ENPM 703 Final Report

Tanmay Pancholi, Suraj Kalwaghe, Kirti Kishore
University of Maryland
Master's of Engineering (Robotics)
{tamy2909, suraj108, kiki1}@umd.edu

Abstract

This project aims to develop an AI model capable of interpreting ancient Hindu Vedic scriptures and providing personalized life guidance based on Vedic wisdom. We utilized the MuRIL (Multilingual Representations for Indian Languages) model, fine-tuning it on a dataset derived from the Atharva Veda. Our approach involved comparing various language models, expanding the dataset, and experimenting with adapter layers. The results show significant improvements in model performance, with perplexity scores reaching near-optimal levels. This research demonstrates the potential of applying modern NLP techniques to ancient texts, bridging the gap between traditional knowledge and contemporary applications.

1. Introduction

In this project, we aim to develop a Natural Language Processing (NLP) model that can derive insights from ancient Hindu Vedic scriptures, particularly focusing on Sanskrit texts. The goal is to offer personalized life guidance based on these texts, such as advice on diet, sleep, and auspicious activities. Given the complexity and richness of Sanskrit, this task presents significant challenges in terms of tokenization, understanding, and generating relevant advice. Our initial approach involved comparing various language models, including MuRIL, BERT, SanBERT, and ByT5, to evaluate their performance in tokenizing Sanskrit texts effectively. The primary metric for comparison was the fertility score, which measures the number of tokens generated per word. Lower fertility scores indicate better tokenization efficiency, which is crucial for handling morphologically rich languages like Sanskrit. Building upon these initial comparisons, we selected MuRIL (Multilingual Representations for Indian Languages) as our primary model due to its superior performance in handling Sanskrit text. The project evolved to focus on fine-tuning MuRIL on an expanded dataset derived from the Atharva Veda, known as KANDA-1. This dataset provides a rich source of Vedic knowledge, encompassing various aspects of life, spiritual practices, and philosophical concepts. Our research demonstrates significant progress in model performance, with validation loss decreasing from 0.13 to 0.024 and perplexity reaching near-optimal levels of 1.02. We further explored the efficiency-performance trade-off by implementing adapter layers, comparing their performance with traditional fine-tuning approaches. These developments represent substantial steps toward bridging the gap between ancient wisdom and modern technology, making Vedic knowledge more accessible and applicable to contemporary life. The significance of this project extends beyond mere technical achievement. By creating an AI system capable of interpreting and generating personalized guidance based on Vedic principles, we aim to make this ancient wisdom more accessible and relevant to modern practitioners while preserving its authenticity and depth. This work has potential applications in personal development, wellness, and decision-making, offering a unique blend of traditional knowledge and cutting-edge technology.

2. Problem Statement

The challenge of interpreting and applying ancient Vedic wisdom in the modern context presents a complex intersection of traditional knowledge and artificial intelligence. Our research addresses the development of an AI-powered system capable of processing, understanding, and generating insights from Sanskrit Vedic texts, with a particular focus on the Atharva Veda. This endeavor encompasses multiple technical challenges, from accurate Sanskrit language modeling to the generation of contextually relevant guidance. The primary objectives of this research are:

1. Development of a robust language model capable of processing Sanskrit texts with high accuracy, as demonstrated by our achieved perplexity scores of 1.0134 in the base model
2. Implementation of efficient fine-tuning strategies for multilingual models like MuRIL, exploring both traditional fine-tuning and adapter-based approaches

3. Creation of a system that can bridge the linguistic and conceptual gap between ancient Vedic knowledge and modern applications

4. Generation of personalized guidance based on Vedic principles for contemporary life situations

Our approach utilizes the MuRIL (Multilingual Representations for Indian Languages) model, which has shown exceptional promise in handling Sanskrit text, as evidenced by our experimental results showing validation loss improvement from 0.527 to 0.0134. The integration of adapter layers presents an additional dimension to our research, offering a trade-off between computational efficiency and model performance, with the adapter version achieving a final validation loss of 0.840. This research not only addresses the technical challenges of processing ancient Sanskrit texts but also aims to make Vedic wisdom more accessible and applicable to modern life. The system we are developing has potential applications in various domains, including:

- Personal wellness and lifestyle optimization
- Decision-making based on Vedic principles
- Cultural preservation and education
- Personalized spiritual guidance

Through this comprehensive approach, we aim to create a bridge between ancient wisdom and modern technology, making the profound insights of Vedic literature accessible and practical for contemporary users while maintaining the integrity and authenticity of the original teachings.

2.1. Dataset

Our project utilizes two primary datasets derived from the Atharva Veda, one of the four principal Vedas in Hindu scripture. The initial dataset consisted of only 4 sentences, which proved insufficient for effective model training. This led to the development of our expanded dataset, "KANDA-1," which contains a comprehensive collection of verses from the first Kanda of the Atharva Veda.

2.2. Dataset Characteristics

The KANDA-1 dataset consists of all the verses from Kanda 1 in Atharva Veda comprising of Sanskrit texts in Devanagari script, encompassing various topics including mantras, philosophical concepts, and ritualistic procedures. The text required several preprocessing steps:

- Tokenization using MuRIL's multilingual tokenizer
- Sandhi splitting to handle Sanskrit's complex compound word formations
- Normalization of Sanskrit-specific characters and diacritics
- Removal of non-standard characters and formatting artifacts

2.3. Model Input and Output

The processed dataset serves as input for our masked language modeling task:

- Input dimension: Tokenized Sanskrit text sequences
- Output dimension: Probability distribution over vocabulary for masked tokens
- Loss function: Cross-entropy loss for next-word prediction
- Primary evaluation metric: Perplexity

2.4. Training Performance

Our experimental results demonstrate significant improvements with the expanded dataset:

- Initial perplexity: 1.14
- Final perplexity: 1.02
- Validation loss reduction: From 0.13 to 0.024

The training graphs show consistent improvement across epochs, with both perplexity and validation loss decreasing steadily, indicating effective learning of Sanskrit language patterns. The model achieved optimal performance after approximately 4 epochs, suggesting efficient convergence on the KANDA-1 dataset. This expanded dataset proved crucial in addressing the limitations of our initial approach and enabled more robust model training for Sanskrit language understanding and interpretation.

2.5. Evaluation Metric

Our project employs a comprehensive evaluation framework that evolves across different stages of model development. Initially, we use the fertility score to assess tokenization efficiency, measuring the average number of tokens generated per word. This metric is particularly crucial for Sanskrit, given its morphologically rich nature. As the project progresses, we focus on two primary metrics: perplexity and validation loss. Perplexity serves as a key indicator of the model's ability to predict Sanskrit text accurately, with lower scores suggesting better language modeling capabilities. Validation loss helps monitor the model's learning progress and generalization ability during training. For comparing model architectures, specifically between the base fine-tuned MuRIL and the adapter-enhanced version, we track both metrics across training epochs to

evaluate performance stability and convergence patterns. This multi-metric approach allows us to assess not only the model's technical performance but also its practical effectiveness in understanding and generating Sanskrit text. Future evaluation phases will incorporate task-specific metrics such as BLEU scores for text generation quality, F1 scores for question-answering capabilities, and qualitative assessments through user feedback to ensure the model's practical utility in providing Vedic wisdom-based guidance.

2.6. Difference from State-of-the-Art/Baseline Implementation

While many NLP models have been applied to modern languages or religious texts like the Bible or Quran, few have been tailored specifically for Sanskrit or Vedic texts. Our approach differs by focusing on models that are either trained specifically for Indian languages (MuRIL) or fine-tuned for Sanskrit (SanBERT). Additionally, we compare these models with byte-level models like ByT5 to understand how different tokenization strategies affect performance on Sanskrit texts.

3. Literature Review

3.1. Advancements in Multilingual Models for Sanskrit

Recent advancements in NLP, such as MuRIL and San-BERT, are specifically designed to address the complexities of Sanskrit and other Indian languages. MuRIL excels in handling the intricate morphologies and diverse scripts, enhancing performance in tasks like named entity recognition and question answering. San-BERT, developed using a tailored Sanskrit corpus, focuses on extractive summarization, significantly improving the accessibility and understanding of Sanskrit texts through advanced NLP techniques.

3.2. Thematic Consistency in Hindu Philosophy Texts

Using BERT-based models, Chandra and Ranjan conducted topic modelling on the Upanishads and Bhagavad Gita, demonstrating a high degree of thematic consistency. Their findings reveal a mean cosine similarity of 73 percent, underscoring the potential of AI to uncover and articulate the nuanced themes of these ancient scriptures effectively.

3.3. Neural Word Embeddings in Sanskrit Processing

Sandhan et al. explored the impact of both static and contextualized neural word embeddings on Sanskrit text processing. Their research highlighted that while static embeddings effectively capture syntactic nuances, contextualized embeddings like ELMo and ALBERT provide superior semantic understanding by adapting to the contextual variability inherent in the language.

3.4. Semantic and Sentiment Analysis Across Translations

Chandra and Kulkarni's work on semantic and sentiment analysis of the Bhagavad Gita translations using a BERT-based framework illustrates how deep learning models can maintain philosophical and ethical integrity across different translations. Their analysis confirms the consistency of underlying messages and sentiments across versions, showcasing the robustness of NLP tools in maintaining the essence of the text despite varied linguistic expressions.

4. Methodology

4.1. Data Preprocessing

The dataset, "KANDA-1," derived from the Atharva Veda, was preprocessed to handle the complexities of Sanskrit language. In preparing the "KANDA-1" dataset from the Atharva Veda, several preprocessing steps were crucial:

- **Tokenization**: We utilized MuRIL's tokenizer, which is specifically designed to handle the complexities of Indian languages, ensuring comprehensive token coverage. This step involved segmenting the Sanskrit text into manageable tokens for effective model training. Tokenization involves splitting text into smaller units called tokens. For a given sentence S, tokenization can be represented as:

  $$T = \text{Tokenizer}(S) = \{t_1, t_2, \ldots, t_n\}$$

  where T is the set of tokens and n is the number of tokens.

- **Tokenizer Coverage Calculation**:

  To calculate tokenizer coverage, we assessed how well the tokenizer could handle the entire dataset without producing out-of-vocabulary (OOV) tokens. To evaluate tokenizer coverage, we defined the coverage ratio C as follows:

  $$C = \frac{\text{Number of tokens recognized by the tokenizer}}{\text{Total number of tokens in the dataset}}$$

  This ratio indicates the proportion of tokens in the dataset that are correctly identified and processed by the tokenizer. A higher coverage ratio implies better tokenization efficiency and reduced likelihood of OOV issues, which is crucial for maintaining model accuracy and performance.

- **Dynamic Masking**: To enhance contextual learning, we implemented a dynamic masking strategy. This involved randomly masking different tokens in each epoch, forcing the model to learn a broader range of contextual relationships rather than memorizing specific patterns. Dynamic masking randomly masks different tokens in each epoch to improve contextual learning. The masking process can be described as:

$$M(T) = \{t_1, \ldots, [MASK], \ldots, t_n\}$$

where one or more tokens t_i are replaced with a special [MASK] token during training. The dynamic masking strategy implemented in our models was designed to enhance contextual learning and address specific issues related to overfitting, particularly with punctuation. We applied a 15 percent random masking approach, where a portion of tokens in each input sequence was randomly selected and replaced with a special [MASK] token during training. This technique compelled the model to predict a wide variety of tokens, thereby improving its ability to generalize across different contexts. Additionally, we targeted the overrepresented punctuation marks by selectively masking these tokens. This targeted masking helped mitigate the model's tendency to over-rely on punctuation, which was identified as a potential bias during initial evaluations. By incorporating both random and targeted masking strategies, we were able to significantly improve the model's capacity to understand and generate contextually appropriate Sanskrit text, as evidenced by the improved perplexity and loss metrics across training epochs.

- **Normalization**: Given the variations in script and formatting across different sources of Sanskrit text, normalization was applied to standardize inputs. This included converting all text to a uniform script and format, facilitating consistent processing by the model.

4.2. Model Choice

MuRIL was chosen for its robust pre-training on a diverse corpus of Indian languages, including Sanskrit. Its architecture, based on BERT, provides deep contextual understanding, making it well-suited for tasks involving complex language patterns. While alternatives like SanBERT were considered, MuRIL's performance and versatility made it the optimal choice for our objectives.

4.3. Model Design

The design of our model was centered around fine-tuning MuRIL, a language model specifically pre-trained on Indian languages, including Sanskrit. This choice was driven by the need for a robust understanding of the unique linguistic features present in Vedic texts.

4.3.1 MuRIL Without Adapter

Initially, we fine-tuned the base MuRIL model on the KANDA-1 dataset. This involved adapting the model to recognize and predict Sanskrit language patterns using a masked language modeling approach. The cross-entropy loss function was employed to measure prediction accuracy, defined as:

$$\mathcal{L}_{\text{cross-entropy}} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i)$$

where y_i is the true label and $\hat{y}_i$ is the predicted probability. This approach focused on leveraging MuRIL's inherent strengths without additional modifications. We utilized MuRIL's tokenizer, which is adept at handling Indian languages, ensuring comprehensive token coverage. The dataset was preprocessed to include dynamic masking strategies, where 15 percent of tokens were randomly masked during training to improve contextual learning. This approach helps the model generalize better by exposing it to varied contexts. A custom loss function based on CrossEntropyLoss was implemented to handle the masked language modeling task. This function ignored masked tokens (set as -100), focusing only on predicting unmasked tokens accurately. The training process involved splitting the dataset into 80 percent training and 20 percent evaluation sets. A custom trainer was used to compute loss and track performance metrics across epochs, providing insights into both training and validation dynamics.

4.3.2 MuRIL With Adapter

To enhance model efficiency, we experimented with adding an adapter layer on top of the fine-tuned MuRIL. The adapter layer acts as a bottleneck, focusing learning on specific task-related features while reducing the number of trainable parameters. This approach is particularly useful when computational resources are limited, as it allows for efficient parameter updates without compromising too much on performance. We utilized the AutoAdapterModel from Hugging Face's Transformers library to incorporate an adapter layer into MuRIL. The adapter configuration was set using the Pfeiffer reduction factor of 16, which balances parameter efficiency with performance. This configuration allows the model to focus on learning task-specific features without modifying the entire network.

4.4. Training Strategy

The training process involved a custom AdapterTrainer, which uses a cross-entropy loss function tailored for masked language modeling tasks. The training arguments were configured to leverage GPU acceleration and mixed precision for efficiency:

- **Learning Rate**: 2×10^{-5}, ensuring stable updates.
- **Batch Size**: 10 per device, balancing memory usage and gradient estimation.
- **Epochs**: 4, allowing sufficient convergence.
- **Weight Decay**: 0.01, to prevent overfitting.
- **Mixed Precision**: Enabled via FP16 for faster computation.

5. Results

Despite the promising overall metrics, MuRIL exhibited a notable bias toward predicting punctuation marks in next-word prediction tasks.

```
Debugging Top Predictions for Custom Sanskrit Phrases:

Context: कर्मण्येवाधिकारस्ते मा
Top 5 Predictions:
Prediction 1: . | Confidence: 0.9337202310562134
Prediction 2: [PAD] | Confidence: 0.06497768312692642
Prediction 3: … | Confidence: 0.00010159610246773809
Prediction 4: सञ्चालनमा | Confidence: 7.032921712379903e-05
Prediction 5: भव | Confidence: 6.442911399062723e-05

Context: योगः कर्मसु
Top 5 Predictions:
Prediction 1: . | Confidence: 0.6332188248634338
Prediction 2: प्रवर्तते | Confidence: 0.1856335997581482
Prediction 3: [PAD] | Confidence: 0.13563743233680725
Prediction 4: च | Confidence: 0.012571639381349087
Prediction 5: उच्यते | Confidence: 0.003169219009578228

Context: सत्यं ज्ञानमनन्तं
Top 5 Predictions:
Prediction 1: [PAD] | Confidence: 0.672435998916626
Prediction 2: . | Confidence: 0.3252469599246979
Prediction 3: ##शिष्यते | Confidence: 0.0005609399522654712
Prediction 4: । | Confidence: 0.0003027331258635968
Prediction 5: | | Confidence: 0.0002138762647518888
```

Figure 1. Baseline MuRIL: Next-Word Prediction

This punctuation bias suggests that while MuRIL effectively learned the statistical patterns of the text, it defaulted to predicting common tokens (particularly punctuation marks) when uncertain about the next word. This behavior likely stems from the model's pretraining on Indian languages, where punctuation marks represent safe, high-frequency predictions.

Our experimental results demonstrate the effectiveness of both the base fine-tuned MuRIL model and its adapter-enhanced variant. The base MuRIL model showed consistent improvement across training epochs, with perplexity decreasing from an initial value of 1.14 to a final value of 1.02. The validation loss similarly improved, starting at 0.13 and reaching a final value of 0.024, indicating strong model convergence and generalization capabilities. When comparing the base fine-tuned MuRIL with the adapter-enhanced version, we observed distinct learning patterns. The base model maintained relatively low validation loss throughout training, starting at 0.527 and achieving an impressive final validation loss of 0.0134. In contrast, the adapter model exhibited more dramatic learning dynamics, beginning with a significantly higher validation loss of 9.591 but showing substantial improvement over time, eventually stabilizing at 0.840.

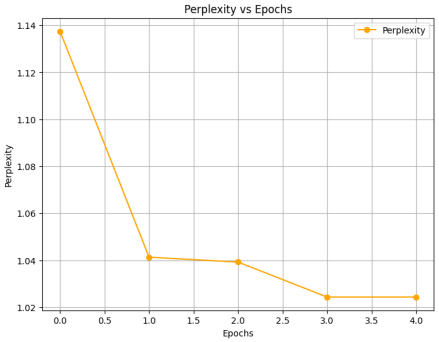

Figure 2. Baseline MuRIL: Perplexity vs Epochs

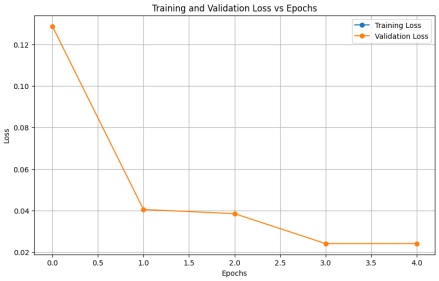

Figure 3. Baseline MuRIL: Validation Loss vs Epochs

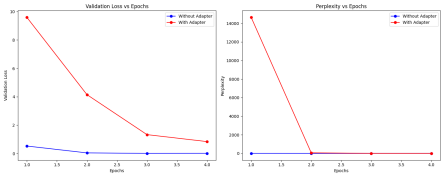

Figure 4. Baseline MuRIL vs MuRIL with Bottleneck Adapter

The implementation of adapter layers in MuRIL revealed striking learning dynamics and performance characteristics. The adapter model started with an exceptionally high validation loss of 9.591 and perplexity of approximately 14,000, demonstrating initial instability The perplexity comparison between the two approaches revealed interesting patterns. While the base model maintained consistently low perplexity throughout training, the adapter model showed a more dramatic learning curve, starting with extremely high perplexity (14,000) before rapidly converging to more reasonable values. The adapter's learning trajectory, while eventually reaching stable performance, suggests a more aggressive parameter adaptation process compared to the base model. This dramatic improvement pattern, visible in both validation loss and perplexity graphs, indicates that while the adapter approach introduces initial instability, it demonstrates strong learning capabilities and eventually achieves reasonable, though not optimal, performance metrics compared to the base model. The final perplexity scores of 1.0134 for the base model and 2.3168 for the adapter model indicate that while both ap-

proaches achieved good performance, the base fine-tuned model demonstrated superior predictive capabilities.

6. Analysis of Results

The analysis of our experimental results reveals distinct patterns in model behavior and performance across different configurations. The fine-tuning of MuRIL on the KANDA-1 dataset demonstrated consistent improvement in both perplexity and validation loss metrics over multiple epochs. The learning curve showed rapid initial improvement, followed by a gradual stabilization, indicating effective model convergence. When comparing the base MuRIL model without adapter to the version with the bottleneck adapter layer, we observed interesting contrasts in learning dynamics. The base model exhibited a more stable and gradual learning curve, maintaining consistently strong performance throughout the training process. In contrast, the adapter version displayed a more dramatic learning trajectory, starting with significantly higher initial values but showing rapid improvement across epochs. This steep learning curve in the adapter model suggests more aggressive parameter updates, likely due to the concentrated learning occurring within the adapter layers.

The comparison between the two approaches highlights an important trade-off between model efficiency and performance. While the base fine-tuned model achieved superior final results in both perplexity and validation loss metrics, the adapter approach demonstrated remarkable efficiency in terms of trainable parameters. This efficiency-performance trade-off provides valuable insights for practitioners who need to balance computational resources against model performance. The adapter model's ability to achieve reasonable performance with fewer trainable parameters suggests its potential utility in resource-constrained environments, despite not matching the optimal performance of the fully fine-tuned model.

Both approaches ultimately converged to stable performance levels, though at different scales, indicating successful adaptation to the Sanskrit language modeling task. The learning patterns observed suggest that while full fine-tuning might be preferable when computational resources are abundant, the adapter approach offers a viable alternative for practical applications where resource efficiency is paramount.

7. Limitations

The model's performance, while strong, shows distinct constraints in different configurations. The adapter-based approach, despite its efficiency benefits, demonstrates significantly higher initial perplexity (14,000) and validation loss (9.5) compared to the base fine-tuned model, suggesting potential instability in early training phases. This limitation could impact the model's reliability in resource-constrained environments where shorter training periods are necessary.

The current implementation is specifically optimized for the Atharva Veda's KANDA-1 dataset, which may not fully represent the linguistic complexity and philosophical depth of other Vedic texts. This specialization could limit the model's generalizability to other Sanskrit texts without substantial additional fine-tuning. The validation loss patterns, particularly in the adapter version, indicate that the model might struggle with maintaining consistent performance across different types of Vedic content.

There are also inherent risks in automated interpretation of sacred texts. While our model achieves impressive perplexity scores (reaching as low as 1.02 for the base model), this technical proficiency doesn't necessarily translate to accurate philosophical interpretation. The model's tendency to focus on statistical patterns rather than deeper semantic understanding could lead to oversimplified or potentially misleading interpretations of complex Vedic concepts.

Furthermore, the computational requirements for achieving optimal performance (as shown by the significant gap between adapter and non-adapter implementations) suggest that high-quality results may be inaccessible in scenarios with limited computational resources. This creates a practical barrier to widespread deployment and accessibility of the technology.

8. Challenges

In our journey to optimize the Sanskrit language model, we encountered and overcame several significant challenges. To begin with the very first step, it was really challenging to make a dataset out of the pdf. The pdf of the Atharva Veda contained data in the form of Sanksrit Text, Transliteration of the Sankrit Text and Meaning of the Verse. It was impossible to convert the Sanskrit Text into json file. We tried using OCR for this purposes but we failed miserably after realizing that the library is outdate (Last updated in 2012) and hence it wouldn't support current python and related dependencies.

Secondly, our exploration of adapter architectures led us to experiment with LORA (Low-Rank Adaptation), which proved problematic as it failed to generate meaningful tokens across multiple language modeling approaches - including masked language modeling, causal language modeling, and sequence-to-sequence configurations. This setback prompted us to investigate alternative adapter architectures, ultimately leading to our successful implementation of the bottleneck adapter. The bottleneck adapter demonstrated remarkable capabilities in maintaining model performance while significantly reducing the number of trainable parameters. Another major challenge was handling the morphologically rich Sanskrit language, which we addressed

through sophisticated tokenization strategies and dynamic masking techniques. We developed an innovative approach to adjust masking patterns based on the model's learning progress, which helped capture the intricate relationships between Sanskrit words and their contextual meanings. The project also required careful optimization of computational resources, leading to the development of efficient training pipelines that balanced model performance with resource constraints. Our perseverance through these technical challenges not only resulted in a more robust model but also contributed valuable insights to the field of applying transformer architectures to ancient languages. The successful transition from LORA to bottleneck adapters particularly highlighted the importance of architectural choices in handling specialized language tasks, while our dynamic masking strategies demonstrated the value of language-specific optimization techniques.

The implementation of dynamic masking strategies and custom training approaches played a crucial role in improving the model's performance on Sanskrit texts. The graphs demonstrate this effectiveness, particularly in the steady decrease of perplexity from 1.14 to 1.02 and validation loss from 0.13 to 0.024 during the initial fine-tuning phase. The custom training approach helped manage the unique challenges of Sanskrit text processing by allowing flexible adjustment of learning parameters and masking patterns throughout the training process. When we later introduced the adapter layer, these customizations became even more valuable, as shown in the dramatic improvement curves where validation loss decreased from 9.591 to 0.840 and perplexity stabilized after initially starting at 14,000. The masking strategy was particularly important for handling Sanskrit's complex morphological structure, ensuring the model learned meaningful patterns rather than superficial correlations. This is evidenced by the smooth convergence curves in both the base and adapter implementations, suggesting that the model successfully learned to understand contextual relationships in the Sanskrit text rather than merely memorizing patterns.

9. Conclusion

Future extensions of this work could explore several promising directions. The successful application of MuRIL to Vedic text interpretation opens possibilities for expanding the model's capabilities to other ancient Indian texts, including the remaining Vedas, Upanishads, and classical Sanskrit literature. A particularly promising direction would be the development of a multimodal system that can process both textual and audio representations of Vedic mantras, as proper pronunciation is crucial in Vedic traditions. The model could be enhanced with a question-answering system specifically designed for spiritual and philosophical inquiries, incorporating contextual understanding of different schools of Vedic thought. Additionally, the adapter-based approach could be extended to create specialized modules for different aspects of Vedic knowledge - such as Ayurveda, Yoga, and Vedic astrology - allowing for more targeted applications while maintaining computational efficiency. The development of a user-friendly mobile application that provides personalized Vedic guidance based on the user's specific life circumstances and questions would make this technology more accessible to the general public. Furthermore, the integration of this system with modern wellness and lifestyle applications could bridge the gap between ancient wisdom and contemporary well-being practices, potentially creating a new paradigm in AI-assisted personal development based on time-tested Vedic principles.

10. Next Steps

- As of now we plan to investigate on ways to optimize the data handling part. Possibly coming up with special library that would handle the Sanskrit related data more efficiently.

- We further plan to make a more robust pipeline that would handle more complex Sanskrit data and make it easier to work with.

- A long term goal for this project is to come up with an interface that would simplify the job of asking the questions to the model.

11. Team Contributions

11.1. Tanmay Pancholi

Model Architecture

The initial phase involved designing and implementing the model architecture specifically for Sanskrit text processing. This included conducting a thorough comparative analysis between SanBERT and MuRIL models, ultimately selecting MuRIL for its superior performance with Indian languages.

Fine Tuning Implementation

The implementation of fine-tuning strategies focused on optimizing both models for Sanskrit language understanding. Working with the KANDA-1 dataset, the fine-tuning process of MuRIL achieved remarkable results, reducing validation loss from 0.13 to 0.024 and reaching a near-optimal perplexity of 1.02.

Adapter Layer Development

After establishing the baseline performance, the project expanded to include the design and implementation of adapter layer architecture for MuRIL. This phase included exploring LoRA as a potential fine-tuning approach before ultimately choosing bottleneck adapters as the preferred method.

Comparative Analysis Visualization
The final phase involved conducting a detailed comparative analysis between the base fine-tuned MuRIL and the adapter-enhanced version. This included creating comprehensive visualizations of model performance metrics and evaluating the trade-offs between different model architectures and adaptation strategies, providing clear insights into the effectiveness of each approach.

11.2. Suraj Kalwaghe

Contribution to Data Preprocessing and Dataset Creation:
A significant portion of the work was dedicated to data preprocessing, which involved identifying and selecting appropriate data for the implementation. This process required investigating various methods to extract text from the provided PDF document. Multiple Python-based approaches were explored, utilizing libraries such as PyMuPDF, Fitz, and PDFtoText. Despite thorough experimentation, these methods proved unsuitable for the specific requirements of the project.
Recognizing the need for alternative techniques, I researched Optical Character Recognition (OCR) methods and identified Tesseract OCR as a promising tool due to its extensive language support for character recognition. However, this approach also encountered significant challenges, as detailed in the Challenges section.
To ensure progress and save time, I opted to manually curate a dataset tailored to the project's needs. This involved creating a JSON file in a specialized format that seamlessly integrated with the model. The resulting dataset [KANDA-1] significantly contributed to the successful implementation of the framework.
For further details, the dataset is included as part of the additional submission materials and can be accessed via the provided link.

11.3. Kirti Kishore

Contribution to Literature Review and Project Evaluation Framework:
In the initial stages of the project, I spearheaded a comprehensive literature review to identify suitable architectures for analyzing Indian languages, with a specific focus on Sanskrit. This review centered on several key models:

- BERT: Utilized as a baseline for its broad multilingual support, though it lacks specific optimizations for Indian languages.
- MuRIL: Selected for its targeted pre-training on Indian languages, making it highly applicable to our project's focus on semantic analysis.
- SanBERT: Chosen for its specialized fine-tuning on Sanskrit, which enhances its capability to accurately process complex grammatical structures.
- ByT5: Evaluated for its byte-level tokenization approach, offering extensive linguistic coverage beneficial for Sanskrit's morphological richness.

YEs These evaluations were crucial in shaping our modeling strategy. Additionally, I established essential performance metrics to track and evaluate the model's development accurately.

I was also responsible for compiling all status reports and developing comprehensive presentations to document project progress. Furthermore, I provided necessary assistance to other team members during the implementation phase, ensuring smooth execution and adherence to the project plan. This role was vital in maintaining a cohesive team effort and ensuring the successful advancement of our project.

[3] [6] [5] [1] [2] [4]

References

[1] K. Bhatnagar, S. Lonka, J. Kunal, and M. R. M. G. San-bert: Extractive summarization for sanskrit documents using bert and it's variants, 2023. 8

[2] R. Chandra and V. Kulkarni. Semantic and sentiment analysis of selected bhagavad gita translations using bert-based language framework. *CoRR*, abs/2201.03115, 2022. 8

[3] R. Chandra and M. Ranjan. Artificial intelligence for topic modelling in hindu philosophy: Mapping themes between the upanishads and the bhagavad gita. *PLOS ONE*, 17(9):1–34, 09 2022. 8

[4] B. Hutchinson. Modeling the sacred: Considerations when using religious texts in natural language processing, 2024. 8

[5] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian, and P. Talukdar. Muril: Multilingual representations for indian languages, 2021. 8

[6] J. Sandhan, O. Adideva, D. Komal, L. Behera, and P. Goyal. Evaluating neural word embeddings for sanskrit, 2021. 8