ShrutiGPT : A Transformer-based Approach For Sanskrit Speech Recognition

A Project Report Submitted to the
Department of Computer Science of
Ramakrishna Mission Vivekananda Educational and Research Institute, Belur,
in partial fulfilment of the requirements for the degree of
MSc in Big Data Analytics.

Submitted by
SAMAPAN KAR
ID No. B2330036

Supervisor:
Swami. Punyeshwarananda
Department of Computer Science
Ramakrishna Mission Vivekananda Educational and Research Institute

Department of Computer Science
Ramakrishna Mission Vivekananda Educational and Research Institute
Belur Math, Howrah 711202, West Bengal, India
December 24, 2024

ShrutiGPT : A Transformer-based Approach For Sanskrit Speech Recognition

By

SAMAPAN KAR

Declaration by student:

“I hereby declare that the present dissertation is the outcome of my project work under the guidance of Swami Punyeshwarananda and I have properly acknowledged the sources of materials used in my project report."

____

(Samapan Kar, ID No. B2330036)

A project report in the partial fulfilment of the requirements of the degree of MSc in Big Data Analytics

Examined and approved on

____

by

____

Swami Punyeshwarananda (supervisor)
Department of Computer Science
Ramakrishna Mission Vivekananda Educational and Research Institute

Countersigned by

____

Registrar
Ramakrishna Mission Vivekananda Educational and Research Institute

Department of Computer Science
Ramakrishna Mission Vivekananda Educational and Research Institute
Belur Math, Howrah 711202, West Bengal, India

December 24, 2024

Acknowledgement

The present project work is submitted in partial fulfilment of the requirements for the degree of Master of Science of Ramakrishna Mission Vivekananda University (RKMVU). I express my deepest gratitude to my supervisor Prof. Swami Punyeshwarananda of Ramakrishna Mission Vivekananda Educational and Research Institute for his inestimable support, encouragement, profound knowledge, largely helpful conversations and also for providing me a systematic way for the completion of my project work. His ability to work hard inspired me a lot. I am also extremely grateful to the Vice-Chancellor of this University for his encouragement and support throughout the course. Last but not the least, this work would not have been possible without support of my fellow classmates.

Belur
December 24, 2024

(Samapan Kar)
Department of Computer Science
Ramakrishna Mission Vivekananda Educational and Research Institute

Contents

List of Figures

Chapter 1

Introduction

Sanskrit, one of the most ancient and significant languages in the world, is deeply rooted in India's cultural and intellectual heritage. It is the language of some of the most profound philosophical, scientific, and literary works ever created, making its preservation and accessibility a matter of cultural importance. Despite its historical significance, Sanskrit has seen limited integration into modern technological advancements, especially in the domain of speech processing. The development of Automatic Speech Recognition (ASR) systems for Sanskrit is hindered by several unique challenges that set it apart from widely spoken languages.

One of the primary challenges lies in Sanskrit's phonetic complexity. The language boasts a rich and intricate phonetic structure, characterized by unique sounds such as aspirated and retroflex phonemes, as well as subtle distinctions between vowels and consonants. Capturing these nuances in speech data requires sophisticated models capable of discerning fine-grained acoustic features. Moreover, Sanskrit's grammar is highly inflectional, governed by extensive rules of morphology and syntax, including sandhi (phonetic changes at word boundaries) and compounds. These features pose additional obstacles for transcription accuracy, especially for long and grammatically dense sentences. Compounding these challenges is the scarcity of labeled Sanskrit speech datasets, which limits the ability of traditional machine learning approaches to achieve high accuracy.

The advent of transformer architectures in deep learning has revolutionized natural language processing and speech recognition. Unlike traditional sequence models such as RNNs or LSTMs, transformers utilize self-attention mechanisms to process input sequences in parallel, effectively capturing long-range dependencies and contextual relationships. These capabilities make transformers particularly suited to handling the phonetic and grammatical complexities of Sanskrit. Moreover, their flexibility allows them to be integrated with spectrogram-based audio representations, further enhancing their ability to process rich and nuanced speech data.

Recognizing these potentials, this project introduces ShrutiGPT, a transformer-based ASR system tailored for Sanskrit speech recognition. ShrutiGPT leverages the strengths of transformers to address Sanskrit's unique challenges, combining advanced spectrogram analysis with innovative transcription strategies to improve accuracy. The system explores both Devanagari and SLP1 transcription methods, aiming to strike a balance between phonetic fidelity and computational efficiency. By bridging the gap between Sanskrit's oral tradition and its written form, ShrutiGPT not only addresses the immediate problem of transcription but also contributes to the broader goal of preserving and modernizing Sanskrit for the digital age.

This project represents a step towards making Sanskrit more accessible and engaging for scholars, educators, and learners. It aims to demonstrate how cutting-edge AI techniques can be applied to ancient languages, ensuring their relevance in contemporary technological landscapes. ShrutiGPT is not just a tool for transcription—it is a bridge between tradition and innovation.

Chapter 2

Literature Survey

1. **Vaswani, A. et al. "Attention is All You Need." (2017):**[4]

   This seminal paper introduced the Transformer architecture, a novel model for sequence transduction tasks that relies entirely on self-attention mechanisms, eliminating the need for recurrent or convolutional layers. The Transformer demonstrated superior performance in machine translation tasks, achieving a BLEU score of 28.4 on the WMT 2014 English-to-German dataset and 41.8 on the English-to-French dataset, setting new benchmarks at the time. The architecture's ability to model long-range dependencies and its parallelizable structure have made it a foundation for numerous advancements in natural language processing. In the context of the ShrutiGPT project, the Transformer architecture's strengths are leveraged to address the complexities of Sanskrit speech recognition. Its self-attention mechanism is particularly beneficial for capturing the intricate phonetic and syntactic patterns inherent in Sanskrit, enabling more accurate transcription of spoken language into text.

2. **OpenAI's Whisper: A Transformer-based ASR Model:**[3]

   Whisper, developed by Alec Radford et al. at OpenAI, is a pre-trained transformer-based ASR model designed for multilingual speech recognition. Unlike models like Wav2Vec 2.0, which are pre-trained on 60,000 hours of unlabeled audio, Whisper uses 680,000 hours of labeled audio-transcription data, including 117,000 hours of multilingual data. This extensive dataset allows Whisper to support over 96 languages, many of which are low-resource, and to learn a speech-to-text mapping directly during pre-training, requiring minimal fine-tuning. Whisper achieves competitive performance, with a near 3% word error rate (WER) on the LibriSpeech test-clean subset and 4.7% WER on TED-LIUM datasets. Its transformer encoder-decoder architecture processes audio spectrograms to predict text tokens accurately, making it highly effective across diverse linguistic contexts. The ShrutiGPT project leverages Whisper's methodology and architecture to tackle the phonetic and grammatical complexities of Sanskrit. By adapting this transformer-based approach, ShrutiGPT aims to achieve high accuracy in Sanskrit ASR, inspired by Whisper's success with low-resource languages.

3. **End-to-End Speech Recognition for Sanskrit using Self-Supervised Learning:**[2]

   This research paper addresses the challenge of developing an ASR system for Sanskrit, a language with limited speech data resources. The authors propose an end-to-end speech recognition model utilizing self-supervised learning to overcome data scarcity. Their methodology involves pre-training a model on a large corpus of unlabeled speech data to learn general acoustic representations, followed by fine-tuning on a smaller, labeled Sanskrit dataset. The results indicate that this approach significantly improves recognition accuracy for Sanskrit, demonstrating the effectiveness of self-supervised learning in low-resource settings. The ShrutiGPT project aligns with this research by adopting similar strategies to mitigate data limitations in Sanskrit ASR. By incorporating self-supervised learning techniques, ShrutiGPT aims to enhance its model's performance, ensuring accurate and reliable transcription of Sanskrit speech despite the paucity of extensive labeled datasets.

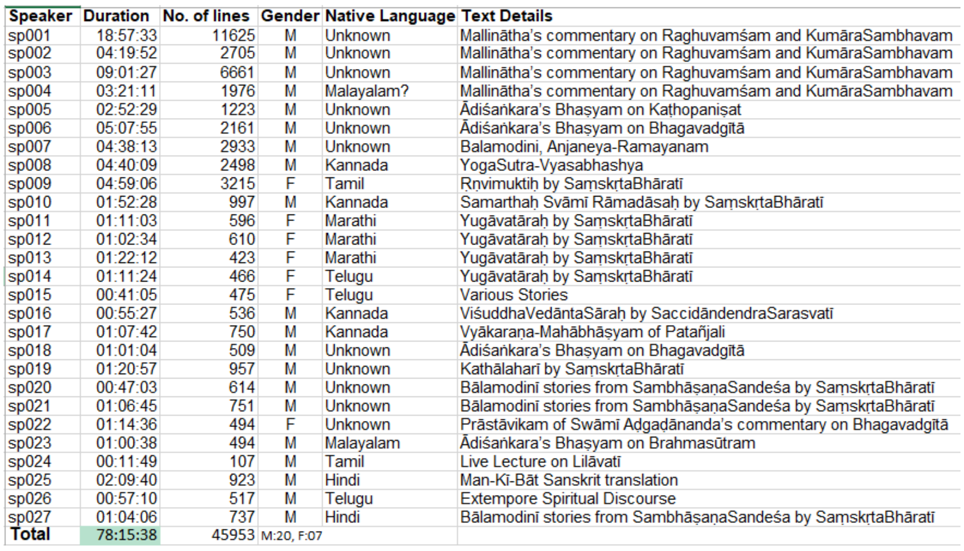
   This Sanskrit ASR system was trained using the Vaksancaya dataset, achieving a validation WER of 12.5% after 35 hours of training. With the integration of a language model, the WER improved to 2.4% on the train set and 5.1% on the test set. Compared to previous approaches, the proposed self-supervised method outperformed others, including the unsupervised approach with a WER of 9.89% and the CTC with spectrogram augmentation approach with a WER of 7.64%.

Chapter 3

Proposed Algorithm

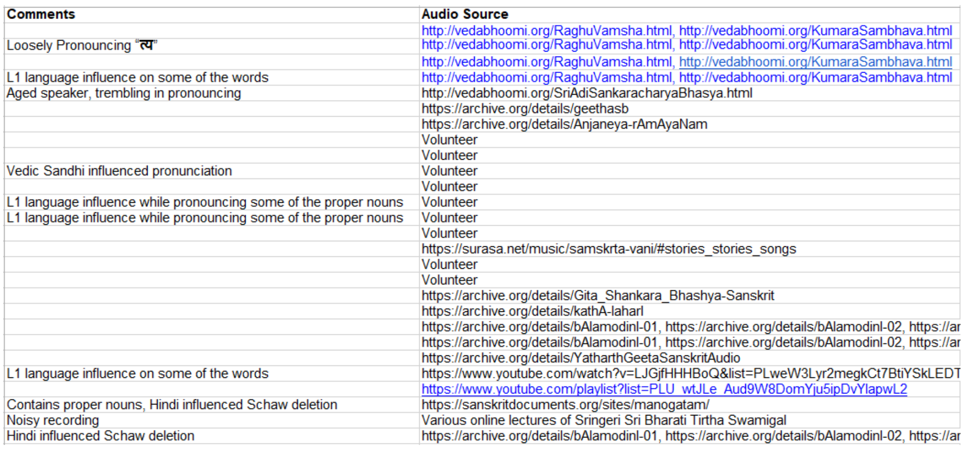
3.1 Data description

- Corpus **Vāksañcayaḥ** [1], has more than 78 hours of data with an overall vocabulary size of 91,000 words and recordings of about 46,000 sentences. Each with a sampling rate of 22 KHz.
- The contents span over 3 time periods categorised into pre-classical literature (1,500 BCE - 100 BCE), classical literature (300 CE - 800 CE) and modern literature (900 CE to now).
- The corpus is intended to address the challenges in interfacing the speech and the text covered by the disciplines of phonetics (śikṣā), and grammar (vyākaraṇa).
- In the Sanskrit literature, frequency of commonly used words changes from one topical domain to another, specifically one Śāstra (branch of learning) to another.
- The corpus encompasses a diverse range of domains, including classical works like philosophy, literature, poetry, and grammar, as well as contemporary recordings such as stories, live lectures, spiritual discourses, and podcasts, ensuring a comprehensive representation of Sanskrit vocabulary.



Speaker	Duration	No. of lines	Gender	Native Language	Text Details
sp001	18:57:33	11625	M	Unknown	Mallinātha's commentary on Raghuvamśam and KumāraSambhavam
sp002	04:19:52	2705	M	Unknown	Mallinātha's commentary on Raghuvamśam and KumāraSambhavam
sp003	09:01:27	6661	M	Unknown	Mallinātha's commentary on Raghuvamśam and KumāraSambhavam
sp004	03:21:11	1976	M	Malayalam?	Mallinātha's commentary on Raghuvamśam and KumāraSambhavam
sp005	02:52:29	1223	M	Unknown	Ādiśaṅkara's Bhaṣyam on Kaṭhopaniṣat
sp006	05:07:55	2161	M	Unknown	Ādiśaṅkara's Bhaṣyam on Bhagavadgītā
sp007	04:38:13	2933	M	Unknown	Balamodini, Anjaneya-Ramayanam
sp008	04:40:09	2498	M	Kannada	YogaSutra-Vyasabhashya
sp009	04:59:06	3215	F	Tamil	Ṛṇvimuktiḥ by SaṃskṛtaBhāratī
sp010	01:52:28	997	M	Kannada	Samarthaḥ Svāmī Rāmadāsaḥ by SaṃskṛtaBhāratī
sp011	01:11:03	596	F	Marathi	Yugāvatāraḥ by SaṃskṛtaBhāratī
sp012	01:02:34	610	F	Marathi	Yugāvatāraḥ by SaṃskṛtaBhāratī
sp013	01:22:12	423	F	Marathi	Yugāvatāraḥ by SaṃskṛtaBhāratī
sp014	01:11:24	466	F	Telugu	Yugāvatāraḥ by SaṃskṛtaBhāratī
sp015	00:41:05	475	F	Telugu	Various Stories
sp016	00:55:27	536	M	Kannada	ViśuddhaVedāntaSāraḥ by SaccidāndendraSarasvatī
sp017	01:07:42	750	M	Kannada	Vyākaraṇa-Mahābhāṣyam of Patañjali
sp018	01:01:04	509	M	Unknown	Ādiśaṅkara's Bhaṣyam on Bhagavadgītā
sp019	01:20:57	957	M	Unknown	Kathālaharī by SaṃskṛtaBhāratī
sp020	00:47:03	614	M	Unknown	Bālamodinī stories from SambhāṣaṇaSandeśa by SaṃskṛtaBhāratī
sp021	01:06:45	751	M	Unknown	Bālamodinī stories from SambhāṣaṇaSandeśa by SaṃskṛtaBhāratī
sp022	01:14:36	494	F	Unknown	Prāstāvikam of Swāmī Aḍgaḍānanda's commentary on Bhagavadgītā
sp023	01:00:38	494	M	Malayalam	Ādiśaṅkara's Bhaṣyam on Brahmasūtram
sp024	00:11:49	107	M	Tamil	Live Lecture on Līlāvatī
sp025	02:09:40	923	M	Hindi	Man-Kī-Bāt Sanskrit translation
sp026	00:57:10	517	M	Telugu	Extempore Spiritual Discourse
sp027	01:04:06	737	M	Hindi	Bālamodinī stories from SambhāṣaṇaSandeśa by SaṃskṛtaBhāratī
Total	78:15:38	45953	M:20, F:07		

Figure 3.1: Audio details



Comments	Audio Source
	http://vedabhoomi.org/RaghuVamsha.html, http://vedabhoomi.org/KumaraSambhava.html
Loosely Pronouncing "त्य"	http://vedabhoomi.org/RaghuVamsha.html, http://vedabhoomi.org/KumaraSambhava.html
	http://vedabhoomi.org/RaghuVamsha.html, http://vedabhoomi.org/KumaraSambhava.html
L1 language influence on some of the words	http://vedabhoomi.org/RaghuVamsha.html, http://vedabhoomi.org/KumaraSambhava.html
Aged speaker, trembling in pronouncing	http://vedabhoomi.org/SriAdiSankaracharyaBhasya.html
	https://archive.org/details/geethasb
	https://archive.org/details/Anjaneya-rAmAyaNam
	Volunteer
	Volunteer
Vedic Sandhi influenced pronunciation	Volunteer
	Volunteer
L1 language influence while pronouncing some of the proper nouns	Volunteer
L1 language influence while pronouncing some of the proper nouns	Volunteer
	Volunteer
	https://surasa.net/music/samskrta-vani/#stories_stories_songs
	Volunteer
	Volunteer
	https://archive.org/details/Gita_Shankara_Bhashya-Sanskrit
	https://archive.org/details/kathA-laharI
	https://archive.org/details/bAlamodinI-01, https://archive.org/details/bAlamodinI-02, https://ar
	https://archive.org/details/bAlamodinI-01, https://archive.org/details/bAlamodinI-02, https://ar
	https://archive.org/details/YatharthGeetaSanskritAudio
L1 language influence on some of the words	https://www.youtube.com/watch?v=LJGjfHHHBoQ&list=PLweW3Lyr2megkCt7BtiYSkLEDT
	https://www.youtube.com/playlist?list=PLU_wtJLe_Aud9W8DomYju5ipDvYlapwL2
Contains proper nouns, Hindi influenced Schaw deletion	https://sanskritdocuments.org/sites/manogatam/
Noisy recording	Various online lectures of Sringeri Sri Bharati Tirtha Swamigal
Hindi influenced Schaw deletion	https://archive.org/details/bAlamodinI-01, https://archive.org/details/bAlamodinI-02, https://ar

Figure 3.2: Audio sources

3.2 Methodology

3.2.1 Dataset Preparation

Given the scarcity of large-scale annotated Sanskrit datasets, data augmentation plays a crucial role in this project. Transformer architectures, like the one used in ShrutiGPT, require extensive amounts of data to effectively learn the complex patterns in speech signals. However, Sanskrit, being a low-resource language, lacks the abundance of labeled speech data available for more commonly spoken languages. This limitation necessitates the use of augmentation to artificially expand the dataset and improve model performance.

Data Augmentation Techniques:

To simulate a wide variety of acoustic conditions and ensure better generalization, several augmentation processes are applied to the original dataset:

- **Noise Addition:** Background noise is added to mimic real-world environments like crowded areas or open spaces.
- **Time Stretching:** The audio speed is slightly varied to account for natural fluctuations in speech tempo.
- **Pitch Shifting:** The pitch is adjusted to simulate variations in speaker vocal tones.

Extent of Application:

Each augmentation technique is applied at a controlled scale to maintain a balance between dataset diversity and the integrity of the original data. For example:

- Noise levels are kept subtle to ensure the speech remains intelligible.
- Time stretching and pitch shifting are limited to $\pm 5\%$ variations to retain the naturalness of the speech.
- Approximately 2-3 augmented versions are generated per original sample, tripling the dataset size and providing sufficient variety for the transformer model to train effectively.

By incorporating these augmentation strategies, the dataset becomes robust to noise, speaker variations, and environmental conditions, significantly enhancing the model's ability to generalize across diverse speech inputs.

3.2.2 Feature extraction

Feature extraction is a critical step in speech recognition as it transforms raw audio signals into a structured representation that models can process efficiently. Transformers, which are designed for sequence-to-sequence tasks, require input in a form that highlights relevant features, such as phonetic and temporal patterns, while discarding irrelevant noise or redundancy in the raw waveform.

Why Spectrogram Analysis?

A spectrogram provides a visual representation of the frequency content of an audio signal over time, making it an ideal input for speech processing. By capturing both frequency and temporal information, spectrograms allow the model to understand how phonemes evolve over time, which is essential for recognizing words and sentences in speech.

Why Use Log-Mel Spectrogram Instead of Vanilla Spectrogram?

- **Human Perception Alignment:** The Mel scale is designed to mimic the human auditory system's perception of sound frequencies, giving more weight to lower frequencies where humans discern speech more effectively.
- **Dynamic Range Management:** By applying a logarithmic transformation, the log-Mel spectrogram compresses the wide dynamic range of audio signals, making the input more suitable for models to process and reducing the sensitivity to variations in loudness.
- **Dimensionality Reduction:** Compared to a vanilla spectrogram, the Mel spectrogram reduces the number of frequency bins, simplifying the input without losing essential speech information.

Process

In ShrutiGPT, the raw audio waveform is converted into a log-Mel spectrogram using the following steps:

1. **Frame Division:** The audio signal is divided into small overlapping frames.
2. **Fourier Transform:** Each frame undergoes a Fourier Transform to calculate the frequency components, forming the vanilla spectrogram.
3. **Mel Filter Bank Application:** The spectrogram is then passed through a Mel filter bank, which maps the frequencies to the Mel scale.
4. **Logarithmic Transformation:** Finally, the Mel-scaled spectrogram is converted to a log scale to compress the dynamic range and emphasize smaller amplitude variations.

By using log-Mel spectrograms, ShrutiGPT ensures that the model focuses on the most critical aspects of the audio input, aligning closely with human auditory characteristics and enabling more accurate recognition of Sanskrit speech.

3.2.3 Transformer Model Architecture

The Transformer model is designed to convert audio spectrograms into character sequences. Below is the detailed architecture:

Components of the Transformer

1. **Encoder**

   The encoder processes the input spectrogram features and produces context-aware embeddings. It consists of multiple identical layers with the following components:

   - **Multi-Head Attention:**
     - Uses 8 attention heads to compute self-attention scores, allowing the model to focus on relevant parts of the input.
   - **Feed-Forward Neural Network:**
     - Contains two fully connected layers:

        * The first layer has 512 neurons with ReLU activation.
        * The second layer maps back to the 128-dimensional embedding space.

  - **Residual Connections and Layer Normalization:**
    - Residual connections are added after each sub-layer, followed by layer normalization to stabilize training.
  - **Dropout:** A 30% dropout is applied to the outputs of the attention and feed-forward layers.

  Output: Encoded representations with the same length as the input sequence and a feature size of 128.

2. **Decoder**

   The decoder generates the target character sequence based on the encoder outputs and previous target tokens. It consists of multiple identical layers with the following components:

   - **Causal (Masked) Multi-Head Attention:**
     - Prevents the decoder from attending to future tokens during training by masking out future positions.
   - **Cross-Attention:**
     - Enables the decoder to focus on relevant parts of the encoder's output for each step in the target sequence.
   - **Feed-Forward Neural Network:**
     - Similar to the encoder, with two fully connected layers: 512 neurons in the first layer and 128 neurons in the second layer.
   - **Residual Connections and Layer Normalization:**
     - Each sub-layer output is combined with the input via residual connections, followed by layer normalization.
   - **Dropout:** A 30% dropout is applied at all stages.

   Output: Processed embeddings with the same length as the target sequence and a feature size of 128.

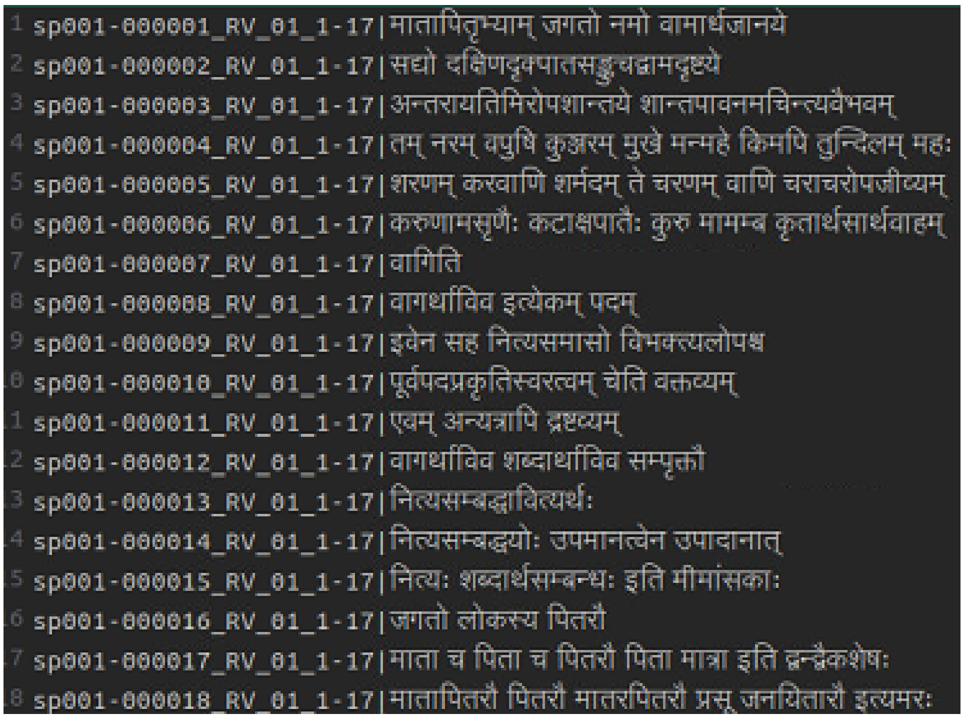
3. **Embedding Layers**

   - **Input Embeddings:**
     - Converts input tokens into dense 128-dimensional vectors.
     - Adds positional encodings to incorporate sequence order information.
   - **Output Embeddings:**
     - A final dense layer maps the decoder's output to probabilities over the vocabulary.

3.2.4 Transcript and Tokenization

Devanagari Script

Devanagari is a phonetic script used for Sanskrit and several Indian languages. While it accurately represents Sanskrit sounds, its complex Unicode structure and large vocabulary size make it challenging for machine learning models due to increased computational overhead and sequence decoding complexity.

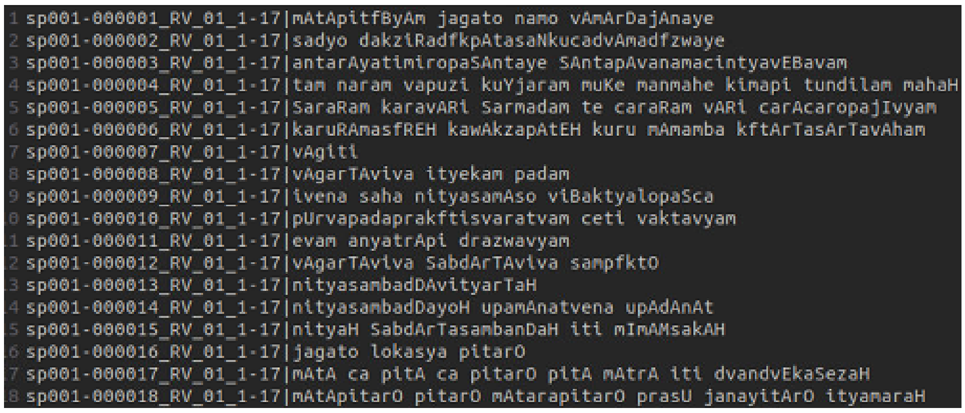


```
sp001-000001_RV_01_1-17|मातापितृभ्याम् जगतो नमो वामार्धजानये
sp001-000002_RV_01_1-17|सद्यो दक्षिणदृक्पातसङ्कुचद्वामदृष्टये
sp001-000003_RV_01_1-17|अन्तरायतिमिरोपशान्तये शान्तपावनमचिन्त्यवैभवम्
sp001-000004_RV_01_1-17|तम् नरम् वपुषि कुञ्जरम् मुखे मन्महे किमपि तुन्दिलम् महः
sp001-000005_RV_01_1-17|शरणम् करवाणि शर्मदम् ते चरणम् वाणि चराचरोपजीव्यम्
sp001-000006_RV_01_1-17|करुणामसृणैः कटाक्षपातैः कुरु मामम्ब कृतार्थसार्थवाहम्
sp001-000007_RV_01_1-17|वागिति
sp001-000008_RV_01_1-17|वागर्थाविव इत्येकम् पदम्
sp001-000009_RV_01_1-17|इवेन सह नित्यसमासो विभक्त्यलोपश्च
sp001-000010_RV_01_1-17|पूर्वपदप्रकृतिस्वरत्वम् चेति वक्तव्यम्
sp001-000011_RV_01_1-17|एवम् अन्यत्रापि द्रष्टव्यम्
sp001-000012_RV_01_1-17|वागर्थाविव शब्दार्थाविव सम्पृक्तौ
sp001-000013_RV_01_1-17|नित्यसम्बद्धावित्यर्थः
sp001-000014_RV_01_1-17|नित्यसम्बद्धयोः उपमानत्वेन उपादानात्
sp001-000015_RV_01_1-17|नित्यः शब्दार्थसम्बन्धः इति मीमांसकाः
sp001-000016_RV_01_1-17|जगतो लोकस्य पितरौ
sp001-000017_RV_01_1-17|माता च पिता च पितरौ पिता मात्रा इति द्वन्द्वैकशेषः
sp001-000018_RV_01_1-17|मातापितरौ पितरौ मातरपितरौ प्रसू जनयितारौ इत्यमरः
```

Figure 3.3: Devanagari transcripts

SLP1 (Sanskrit Library Phonetic Basic)

SLP1 is a transliteration scheme for Sanskrit that uses ASCII characters to represent phonemes, offering a compact and efficient way to process Sanskrit text.



```
sp001-000001_RV_01_1-17|mAtApitfByAm jagato namo vAmArDajAnaye
sp001-000002_RV_01_1-17|sadyo dakziRadfkpAtasaNkucadvAmadfzwaye
sp001-000003_RV_01_1-17|antarAyatimiropaSAntaye SAntapAvanamacintyavEBavam
sp001-000004_RV_01_1-17|tam naram vapuzi kuYjaram muKe manmahe kimapi tundilam mahaH
sp001-000005_RV_01_1-17|SaraRam karavARi Sarmadam te caraRam vARi carAcaropajIvyam
sp001-000006_RV_01_1-17|karuRAmasfREH kawAkzapAtEH kuru mAmamba kftArTasArTavAham
sp001-000007_RV_01_1-17|vAgiti
sp001-000008_RV_01_1-17|vAgarTAviva ityekam padam
sp001-000009_RV_01_1-17|ivena saha nityasamAso viBaktyalopaSca
sp001-000010_RV_01_1-17|pUrvapadaprakftisvaratvam ceti vaktavyam
sp001-000011_RV_01_1-17|evam anyatrApi drazwavyam
sp001-000012_RV_01_1-17|vAgarTAviva SabdArTAviva sampfktO
sp001-000013_RV_01_1-17|nityasambadDAvityarTaH
sp001-000014_RV_01_1-17|nityasambadDayoH upamAnatvena upAdAnAt
sp001-000015_RV_01_1-17|nityaH SabdArTasambanDaH iti mImAMsakAH
sp001-000016_RV_01_1-17|jagato lokasya pitarO
sp001-000017_RV_01_1-17|mAtA ca pitA ca pitarO pitA mAtrA iti dvandvEkaSezaH
sp001-000018_RV_01_1-17|mAtApitarO pitarO mAtarapitarO prasU janayitArO ityamaraH
```

Figure 3.4: SLP1 transcripts

Advantages of SLP1 over Devanagari

- Reduces the size of the token set, simplifying tokenization. Speeds up training by reducing complexity.
- Uses single-byte ASCII characters, avoiding Unicode complexities. Improves sequence decoding accuracy, leading to fewer errors.
- Simplifies training and inference, making it faster and more resource-efficient. Simplifies post-processing, as SLP1 can be easily converted back to Devanagari for user-friendly output.

In summary, SLP1 provides a lightweight, precise, and computationally efficient alternative to Devanagari for Sanskrit speech recognition in ShrutiGPT.

3.2.5 Training Pipeline

The training pipeline for ShrutiGPT is designed to ensure effective model optimization, stability during training, and robustness to various input conditions. Below are the key components of the training pipeline:

1. **Loss Function**

The primary objective during training is to minimize the difference between the model's predicted output and the ground truth. This is achieved through:

- **Cross-Entropy Loss:** Used for the token-level predictions, it ensures that the model effectively learns to map input spectrograms to the correct sequence of tokens. Each predicted token's probability is compared with the true token, and the loss is accumulated over the entire output sequence.
- **Regularization Terms:** Additional penalties, such as L2 regularization, are included to prevent overfitting by discouraging overly complex models. Regularization also stabilizes the training process, ensuring better generalization to unseen data.

2. **Optimizer and Learning Rate Scheduler**

   To effectively navigate the loss landscape, the pipeline employs a carefully chosen optimizer and learning rate strategy:

   - **Adam Optimizer:** Adam combines the benefits of momentum-based gradient descent and adaptive learning rates. It dynamically adjusts the learning rate for each parameter, ensuring fast convergence while mitigating oscillations in the loss.
   - **Learning Rate Warm-Up:** At the beginning of training, a small learning rate is used and gradually increased. This prevents large updates to model weights, which can destabilize training in early stages.
   - **Cosine Decay Scheduler:** After the warm-up phase, the learning rate gradually decreases using a cosine decay schedule. This helps the model converge more effectively by making smaller adjustments as it approaches the optimal parameters.

3. **Data Augmentation**

   To improve the model's robustness and generalization capabilities, audio data augmentation is employed. By incorporating these augmented samples into each training batch, the model gains resilience to noise, variability, and unseen accents, ultimately improving its real-world performance.

4. **Batch Organization and Token Constraints**

   To ensure consistency and efficiency:

- Batches are organized to contain spectrograms and tokenized transcripts of similar lengths, optimizing GPU memory usage.
- Transcripts are truncated or padded to predefined maximum lengths, simplifying sequence handling during training.

This structured approach ensures smooth training and avoids computational bottlenecks.

3.2.6 Evaluation Metrics

Evaluating the performance of a speech recognition model requires comprehensive metrics that capture various aspects of accuracy. The following metrics are used:

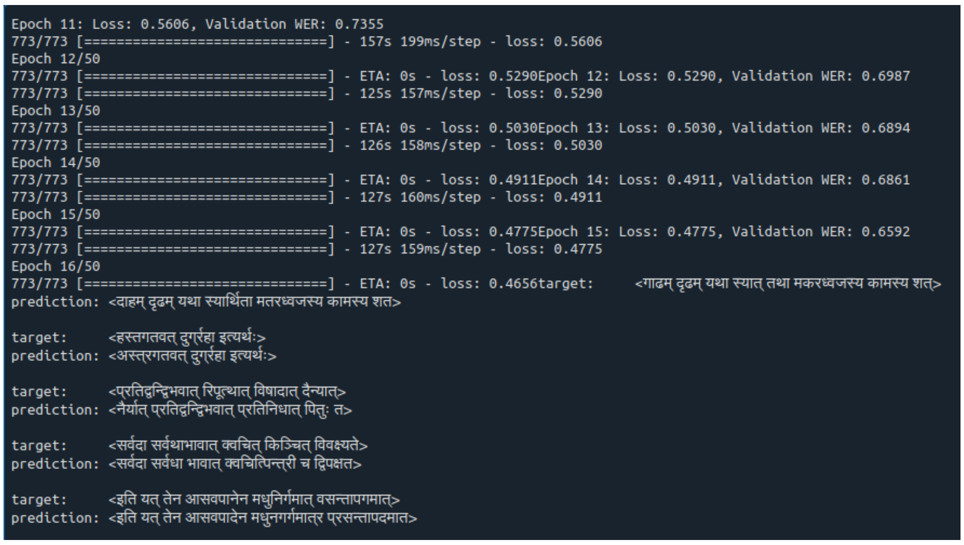
- **Word Error Rate (WER):** WER measures the percentage of words that are incorrectly predicted by the model. It considers insertions, deletions, and substitutions, providing a clear picture of transcription accuracy at the word level. A lower WER indicates better model performance and improved usability for real-world scenarios.

Chapter 4

Results

4.1 Devanagari

The ShrutiGPT model demonstrated significant progress throughout the training process, with both the training loss and validation Word Error Rate (WER) showing consistent improvements.



```
Epoch 11: Loss: 0.5606, Validation WER: 0.7355
773/773 [==============================] - 157s 199ms/step - loss: 0.5606
Epoch 12/50
773/773 [==============================] - ETA: 0s - loss: 0.5290Epoch 12: Loss: 0.5290, Validation WER: 0.6987
773/773 [==============================] - 125s 157ms/step - loss: 0.5290
Epoch 13/50
773/773 [==============================] - ETA: 0s - loss: 0.5030Epoch 13: Loss: 0.5030, Validation WER: 0.6894
773/773 [==============================] - 126s 158ms/step - loss: 0.5030
Epoch 14/50
773/773 [==============================] - ETA: 0s - loss: 0.4911Epoch 14: Loss: 0.4911, Validation WER: 0.6861
773/773 [==============================] - 127s 160ms/step - loss: 0.4911
Epoch 15/50
773/773 [==============================] - ETA: 0s - loss: 0.4775Epoch 15: Loss: 0.4775, Validation WER: 0.6592
773/773 [==============================] - 127s 159ms/step - loss: 0.4775
Epoch 16/50
773/773 [==============================] - ETA: 0s - loss: 0.4656target:     <गाढम् दृढम् यथा स्यात् तथा मकरध्वजस्य कामस्य शत्>
prediction: <दाहम् दृढम् यथा स्यार्थिता मतरध्वजस्य कामस्य शत>

target:     <हस्तगतवत् दुर्ग्रहा इत्यर्थः>
prediction: <अस्त्रगतवत् दुर्ग्रहा इत्यर्थः>

target:     <प्रतिद्वन्द्विभवात् रिपूत्थात् विषादात् दैन्यात्>
prediction: <नैर्यात् प्रतिद्वन्द्विभवात् प्रतिनिधात् पितुः त>

target:     <सर्वदा सर्वथाभावात् क्वचित् किञ्चित् विवक्ष्यते>
prediction: <सर्वदा सर्वधा भावात् क्वचित्पिन्त्री च द्विपक्षत>

target:     <इति यत् तेन आसवपानेन मधुनिर्गमात् वसन्तापगमात्>
prediction: <इति यत् तेन आसवपादेन मधुनगर्गमात्र प्रसन्तापदमात>
```

Figure 4.1: Training process(Devanagari transcript)

4.1.1 Training Loss

The model's training loss exhibited a steady downward trend, reaching its lowest value of 0.38 during the training phase. This decrease reflects the model's ability to effectively learn patterns in the spectrogram data and improve its parameter optimization.

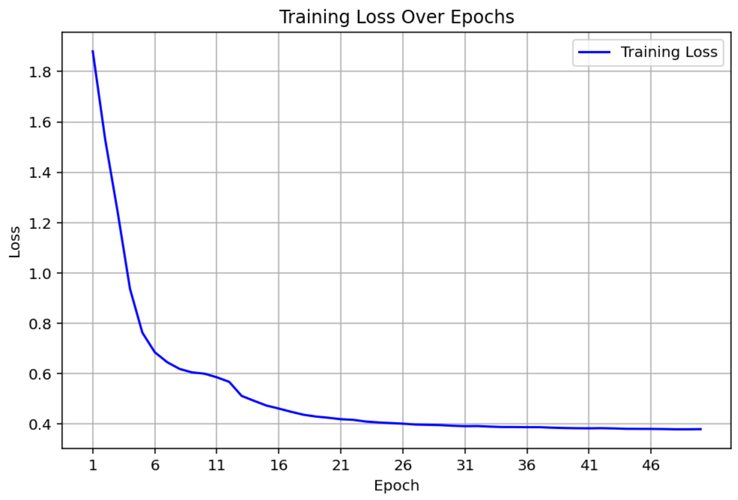


Figure 4.2: Training loss over epochs

4.1.2 Validation WER

The validation WER, which measures the percentage of transcription errors, also improved significantly over time. Starting from a high value, the WER gradually reduced and stabilized around 0.60, indicating the model's growing capacity to generalize across unseen validation data.

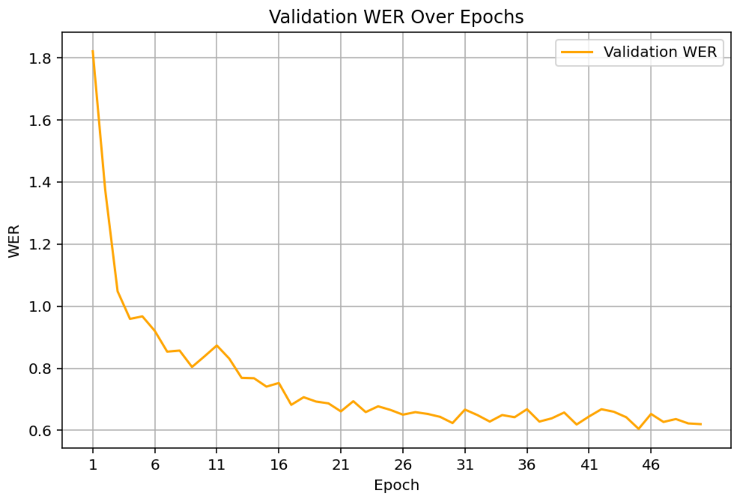
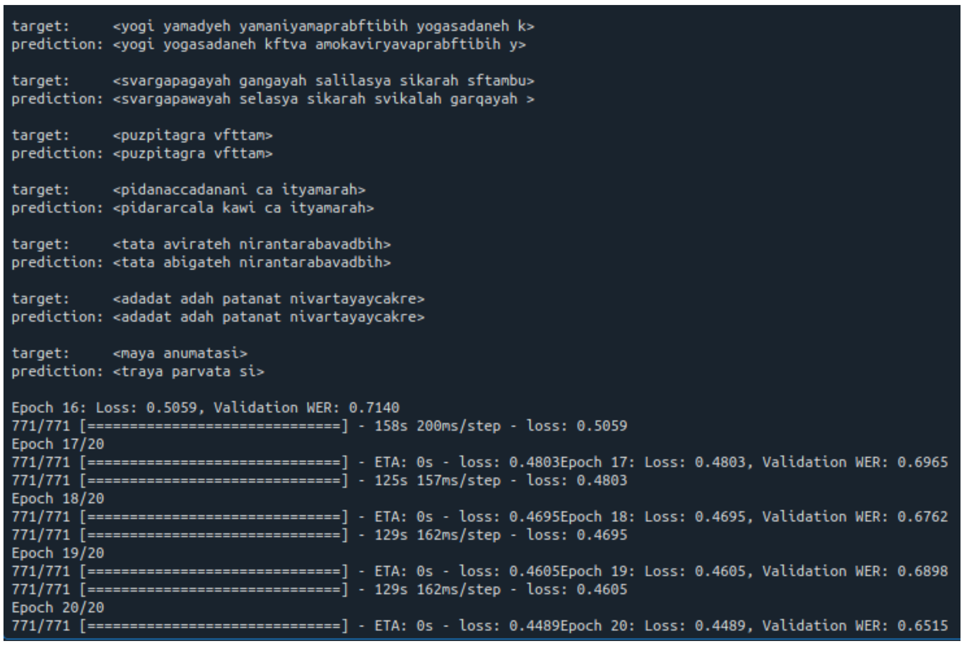


Figure 4.3: Validation WER over epochs

4.2 SLP1

SLP1 transcription offers an effective framework for low-resource language processing, with ShrutiGPT laying the groundwork for scalable Sanskrit speech recognition systems.



```
target:     <yogi yamadyeh yamaniyamaprabftibih yogasadaneh k>
prediction: <yogi yogasadaneh kftva amokaviryavaprabftibih y>

target:     <svargapagayah gangayah salilasya sikarah sftambu>
prediction: <svargapawayah selasya sikarah svikalah garqayah >

target:     <puzpitagra vfttam>
prediction: <puzpitagra vfttam>

target:     <pidanaccadanani ca ityamarah>
prediction: <pidararcala kawi ca ityamarah>

target:     <tata avirateh nirantarabavadbih>
prediction: <tata abigateh nirantarabavadbih>

target:     <adadat adah patanat nivartayaycakre>
prediction: <adadat adah patanat nivartayaycakre>

target:     <maya anumatasi>
prediction: <traya parvata si>

Epoch 16: Loss: 0.5059, Validation WER: 0.7140
771/771 [==============================] - 158s 200ms/step - loss: 0.5059
Epoch 17/20
771/771 [==============================] - ETA: 0s - loss: 0.4803Epoch 17: Loss: 0.4803, Validation WER: 0.6965
771/771 [==============================] - 125s 157ms/step - loss: 0.4803
Epoch 18/20
771/771 [==============================] - ETA: 0s - loss: 0.4695Epoch 18: Loss: 0.4695, Validation WER: 0.6762
771/771 [==============================] - 129s 162ms/step - loss: 0.4695
Epoch 19/20
771/771 [==============================] - ETA: 0s - loss: 0.4605Epoch 19: Loss: 0.4605, Validation WER: 0.6898
771/771 [==============================] - 129s 162ms/step - loss: 0.4605
Epoch 20/20
771/771 [==============================] - ETA: 0s - loss: 0.4489Epoch 20: Loss: 0.4489, Validation WER: 0.6515
```

Figure 4.4: Training process(SLP1 transcript)

4.2.1 Training Loss

The model achieved a lowest training loss of 0.48, which demonstrates its effective learning of Sanskrit speech-to-text mappings. This reduction in loss highlights the model's capacity to optimize its parameters while handling the intricate phonetic and grammatical structures of Sanskrit.

4.2.2 Validation WER

The validation WER improved consistently over epochs, stabilizing around 0.65. This metric reflects the model's ability to accurately transcribe unseen speech data using the SLP1 phonetic script. The use of SLP1 transcription appears to streamline the mapping between Sanskrit phonetics and text, contributing to this performance improvement.

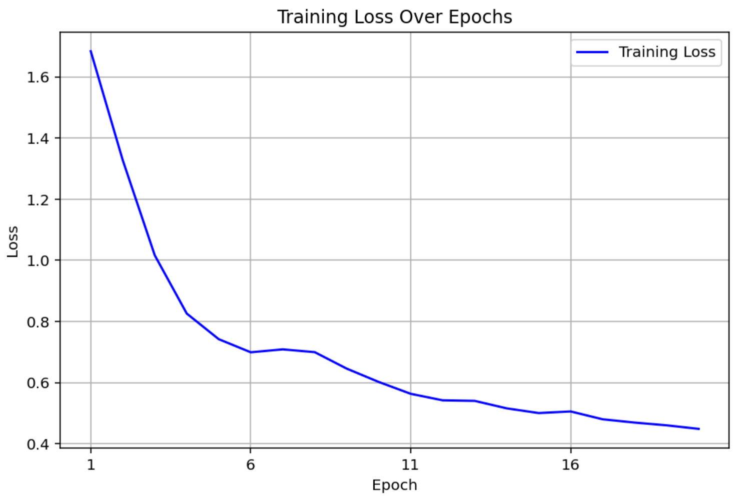


Figure 4.5: Training loss over epochs

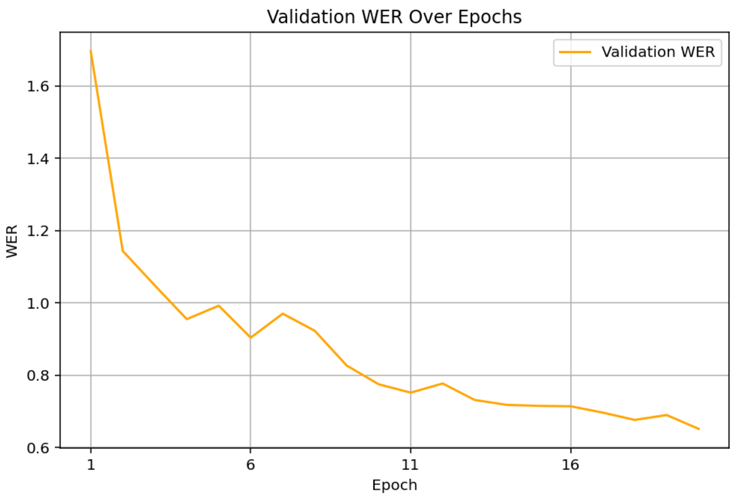


Figure 4.6: Validation WER over epochs

Chapter 5

Conclusion & Future Work

5.1 Conclusion

The ShrutiGPT project represents a significant step forward in the application of AI for Sanskrit speech recognition, showcasing how advanced transformer-based architectures can address challenges inherent to low-resource languages. By employing innovative techniques such as spectrogram-based feature extraction and SLP1 transcription, the model achieved improved recognition accuracy while simplifying computational requirements.

Both transcription systems offer complementary advantages for Sanskrit speech recognition. Devanagari transcription excels in overall accuracy and alignment with the language's grammatical structure, making it a strong choice for general-purpose applications. On the other hand, SLP1 provides a streamlined, script-agnostic approach that simplifies phonetic mapping, proving advantageous for specific tasks or constrained computational environments.

For future work, a hybrid approach combining the phonetic simplicity of SLP1 with the grammatical richness of Devanagari could further enhance model performance. Additionally, refining post-processing methods and expanding the dataset to include diverse sentence structures will improve the model's ability to handle complex and ambiguous inputs.

In conclusion, ShrutiGPT sets a strong foundation for Sanskrit speech recognition, demonstrating adaptability to both traditional and modern transcription systems.

5.1.1 Key Takeways:

1. **Spectrogram-Based Inputs:** The use of log-Mel spectrograms provided an effective way to capture phonetic and temporal patterns, crucial for speech-to-text transformation. This approach aligns closely with human auditory perception, enhancing the model's ability to decipher complex Sanskrit phonemes.

2. **SLP1 Transcription:** Transliteration into SLP1 significantly reduced token complexity compared to the Devanagari script. This streamlined tokenization, expedited training, and improved inference accuracy while retaining fidelity to Sanskrit's phonetic structure.

3. **Challenges and Opportunities:** Despite its success, the project encountered challenges, particularly in handling Sanskrit's grammatical intricacies and longer speech sequences. These issues underscore the need for future enhancements to extend the model's versatility.

5.2 Future Works:

To further advance the capabilities and robustness of Sanskrit speech recognition systems, the following directions are proposed:

1. **Dataset Expansion:** The current dataset primarily covers standard Sanskrit pronunciations. Future work should focus on creating larger, more diverse datasets that encompass variations in regional accents, speaker age groups, and recording environments. This expansion will improve the model's ability to generalize across real-world applications.

2. **Integration of Post-Processing Modules:** Sanskrit's grammatical structure is highly rule-based and syntactically rich. Integrating a grammar correction module that applies post-processing to the generated transcripts can help refine outputs to be syntactically accurate, making the system more useful for academic and practical applications.

3. **Exploration of Advanced Architectures:** Transformer architectures, while powerful, are not without limitations in handling long-range dependencies. Future work could explore advanced variants such as Conformers, which combine convolutional neural networks and self-attention mechanisms to better

model temporal dependencies in audio sequences. This can potentially enhance the recognition accuracy for longer and more complex Sanskrit phrases.

4. **Real-Time Deployment:** Future iterations of ShrutiGPT could focus on real-time recognition, making it suitable for applications such as live lectures, religious ceremonies, and conversational AI tools in Sanskrit.

ShrutiGPT is a testament to the transformative potential of AI in preserving and revitalizing linguistic heritage. By bridging the gap between ancient Sanskrit and modern accessibility, it paves the way for AI-driven solutions that honor and elevate cultural and linguistic diversity. While challenges remain, the project lays a robust foundation for future innovations in low-resource language processing, making Sanskrit more accessible to scholars, educators, and enthusiasts worldwide.

Bibliography

[1] Devaraja Adiga, Rishabh Kumar, Amrith Krishna, Preethi Jyothi, Ganesh Ramakrishnan, and Pawan Goyal. Automatic speech recognition in sanskrit: A new speech corpus and modelling insights. *arXiv preprint arXiv:2106.05852*, 2021.

[2] S Shashank Holla, TN Mahesh Kumar, Jeevan Revaneppa Hiretanad, KT Deepak, and AV Narasimhadhan. End-to-end speech recognition for low resource language sanskrit using self-supervised learning. In *2022 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, pages 148–152. IEEE, 2022.

[3] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[4] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.