Large Language Model For Telugu

Birudugadda Srivibhav   Pavan Deekshith

Department of Computer Science and Engineering
Project Supervisor: Prof. Mayank Singh
Indian Institute of Technology Gandhinagar, India

1. Introduction

- **Objective:** To develop a comprehensive Telugu Language Model (LLM) using diverse datasets and state-of-the-art language processing techniques.
- **Research Scope:** Involves analyzing existing Telugu LLMs, collecting extensive data from various sources, and implementing advanced data processing methods.
- **Significance:** Aims to enhance the representation and understanding of Telugu in natural language processing tasks, contributing to the broader field of language technology for Indic languages.

2. Need for a Telugu Language Model

- **Tailored for Telugu:** Existing multilingual models often yield poor fertility scores when applied to Telugu due to suboptimal language-specific tuning.
- **Diverse Dataset Requirements:** Current Telugu models rely heavily on limited, domain-specific datasets, hindering their ability to handle diverse linguistic contexts effectively.
- **Versatility Across Applications:** Existing Telugu models are often specialized for specific tasks, limiting their applicability across different domains.

3. Dataset Statistics

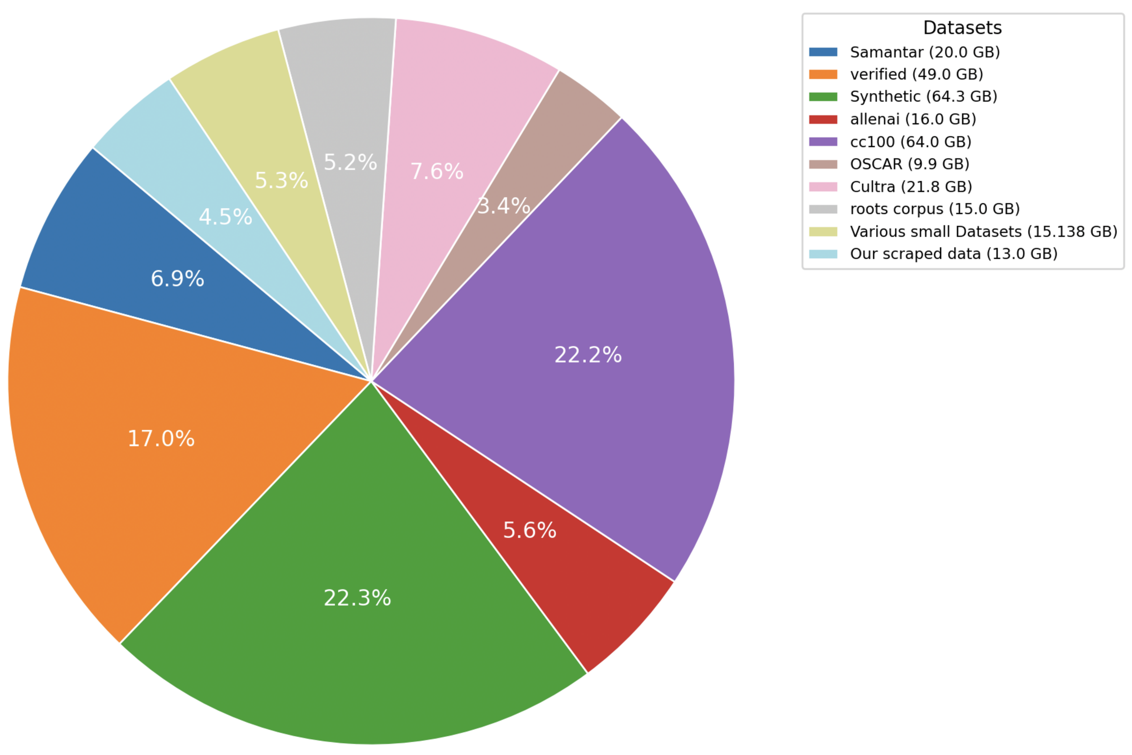

Figure 1: Distribution of dataset sizes before deduplication (288 GB)

4. Methodology

- **Dataset Collection:** Amassed approximately 288 GB of Telugu text data from existing datasets, websites, and PDFs, including popular existing dataset sources like the Roots Corpus, AI4Bharat-IndicNLP, cc-100, mc-4 and OSCAR.
- **Analysis of Existing Models:** Interacted with existing Telugu LLMs, including models like Chandamama Kathalu, Llama-3-8b-Telugu Romanized etc, to understand their capabilities and limitations.
- **Web Scraping and PDF Conversion:** Developed and implemented systems for web scraping and PDF-to-text conversion, enabling the extraction of valuable historical and contemporary Telugu text data.
- **Deduplication of Data:** Applied advanced techniques such as sim-hash and min-hash algorithms for data deduplication.
- **Data Cleaning:** Implemented processes to remove vulgar words, English text, and promotional content from the collected data, enhancing its quality and relevance.
- **Tokenization and Pre-Training:** Successfully tokenized a substantial portion of the dataset and commenced the pre-training phase, drawing inspiration from the Llama architecture.

5. Deduplication Overview

Step 1: Sim-Hash Calculation

- Calculated sim-hashes for all 4.6 crore (46 million) files to generate unique identifiers based on content similarity.
- Distributed sim-hashes across 77,020 CSV files, organizing the dataset for efficient duplicate detection.

Step 2: Exact Duplicate Removal

- Removed exact duplicates by comparing sim-hashes across files and retained one unique instance per CSV file.

Step 3: Near Duplicate Detection

- Utilized the MinHash model from the datasketch library, which generates compact representations (MinHash signatures) of data items.
- Implemented nearest neighbor methods to efficiently detect near duplicates by comparing MinHash signatures, identifying items with high similarity.

6. Deduplication Results

Data Sources	Files before deduplication	Files after deduplication
Datasets	4,49,34,377	3,42,91,959
Websites	5,82,526	5,75,627
PDF's	1,48,418	99,701
Total	**4,56,65,321**	**3,49,67,287**

Figure 2: Files before and after deduplication from various data sources

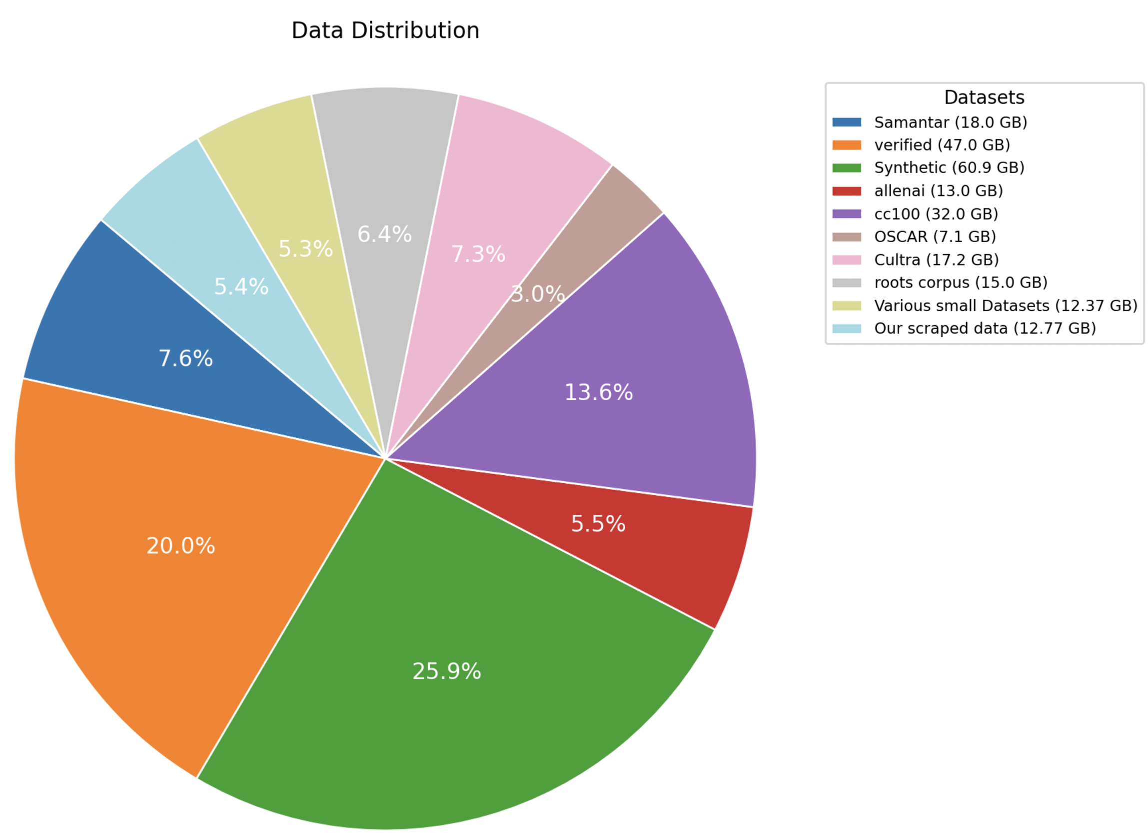

Figure 3: Distribution of dataset sizes after deduplication (235 GB)

7. Tokenizer Overview

Dataset Preparation

- We created five batches for tokeniser training from a 15% (38GB) partition of the deduplicated dataset (total 235GB). The batches are as follows: Batch-(4.1GB), Batch-2 (9.9GB), Batch-3 (15GB), Batch-4 (25GB), and Batch-5 (38GB).
- The first four batches are randomly sampled subsets of this 15% partition, while the fifth batch covers the entire 38GB of data. This approach ensures comprehensive coverage of the language and context.

Subword Segmentation with SentencePiece BPE:

- Utilized SentencePiece BPE Tokenizer from the tokenizers library, combining SentencePiece and Byte-Pair Encoding (BPE) to segment text into subword units to create a 32,768 token vocabulary.
- This approach is especially effective for agglutinative languages like Telugu, providing robust handling of complex word formations and enhancing overall language modeling performance.

8. Fertility Scores

Vocab size - 32768 fixed for all experiments

S.No	1000 Sentences		5380 Sentences	
	Average	Maximum	Average	Maximum
Batch_1(4.1GB)	1.7175	5.066	1.9044	11.22
Batch_2(9.9GB)	2.784	5.5	2.891	12.33
Batch_3(15 GB)	2.784	5.5	2.891	12.33
Batch_4(25 GB)	2.784	5.5	2.891	12.33
Batch_5(38 GB)	2.784	5.5	2.891	12.33

S.N0	Frequency = 5	Frequency = 7	Frequency = 9
Batch_1	1.7175	1.7175	1.7175

Figure 4: Fertility scores for different batch sizes and sentence counts

9. Model Architecture

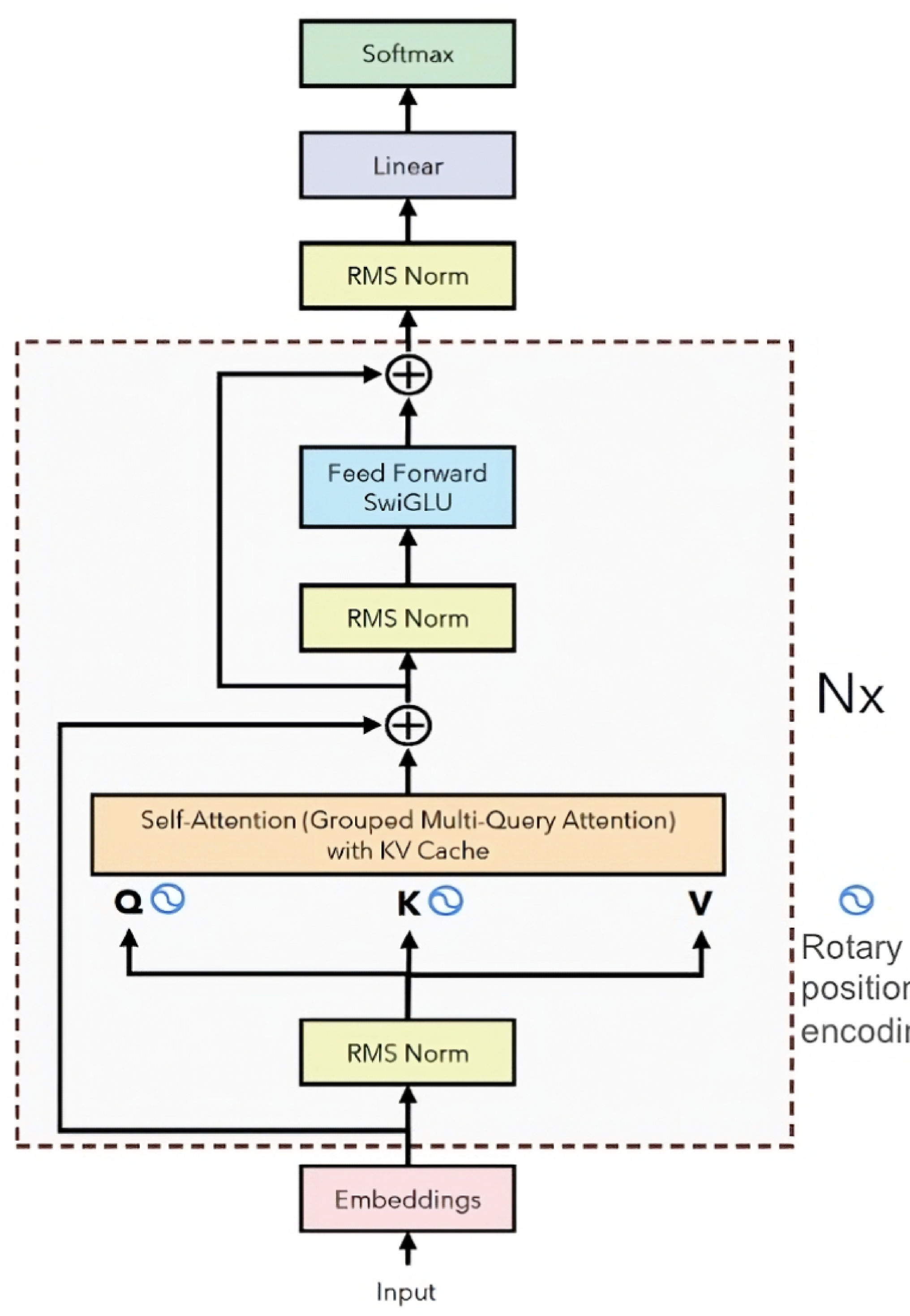

Figure 5: Llama Architecture

10. References

1. R. Ángel, "Dataset deduplication using Spark's MLlib - Towards Data Science," Medium, Dec. 08, 2021. [Online]. Available: https://towardsdatascience.com/deduplication-using-sparks-mllib-4a08f65e5ab9
2. V. Yadav, "Exploring and building the Llama Architecture: A Deep Dive into Components, Coding, and Inference Techniques," Medium, Apr. 25, 2024. [Online]. Available: https://medium.com/@vi.ai_/exploring-and-building-the-llama-3-architecture-a-deep-dive-into-components-coding-and-43d4097cfbbb
3. H. Laurençon et al., "The BigScience ROOTS Corpus: a 1.6TB composite multilingual dataset," arXiv.org, Mar. 07, 2023. [Online]. Available: https://arxiv.org/abs/2303.03915

11. Acknowledgements

We extend our gratitude to:

- Prof. Mayank Singh for his continuous guidance, support, and encouragement throughout the project.
- The staff of the LINGO Labs.