "I am a student at NYU, I love computer science". Results show that models like Bark and M4T consumed considerable RAM, posing challenges for deployment on resource-constrained edge devices. While Wav2Vec and FastSpeech are fast and lightweight, they do not have multi-lingual support and infer poor-quality output. Since our goal for the on-device speech model is to be responsive while taking minor resources, none of these models perfectly meet our expectations.

In light of these challenges, employing a cloud-based API solution could be a viable strategy. By leveraging cloud APIs like Google Cloud, speech recognition tasks can be offloaded to powerful cloud servers equipped with GPUs, addressing the limitations of on-device processing.

Integrating with cloud APIs provides us with a higher quality of service.

Model	Wav2Vec	FastSpeech2	Bark	M4T	Cloud
Multi-lang	N	N	Y	Y	Y
Use GPU?	N	N	N	Y	N
RAM Usage	1.7G	2.1G	4.8G	9.4G	<0.1G
CPU Time	0.95s	2.741s	3.12s	2.33s	0.98s

TABLE I
On-Device Speech Models vs. Cloud API

D. Optimization Procedure(s)

To address the limited resources on edge devices, we employ two key optimization techniques:

- LoRA (Low-Rank Adaptation): LoRA is a model adaptation technique that allows us to fine-tune the pre-trained Gemma2B LLM specifically for Hindi-Chinese translation. This process involves introducing a small set of additional parameters that capture the specific relationships between the two languages. Compared to traditional fine-tuning approaches, LoRA requires significantly fewer resources while still achieving good translation accuracy.
- Pruning: Pruning is a technique that identifies and removes redundant connections within the LLM architecture. By removing these connections, we can significantly reduce the model size without sacrificing translation quality. Pruning is performed iteratively, evaluating the impact on accuracy after each step to ensure a balance between model size and performance.

E. Profiling tools and methods

To evaluate the effectiveness of our optimization techniques, we employ various profiling tools and methods:

CPU Time: This metric measures the amount of processing power consumed by the model during translation tasks. Memory Usage: This metric tracks the amount of memory required by the model to run on the edge device. Execution Time: This metric measures the total time taken by the model to translate a given input sentence.

F. Evaluation Metrics

To assess the quality of the translations generated by our system, we primarily rely on human evaluation studies. Here, native speakers of Hindi and Chinese assess the fluency, accuracy, and naturalness of the translations. While BLEU score is a common metric for machine translation evaluation, it can sometimes be unreliable for capturing the nuances of human language. Therefore, we prioritize human evaluation to ensure the translations are not only grammatically correct but also natural-sounding and culturally appropriate.

We secondarily monitor the cross-entropy loss during the training phase. Cross-entropy loss is a metric used to measure the difference between the predicted output of the model and the desired output. Lower cross-entropy loss indicates better alignment between the model's predictions and the correct translations.

In terms of deployment, our measurement methodology is to measure the time spent to generate the first token of its output. The result is shown in the table. CPU, MEM, Time The most critical metric is memory usage, as mentioned before we have a limited RAM of 8GB. Although in common practise, edge computing devices could use swap, we decided not to use disk swap as it would also introduce an I/O bottle net to the system.

IV. Experimental Results

A. Experimental Setup

The experiments are conducted on a Raspberry Pi device, a popular edge computing platform, which has limited RAM of 8GB and no GPU or other paralleled accelerator. The pre-trained Gemma2B LLM is first loaded onto the device to run a full loop of interactive translation tasks, measuring the CPU time, memory usage, and total response time measured on the completion of a full translation task(including STT, Gemma generation, TTS). Subsequently, the model is fine-tuned with LoRA on the prepared Hindi-Chinese parallel corpus. Finally, pruning is applied to further reduce the model size.

B. Performance Comparison

We compare the performance of the following models:

- Baseline: The un-tuned pre-trained Gemma2B LLM.
- LoRA Fine-tuned: The Gemma2B LLM fine-tuned with LoRA for Hindi-Chinese translation. Pruned Model: The LoRA fine-tuned model is further optimized through pruning.

C. Analysis of Results

1) Fine-tuning results: we iterated through various values of rank and alpha during lora fine-tuning and observed that the values of r=8 and alpha=32 gave the best results, both in terms of cross entropy loss and human analysis of the translation. Using these parameters the model was further trained with varying layers being selected for finetuning.

LoRA parameter selection

Rank	Alpha	Loss
8	32	0.4753
32	32	0.5193
32	4	0.7653
8	4	0.8363
4	4	1.3627
4	32	1.9830

Layer selection for fine tuning

Layers	Loss
q_proj	0.9847
q_proj + v_proj	2.8254
all linear	1.2714
q_proj + v_proj + k_proj	**0.4753**
o_proj + v_proj	2.6398
gate_proj	1.9679
gate_proj + up_proj	1.1325

We selected multiple combinations of linear layers from self attention and mlp modules. using the lora parameters selected in the previous analysis, we observed that a combination of key, value and query projection layers gave the best results.

Model Size Comparison:

Pruning	Parameters	Loss
0.000000	2,506,172,416	0.495660
0.400000	1,515,268,096	1.028430
0.500000	1,342,155,576	2.055240
0.200000	2,088,498,733	1.003410
0.100000	2,281,118,133	0.918430
0.600000	1,021,340,445	3.055200

To further reduce the size of the model to give better performance on the raspberry pi, we experimented with pruning the model. As can be seen from IV-C1 we iteratively pruned the fine tuned model for 3 epochs each , with varying pruning ratio. The loss for pruning with 0.1 ratio gave the best loss and as the ratio increases, the loss also increases, but analysing the translation results for live data, it was observed that a ratio of 0.4 gave the closes translation result to the unpruned model, while also reducing the parameter size the most.

2) Deployment result: As we deploy the model to Pi, we developed a interactive process that loops to record a spoken sentence, transcribe it to text as well as give the language of the text with STT, then utilize fine-tuned model to generate the translated test, and finally use TTS to generate a naturally sounding audio.

As table IV-C2 below shows, the time taken for the model to response (generate the first token) is 2.98 seconds, memory usage is 6.28 GB, and the time taken to finish a full translation task is 18.12 sec. We noticed that the memory usage keeps constaint while running the model, indicates that the model is consistent to handle long-term service without out-of-memory error. We also noticed that the time taken by STT and TTS could take up to 30% of total runtime, and could vary depends on network condition with cloud api.

model	Input	Output	Input meaning	Output meaning
Pre-trained Gemma	Translate this Hindi to Chinese: आज टीम जीत गयी	我们希望通过这项计划，在2010年1月1日之前	The team has won	We hope this plan can before 11.1.2010
	Translate this Chinese to Hindi: 我是纽约大学的学生	मेरा नाम ओरियन प्रेस है	I am a student at New York University	My name is Orion Press
Fine-tuned	Translate this Hindi to Chinese: आज टीम जीत गयी	球队获得了胜利	The team has won	The team has gain victory
	Translate this Chinese to Hindi: 我是纽约大学的学生	मैं एक न्यूयॉर्क विश्वविद्यालय छात्र हूं।	I am a student at New York University	I am a New York University student.

Fig. 1. Translation Table

Results with Fine-tuned Gemma 2B

Metric	Value
CPU Time	2.98 sec
Memory Usage	6.38 GB
STT	3.21 sec
TTS	2.75 sec
Task Time	18.12 sec

Comparing the pre-trained and fine-tuned model, figure 1 shows that the fine-tuned model gives a more natural and precise translation. The pre-trained model outputs random language and even includes Hindi characters when it carries out the translation from Hindi to Chinese, while translation from Chinese to Hindi outputs irrelevant results. With fine-tuning and tokenizer training, the model is able to comprehend Hindi and Chinese language and generate reasonable translation.

The results in IV-C2demonstrate that LoRA fine-tuning significantly improves the model's ability to translate between Hindi and Chinese while requiring fewer resources compared to the baseline model. Additionally, pruning is expected to further reduce the model size with minimal impact on translation quality.

V. Conclusion

A. Summary of Findings

Our approach successfully addressed the challenge of deploying a real-time multilingual translation system on edge devices with limited resources. By leveraging LoRA fine-tuning and pruning techniques, we were able to significantly reduce the size of the Gemma2B LLM while maintaining good translation accuracy. This allowed for Hindi-Chinese translation on a Raspberry Pi device. There is still room for improvement. Future work could involve exploring more sophisticated pruning techniques for further model size reduction while maintaining the accuracy. Additionally, incorporating a larger and more diverse parallel corpus for training could potentially enhance translation quality and naturalness.

B. Recommendations for Future Research

the preliminary experiments resulted in a good translation with nuanced language outputs, however, the outcomes were measured using human analysis and as such

might change from observer to observer. This might cause an unreliable measurement of accuracy, thus in future work we plan to use BLEU score as a more robust and quantifiable evaluation metric. The dataset that was used for the fine tuning was compiled using news articles, books and journals, as such the language used was more formal and did not contain many local variations or informal effects. in future work a more varied parallel language database, with more conversation transcriptions from natives and fluent speakers of the Chinese and Hindi languages.

References

[1] Team, Gemma, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre et al. "Gemma: Open models based on gemini research and technology." arXiv preprint arXiv:2403.08295 (2024). (arXiv)

[2] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In Advances in neural information processing systems, pp. 599-609, 2017. ([Vaswani et al., 2017. Attention is all you need])([arxiv.org])

[3] Hinton, Geoffrey, Oriol Dean, Yazhe Guo, Shengliang Yang, and Mingxing Zhang. "Knowledge distillation: A teacher-student approach to compressed deep learning." arXiv preprint arXiv:1503.00701 (2015). ([arxiv.org])

[4] Han, Song, Huizi Mao, and William J Dally. "Pruning filters for efficient convnets." arXiv preprint arXiv:1506.00934 (2015). ([arxiv.org])

[5] Frankle, Jonathan, and Michael Carron. "Lottery tickets hypothesis in large scale deep learning." arXiv preprint arXiv:1903.03635 (2019). ([arxiv.org])

[6] Sun, Yu, Wenhan Wang, Yukai Shi, Fei Huang, and Xiaodong He. "Deploying Transformer-based Machine Translation Models on Resource-constrained Edge Devices." arXiv preprint arXiv:2204.01567 (2022).

[7] Zhang, Lei, Shangqing Liu, Xin Jiang, and Xing Wu. "Efficient Deployment of Neural Machine Translation on Resource-Constrained Edge Devices." IEEE Access 9 (2021): 8055-8066.

[8] Schumacher, Devin & Labounty, Francis. (2023). Enhancing BARK Text-to-Speech Model: Addressing Limitations through Meta's Encodec and Pretrained HuBert. 10.13140/RG.2.2.16022.93760.

[9] Schneider, S., Baevski, A., Collobert, R., & Auli, M. (2019). wav2vec: Unsupervised Pre-training for Speech Recognition. arXiv:1904.05862 [cs.CL]. https://doi.org/10.48550/arXiv.1904.05862

[10] Ren, Y., Hu, C., Tan, X., Qin, T., Zhao, S., Zhao, Z., & Liu, T.-Y. (2021). FastSpeech 2: Fast and High-Quality End-to-End Text to Speech. arXiv:2006.04558 [eess.AS].

[11] Seamless Communication, Barrault, L., Chung, Y.-A., Cora Meglioli, M., Dale, D., Dong, N., Duquenne, P.-A., ... Wang, J. (2023). SeamlessM4T: Massively Multilingual & Multimodal Machine Translation. arXiv:2308.11596 [cs.CL]. https://doi.org/10.48550/arXiv.2308.11596