ENG2TEL Transalatron:English to Telugu Translation Project -Bridging Linguistic Gaps

Alokam Gnaneswara Sai & Onteru Prabhas Reddy & Pallekonda Naveen kumar
Indian Institute of Science
Bengaluru, KA, India
{gnaneswaras,prabhasreddy,kpnaveen}@iisc.ac.in
GitHub Repository link: GitHub Repo
Models and Datasets we published: Llama Model , Telugu translated alpaca dataset.

Abstract

While Neural Machine Translation(NMT) excels in translating high-resource languages, its performance still lags when it comes to low-resource languages like Telugu. We attempted to enhance the quality of translations from English to Telugu, overcoming the challenges posed by limited linguistic resources and training data, by utilizing the capabilities of Pre-trained Large Language Models (LLMs). We utilized various fine-tuning paradigms across different pre-trained Large Language Models (LLMs) and noticed several intriguing findings. For fine-tuning, we have considered a subset of the Samanantar (1)dataset due to in short of computational resources and FLORES22 as a test set. We have considered a variety of pre-trained LLMs like NLLB200,mBART50, Llama2 and mt5. Some of these models like NLLB and mBART are good at handling many languages for translation, including Telugu, while others like Llama are newer and haven't been trained much in Telugu yet.

1 Introduction

In recent years, language translation has made significant strides, thanks to advancements in deep learning. These improvements have greatly enhanced translation accuracy and fluency, especially for widely spoken languages like English and French. However, low-resource Indian languages like Telugu, spoken by an estimated 96 million people in India, have not seen the same progress comparatively.

The main challenge in developing effective translation systems for languages like Telugu is the lack of enough data and resources. Also, there aren't many researchers focusing on these languages. This lack of attention has led to a big difference in translation quality as compared to other popular languages.Even current models like GPT-3 lacks understanding of the languages like Telugu. Telugu speakers often struggle to find accurate translation tools, making it hard for them to access information.

Given Telugu's importance in Indian culture, it's crucial to bridge this translation gap and provide better access to information for Telugu speakers. In this work, our aim is to investigate the performance of Fine-tuned Language Models (LLMs) on MT tasks focusing on the Telugu language. We specifically target medium-sized LLMs, including mBART and NLLB (2) with a parameter size of 0.61 billion, as well as the larger Llama-2 7B model.

These models typically prioritize translating one sentence at a time, potentially neglecting discourse phenomena and the broader context. Our objective is to investigate the impact of fine-tuning using an English-Telugu dataset, considering the possibility of both improving and not improving translation quality.

2 Related Work

Recent literature has begun to explore the application of LLMs for MT on low-resource languages, an area that remained relatively under-explored until now. Their findings suggest that while these decoder-only LLMs are competitive, they still lag behind when compared to the encoder-decoder-based multilingual language model like NLLB.
Fine-tuning involves extending the training of the LLMs using additional, task-specific data.This is particularly beneficial when such tailored datasets are available and this is proven to be effective for down stream tasks like translation for low resource languages.Many efficient ways are proposed to efficiently finetune the pretrained LLMs like LoRA.

IndicBART (3)is a multilingual, sequence-to-sequence pre- trained model focusing on Indic languages and English. It currently supports 11 Indian languages and is based on the mBART architecture. Tamil LLaMA 7B (4) base model enhancing the open-source LLaMA model with an addition of 16,000 Tamil tokens, aimed to achieve superior text generation and comprehension in the Tamil language. They employ the LoRA (5) methodology for efficient model training on a comprehensive Tamil corpus, ensuring computational feasibility and model robustness. MalayaLLM focused on translations for Malayalam language, dedicated training on a Malayalam dataset. They had undertaken the continuous pre-training of the LLAMA2 model using a comprehensive Malayalam dataset.

Additionally, the "Samanantar" dataset has emerged as a valuable resource for sentence-level translation between English and Telugu. With parallel English-Telugu sentences of more than 4.6M sentences, this dataset facilitates the development of accurate and contextually relevant translation models. Samanantar enables researchers to overcome the challenges posed by data scarcity, paving the way for improved language processing solutions tailored specifically for Telugu.

3 Methodology

The NLLB , mBART50 models are variants of the Transformer architecture specifically designed for multilingual translation tasks, whereas Llama is based on decoder-based architecture like GPT. They utilizes a shared vocabulary across multiple languages and are pre-trained on a large corpus of text from various languages. Fine-tuning allows us to adapt these models to specific translation tasks by providing them with domain-specific data.

3.1 **Llama2-7b model**

LLaMA is a family of autoregressive LLM, released by Meta AI,in February 2023. Here we considered Llama2 which had 7 billion trainable parameters. The consideration of the Llama model is because Llama is one of the best open-source LLM for fine-tuning, which is proven to be effective.

The Llama fine-tuning procedure is as follows:

3.1.1 Customized Tokenizer

We hypothesized that a bottleneck in performance could occur for low-resource languages like Telugu due to the tokenizer. The Llama tokenizer had a vocabulary size of 32k tokens, with fewer tokens dedicated to low-resource languages like Telugu since it has to support other high-resource languages like English. Additionally, we observed that each Telugu word was divided into an average of 20.03 tokens in the Samanantar dataset which is worse, whereas only 1.6 for English. This highlights the linguistic gap between them. To address this, we added 20k new quality tokens to the Llama tokenizer, sourced from the Telugu version of CulturaX.

The process of adding new tokens is as follows:
a) To generate new quality tokens, we used the Sentence Piece tokenizer on the CulturaX dataset.

b) These tokens were then integrated into the Llama tokenizer, and the Llama pretrained model embedding layer and softmax layers were resized accordingly.

3.1.2 PreTraining Phase

In the training phase of Llama, it might have encountered a minimum number of Telugu training data compared to other languages. So we have again Pretrained the Llama model on the Samanantar Dataset by considering a subset of 60k samples. Formally, the objective during this pretraining phase is to maximize the likelihood of the entire sequence, as represented by:

$$P(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} P(x_t | x_1, x_2, \ldots, x_{t-1})$$

Where:

- $x_1, x_2, \ldots, x_T$: The individual tokens that constitute the sequence.
- $P(x_t | x_1, x_2, \ldots, x_{t-1})$: Represents the conditional probability of the token x_t, which depends on the preceding tokens in the sequence.

3.1.3 Fine-tuning

For fine-tuning we have considered the alpaca dataset which typically consists of the following format.
Instruction Template for fine-tuning:

Translate the following English sentence into Telugu language.
Instruction: *instruction*
Input: *input*
Response: *output*

We have translated the standard alpaca dataset which originally consisted of 52k instructions using Google Translate from English to Telugu and then used for fine-tuning phase.

3.1.4 LoRA Approach and Quantization

The above mentioned PreTraining and fine-tuning phase is happened in a lightweight finetuning paradigm called LoRA. LoRA separates pre-trained and fine-tuned parameters in both phases by decomposing the weight matrix $X \in \mathbb{R}^{m \times n}$ into U and V, where $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{r \times n}$.

This reduces the parameter space from $m \times n$ to $(m \times r) + (r \times n)$, enabling faster training and lower computational cost compared to full fine-tuning. Then these values will be merged with the pretrained weights of the Llama model to obtain one finetuned model.

The Llama model is initially loaded in an 8-bit quantization format, which offers efficiency in terms of memory usage. Moreover, it has been demonstrated that this quantization method does not significantly compromise the model's performance.

3.2 **NLLB model**

NLLB is a recent encoder-decoder-based LLM developed by META, capable of multilingual translation across approximately 200 languages. We chose the NLLB model as a base for the project because it's one of the best models for translation, including Telugu.

After evaluating the NLLB pretrained model and tokenizer, during the tokenization phase, we observed that many Telugu sentences contained unknown tokens, which could diminish

performance. So we preprocessed the Telugu text accordingly to eliminate unknown tokens throughout the dataset.

We experimented with NLLB in the following ways:
1) We directly fine-tuned the NLLB model on a random subset of 60k samples from the Samanantar dataset.
2) The NLLB tokenizer had a vocabulary size of 256k tokens, with fewer tokens dedicated to low-resource languages like Telugu since it supports over 200 languages. Additionally, we observed that each Telugu word was divided into an average of 2.65 tokens in the Samanantar dataset, whereas only 1.42 for English.So a similar approach is followed like Llama where we added 16k new quality tokens to the NLLB tokenizer, sourced from the Telugu version of CulturaX. The resulting new model was fine-tuned similarly to the pretrained NLLB model.

3.3 **mBART model**

mBART (6) is pretrained as a denoising autoencoder, trained to predict the original text X from its corrupted version $g(X)$.mBART is one of the first methods for pretraining a complete sequence-to-sequence model by denoising full texts in multiple languages, while previous approaches have focused only on the encoder, decoder, or reconstructing parts of the text.

Before fine-tuning the pretrained mBART50 model, we preprocess the parallel corpus data. This involves tokenization, where we split the sentences into individual tokens using the mBART50 tokenizer. Additionally, we apply any necessary cleaning and normalization steps to prepare the data for training.

We have conducted full parameter fine-tuning on mBART-50 using a subset of 60k samples from the Samanantar dataset for Telugu.

3.4 **mT5 Transformer model**

The mT5(7) transformer is an immense multilingual text-to-text pre-trained transformer.The mT5 is pre-trained on immense amounts of data already. mT5 supports a maximum token length of 20 tokens. This is what we have used for our model. Longer sentences have been truncated to 20 tokens and Out-Of-Vocabulary (OOV) tokens are discarded. We have conducted full parameter fine-tuning on mt5 using a subset of 50k samples from the Samanantar dataset for Telugu.

4 Experiemental Setup

4.1 **Datasets**

Our fine-tuning set includes the Samanantar which consists of 4.95 million english-telugu sentence pairs. We have utilized only a subset of total dataset due to the in short of computational resources. The dev and test sets are the FLORES-22 Indic dev set and the IN22 test set, These datasets are constructed from documents, thus enabling a natural evaluation of sentence-level translation. While adding tokens in the pretrained Tokenizers we used the Telugu version of the CulturaX dataset. Table 1 summarizes the statistics of the datasets used in the project.

4.2 **Pretrained LLMs**

We investigated a varied collection of pretrained LLMs accessible on HuggingFace. This collection comprises two distinct LLMs, each trained on either English-centric or multilingual data and available in multiple versions with varying parameter sizes. Table 2 summarizes the models included in our study.

Table 1: Dataset Statistics

Usage	Datasets	#sentences
Finetune	Samanantar	5M
Test	FLORES Dev Set	1K
Train	Samanantar	5M
Tokenizer	CultureX(te)	18L
Finetune	alpaca dataset	52k

Table 2: List of evaluated LLMs

Model	Size(Billions)
Llama2-7b	7B
NLLB-200	600M
mBART-50	611M
MT5	300M

4.3 **Fine-tuning Setup**

Models with less than 1 billion parameters are trained on a single NVIDIA A100 GPU with 32GB of memory. We configure the learning rate to 5e-4 and employ the AdamW optimizer (8) for the fine-tuning process. A batch size of 8 is used and the number of epochs is set to 1. The tokenizer is configured with a **max-input-length** of 128 for the source language (English) and a **max-target-length** of 128 for the target language (Telugu).

For larger models like Llama2-7b, we have conducted distributed training across 8 NVIDIA A100 GPUs, each with 32GB of memory. The pretraining took around 40 hours for one epoch on the whole samanantar dataset and fine-tuning took around 4 hours for one epoch on the alpaca dataset. The configurations for the training of Llama are as follows:

Table 3: Pre-Training Configurations

Parameter	Value
Training Data	11GB
Epochs	1
Per device train batch size	16
Batch Size	512
Quantization	8-bit

Table 4: Fine-tuning Configurations

Parameter	Value
Training Data	45k
Epochs	1
Batch Size	512
Dropout Rate	0.1

Table 5: Common Hyperparameters

Parameter	Value
Initial Learning Rate	2e-4
Max Sequence Length	512
LoRA Rank	64
LoRA Alpha	128
LoRA Target Modules	q_proj,v_proj,k_proj,o_proj,gate_proj,down_proj,up_proj
Training Precision	FP16

4.4 **Evaluation Metrics**

We incorporated metrics derived from the "compare-mt" tool, designed for evaluating machine translation, to compare the outputs of multiple systems and assess their performance.

4.4.1 Aggregate Scores :

BLEU Score(9): Measures the similarity between the candidate translation and reference translations using n-gram overlap.

4.4.2 Word Accuracy Analysis :

F-measure by Frequency Bucket: Evaluates the accuracy of word predictions based on the frequency of words in the reference translation.

4.4.3 Sentence Bucket Analysis :

- Bucketing Sentences: Groups sentences based on various statistics such as sentence BLEU, length difference with the reference, and overall length.
- Statistics by Bucket: Calculates statistics (e.g., number of sentences, BLEU score) for each bucket to analyze translation quality across different sentence characteristics.

4.4.4 N-gram Difference Analysis

Consistency of N-gram Translation: Identifies which n-grams one system consistently translates better than the other.

4.4.5 Sentence Examples

Identifying Superior Translations: Finds sentences where one system performs better than the other according to sentence BLEU.

5 Results

5.1 Tokenizer

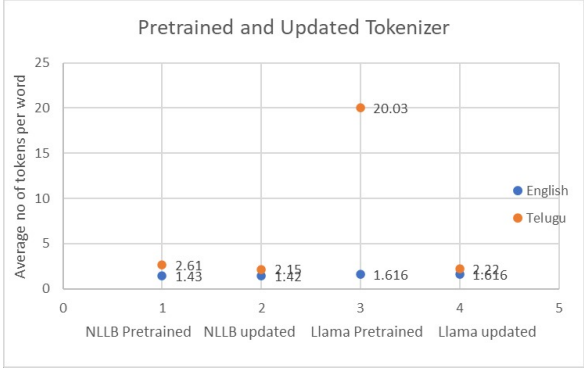

Figure 1: Pretrained vs Updated Tokenizer

The above (Figure-1) shows the average number of tokens a word getting divided by using pretrained and customized tokenizers. Lower the number,the more a LLM is able to recognize the language.As we can observe that by using customized Tokenizers the LLMs are able to recognize more the target language.

5.2 Scores

We've utilized the Flores Dev Test set of 988 English-to-Telugu translations extensively for evaluating our machine translation models.

- Table 6 shows the BLEU score analysis of raw models
- Table 7 shows the BLEU score analysis for the model trained on 100,000 examples.NLLB(a)is model fine-tuned without adding customized tokens where as NLLB(b) is model fine-tuned after adding customized tokens.
- Table 8 displays the length ratio analysis between the reference and translated text.A length ratio of 1 indicates a perfect length translation.
- Table 10provides a sentence bucket analysis based on the length of sentences, with the statistic type being BLEU score
- Table 11 presents the analysis of sentence buckets based on the length difference between the output and reference translations.

Table 6: BLEU Score Analysis of raw models

Metric	mBART	mT5	NLLB	Llama2-7B	Google*
BLEU	4.5	5.4	17	0.6	30.6

Table 7: BLEU Score Analysis after fine-tuning

Metric	mBART	mT5	NLLB(a)	NLLB(b)	Llama2-7B
BLEU	8.5402	7.83	12.8	8.63	3.2

Table 8: Length Ratio Analysis

Metric	mBART	mT5	NLLB	Llama2-7B
Length Ratio	0.8926	0.9146	0.9426	0.8505

Table 9: Word Accuracy Analysis

Frequency	**mBART**	**mT5**	**NLLB**
< 1	0.0000	0.0000	0.0000
1	0.3174	0.2864	0.3512
2	0.4189	0.3891	0.4231
3	0.4619	0.4282	0.4522
4	0.4743	0.4411	0.491
$[5, 10)$	0.4911	0.4707	0.514
$[10, 100)$	0.4656	0.4536	0.483
$[100, 1000)$	0.5357	0.5300	0.554
≥ 1000	0.0000	0.0000	0.0000

Table 10: Sentence Bucket Analysis (bucket type: length, statistic type: BLEU)

Length	**mBART**	**mT5**
< 10	6.4854	7.8442
$[10, 20)$	8.6437	8.4886
$[20, 30)$	6.5985	6.8427
$[30, 40)$	7.6747	6.6193
$[40, 50)$	18.7866	29.2017
$[50, 60)$	0.0000	0.0000
≥ 60	0.0000	0.0000

Table 11: Sentence Bucket Analysis (Length Difference)

Length Diff	**mBART**	**mT5**	**NLLB**
< -20	0	1	0
$[-20, -10)$	31	29	25
$[-10, -5)$	115	116	109
-5	39	59	35
-4	65	72	58
-3	75	96	72
-2	129	120	134
-1	171	123	196
0	143	141	150
1	96	88	99
2	55	55	54
3	31	24	30
4	19	15	15
5	9	12	7
$[6, 11)$	5	10	2
$[11, 21)$	2	15	1
≥ 21	3	12	1

Table 12: Sentence Bucket Analysis (Sentence-Level BLEU)

Sent-Level BLEU	**mBART**	**mT5**	**NLLB**
< 10.0	302	350	273
$[10.0, 20.0)$	497	455	508
$[20.0, 30.0)$	118	123	125
$[30.0, 40.0)$	46	40	48
$[40.0, 50.0)$	17	14	20
$[50.0, 60.0)$	4	3	7
$[60.0, 70.0)$	3	0	5
$[70.0, 80.0)$	0	0	1
$[80.0, 90.0)$	1	3	1
≥ 90.0	0	0	0

6 Analysis

6.1 Customized Tokenizers and Pretraining

Although this method seems feasible for low-resource language translations, there also exist some downside of this. Some of the problems include:
1)The new tokens that are being added should be general and high quality but not biased.
2)We have initialized embeddings for the new tokens completely randomly which can be major downside, because to learn these embeddings itself will take a very large training data and computational resourses.So direct fine-tuning may not produce the expected results. Rigorous Pretraining, again in the target language may help in this scenario.

This is the reason that there is a significant drop for the NLLB(b) model compared to the pre-trained model, while again pre-training in the target language has helped Llama2 perform well.

6.2 Fine-Tuning

Fine-tuning the mT5 and mBART model significantly improved performance, particularly in Telugu translations. Despite the limited Telugu data in the pre-trained corpus, the fine-tuning process allowed the model to adapt more closely to the Telugu language, resulting in notable enhancements in translation quality, as evidenced by the significant increase in BLEU score.

6.3 Data Importance

Fine-tuning on the Samanantar dataset may lead to a drop in BLEU scores due to potential biases and suboptimal translations. This effect is noticeable, particularly in the NLLB model, where fine-tuning yields lower scores compared to the raw model. High-quality training data are crucial to optimize model performance during fine-tuning.

7 Conclusion

In this project, we have exploited LLMs like Llama2, mBART50, NLLB and mt5 and tried fine-tuning the models effectively using a small training size of 60000 samples taken from the Samanantar dataset and were able to obtain a decent increase in BLEU score. We have observed repeating the pretraining step can help LLMs to perform better than direct finetuning while using custom tokenizers. We have observed the importance of data that is been used for fine-tuning setup to build better models.

8 Future-Work

For future work, one can anticipate achieving greater performance enhancements by obtaining higher-quality training data and conducting meticulous training sessions with careful hyperparameter tuning.One can aim to boost performance by training on the complete Samanantar dataset (50,00,000 examples) and exploring one-shot or few-shot tuning of Llama3-8b, a chat-based model, to enhance Telugu translations. Additionally, expanding the Telugu tokenizer to include more tokens can augment the number of available tokens, potentially enhancing the model's ability to capture finer linguistic nuances and improve translation accuracy.

One can aim to explore pre-training on Indic language monolingual corpora from IndicCorp and develop a single-script mT5- or mBART-like model for Indic languages. Subsequently, fine-tuning on machine translation tasks using the Samanantar dataset can lead to more effective translation models tailored to Indic languages.And also one can explore various data filtering techniques and data sampling methods to assess their impact on the performance of the Machine Translation (MT) model during training. For instance, one can aim to experiment with sorting data sources by quality and selectively feeding sentences from high-quality sources in later epochs of training to enhance model performance.

References

[1] Gowtham Ramesh, Sumanth Doddapaneni, Aravinth Bheemaraj, Mayank Jobanputra, Raghavan AK, Ajitesh Sharma, Sujit Sahoo, Harshita Diddee, Mahalakshmi J, Divyanshu Kakwani, Navneet Kumar, Aswin Pradeep, Srihari Nagaraj, Kumar Deepak, Vivek Raghavan, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh Shantadevi Khapra. Samanantar: The largest publicly available parallel corpora collection for 11 indic languages, 2021.

[2] NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. No language left behind: Scaling human-centered machine translation, 2022.

[3] Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M. Khapra, and Pratyush Kumar. Indicbart: A pre-trained model for natural language generation of indic languages, 2021.

[4] Abhinand Balachandran. Tamil-llama: A new tamil language model based on llama 2, 2023.

[5] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[6] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation, 2020.

[7] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer, 2021.

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[9] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.